Cisco IOS Debug Command Reference - Commands I through L

Americas Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
http://www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0883

CONTENTS

debug iapp through debug ip ftp

debug iapp

Use the debug iapp privileged EXEC command to begin debugging of IAPP operations. Use the **no**form of this command to stop the debug operation.

[no] debug iapp {**packets**| **event**| **error**}

Syntax Description

packets	Displays IAPP packets sent and received by the access point. Link test packets are not displayed
event	Displays significant IAPP events
error	Displays IAPP software and protocol errors

Command Default

This command has no default setting.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(11)JA	This command was introduced.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.

Examples

This example shows how to begin debugging of IAPP packets:

```
SOAP-AP# debug iapp packet
```

This example shows how to begin debugging of IAPP events:

```
SOAP-AP# debug iapp events
```

This example shows how to begin debugging of IAPP errors:

```
SOAP-AP# debug iapp errors
```

Related Commands

Command	Description
show debugging	Displays all debug settings

debug idmgr

To enable debugging for the identity manager (IDMGR), use the **debug idmgr** command in privileged EXEC mode. To disable debugging for the IDMGR, use the **no** form of this command.

debug idmgr {**core**| **data**| **db**| **elog**| **flow local**}

Syntax Description

core	Specifies debugging for the Layer 2 (L2) access core process flow.
data	Specifies debugging for data handling.
db	Specifies debugging for database interaction.
elog	Specifies debugging for event logging.
flow local	Specifies debugging for remote and local interaction.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
15.1(2)S	This command was introduced.

Usage Guidelines

You can use the **debug idmgr** command to debug errors such as missing or incorrect attributes in a session or Accounting, Authentication, and Authorization (AAA) records.

Usage Guidelines

The following is sample output from the **debug idmgr** command:

```
Router# debug idmgr core
IDMGR core process flow debugging is on
Router# debug idmgr data
IDMGR data handling debugging is on
Router# debug idmgr db
IDMGR database interaction debugging is on
R1# debug idmgr elog
IDMGR event logging debugging is on
R1# debug idmgr flow local
IDMGR local process flow debugging is on
2w6d: %SYS-5-CONFIG_I: Configured from console by console
2w6d: IDMGR: Enabled core flow debugging
2w6d: IDMGR: Enabled local flow debugging
2w6d: IDMGR: Enabled DB interaction debugging
2w6d: IDMGR:(07EC4890) got an Session Assert Request
2w6d: IDMGR:(07EC4890) Local processing Session Assert Request
2w6d: IDMGR: Set field session-handle 2281701385(88000009) in idmgr db record
```

```
2w6d: IDMGR: Set field aaa-unique-id 16(00000010) in idmgr db record
2w6d: IDMGR: Set field composite-key in idmgr db record
2w6d: IDMGR: Set field idmgr-data in idmgr db record
2w6d: IDMGR:(07EC4890) Adding new record 07640138 for session handle 88000009 to Session
DB
2w6d: IDMGR: Enabled core flow debugging
2w6d: IDMGR: Enabled local flow debugging
2w6d: IDMGR: Enabled DB interaction debugging
2w6d: IDMGR:(07EC4890) got an Session Update Event
2w6d: IDMGR:(07EC4890) Local processing Session Update Event
2w6d: IDMGR:(07EC4890) Search for session record
2w6d: IDMGR: Set field session-handle 2281701385(88000009) in search record
2w6d: IDMGR:(07EC4890) Found match for session handle 88000009
2w6d: IDMGR:(07EC4890) Found record in search get, returning 07640138
2w6d: IDMGR: releasing memory for search record field with type session-handle
2w6d: IDMGR: Set field idmgr-mask 4294967295(FFFFFFFF) in search record
2w6d: IDMGR: releasing memory for search record field with type idmgr-mask
Router#
2w6d: IDMGR:(07EC4890) Updating attribute authen-status in datalist
2w6d: IDMGR:(07EC4890) Updated record 07640138 for 88000009 to Session DB
```

Related Commands

Command	Description
show subscriber session	Displays information about subscriber sessions on an ISG.

debug if-mgr efp-ext

To enable debugging for the interface manager (IF-MGR) Ethernet flow point (EFP) extension, use the **debug if-mgr efp-ext**command in privileged EXEC mode. To turn off debugging for the IF-MGR EFP extension, use the **no**form of this command.

debug if-mgr {**errors**| **trace**} **efp-ext**

no debug if-mgr {**errors**| **trace**} **efp-ext**

Syntax Description

errors	Specifies debugging for IF-MGR EFP extension errors.
trace	Specifies debugging for IF-MGR EFP extension traces.

Command Default

Debugging is disabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(33)SRD1	This command was introduced.

Usage Guidelines

Before you issue the **debug if-mgr efp-ext**command, consider the high volume of output that debug commands usually generate and the amount of time the debugging operation may take.

Examples

The following example shows how to enable debugging for IF-MGR EFP extension errors:

```
Router> enable
Router# debug if-mgr errors efp-ext
Router#
```

debug ima

To display debugging messages for inverse multiplexing over AMT (IMA) groups and links, use the **debug ima**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ima

no debug ima

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging for IMA groups is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(5)T	This command was introduced.
12.0(5)XK	This command was modified.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Examples

The following example shows output when you enter the **debug ima** command while adding two ATM links to an IMA group. Notice that the group has not yet been created with the **interface atm** *slot* **/ima** *group-number* command, so the links are not activated yet as group members. However, the individual ATM links are deactivated.

```
Router# debug ima

IMA network interface debugging is on
Router# config terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# interface atm1/0
Router(config-if)# ima-group 1
Router(config-if)#
01:35:08:IMA shutdown atm layer of link ATM1/0
01:35:08:ima_clear_atm_layer_if ATM1/0
01:35:08:IMA link ATM1/0 removed in firmware
01:35:08:ima_release_channel:ATM1/0 released channel 0.
01:35:08:Bring up ATM1/4 that had been waiting for a free channel.
01:35:08:IMA:no shut the ATM interface.
01:35:08:IMA allocate_channel:ATM1/4 using channel 0.
01:35:08:IMA config_restart ATM1/4
01:35:08:IMA
adding link 0 to Group ATM1/IMA1ATM1/0 is down waiting for IMA group 1 to be activated
01:35:08:Link 0 was added to Group ATM1/IMA1
01:35:08:ATM1/0 is down waiting for IMA group 1 to be created.
01:35:08:IMA send AIS on link ATM1/0
01:35:08:IMA Link up/down Alarm:port 0, new status 0x10, old_status 0x1.
```

```
01:35:10:%LINK-3-UPDOWN:Interface ATM1/4, changed state to up
01:35:10:%LINK-3-UPDOWN:Interface ATM1/0, changed state to down
01:35:11:%LINEPROTO-5-UPDOWN:Line protocol on Interface ATM1/4, changed state to up
01:35:11:%LINEPROTO-5-UPDOWN:Line protocol on Interface ATM1/0, changed state to down
Router(config-if)# int atm1/1
Router(config-if)# ima-group 1
Router(config-if)#
01:37:19:IMA shutdown atm layer of link ATM1/1
01:37:19:ima_clear_atm_layer_if ATM1/1
01:37:19:IMA link ATM1/1 removed in firmware
01:37:19:ima_release_channel:ATM1/1 released channel 1.
01:37:19:Bring up ATM1/5 that had been waiting for a free channel.
01:37:19:IMA:no shut the ATM interface.
01:37:19:IMA allocate_channel:ATM1/5 using channel 1.
01:37:19:IMA config_restart ATM1/5
01:37:19:IMA adding link 1 to Group ATM1/IMA1ATM1/1 is down waiting for IMA group 1 to be
activated
01:37:19:Link 1 was added to Group ATM1/IMA1
01:37:19:ATM1/1 is down waiting for IMA group 1 to be created.
01:37:19:IMA send AIS on link ATM1/1
01:37:19:IMA Link up/down Alarm:port 1, new status 0x10, old_status 0x1.
Router(config-if)#
01:37:21:%LINK-3-UPDOWN:Interface ATM1/5, changed state to up
01:37:21:%LINK-3-UPDOWN:Interface ATM1/1, changed state to down
01:37:22:%LINEPROTO-5-UPDOWN:Line protocol on Interface ATM1/5, changed state to up
01:37:22:%LINEPROTO-5-UPDOWN:Line protocol on Interface ATM1/1, changed state to down
```

Related Commands

Command	Description
debug backhaul-session-manager set	Displays debugging messages for ATM errors, and reports specific problems such as encapsulation errors and errors related to OAM cells.
debug events	Displays debugging messages for ATM events, and reports specific events such as PVC setup completion, changes in carrier states, and interface rates.

debug installer

To enable debugs in the installer, use the **debug installer** command in Privileged EXEC mode. To disable debugging use the **no** form of the command.

debug installer [**all**| **process**| **issu**| **common**]

Syntax Description

all	Enables all installer debugs
process	Enables all the debugs inside Installer process
issu	Enables all the debugs inside the installer's Bash provisioning scripts
common	Enables all the debugs inside the installer common code

Command Default

No debugs enabled

Command Modes

Privileged EXEC

Command History

Release	Modification
IOS XE 3.2.0 SE	Command introduced.

Privileged EXEC

Usage Guidelines

The debug output for the above commands is displayed to the console and/or the IOS logging buffer.

It's always a good idea to turn on **debug installer all** when troubleshooting installer related problems

Examples

To enable all installer debugs, perform the following:

```
        infra-p2-3#debug installer all
All installer debugging is on
```

Related Commands

Command	Description
show version	To display information about the currently loaded software along with hardware and device information, use the **show version** command.

debug interface

To display interface descriptor block debugging messages, use the **debug interface**command in privileged EXEC mode. To disable the debugging messages, use the **no** form of this command.

debug interface *type number*

no debug interface *type number*

Syntax Description

type number	Interface type and number. In the case of an ATM interface, you get the following options once you enter the interface type and number: • **vc** --Displays information about the virtual circuit. • [*vpi* /]*vci*--Specifies the virtual channel identifier (VCI) or virtual path identifier/virtual channel identifier (VPI/VCI) pair, if the interface to be debugged is an ATM-encapsulated interface. Valid values for *vpi* are 0 to 255. Valid values for *vci* are 1 to 65535.

Command Default

By default, debugging messages are not enabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.3(4)T	This command was introduced.
12.0(26)S	This command was integrated into Cisco IOS Release 12.0(26)S.

Examples

The following is sample output from the **debug interface** command:

```
Router# debug interface ATM 1/0 vc 0/5
Condition 1 set
*Jan 31 19:36:38.399: ATM VC Debug: Condition 1, atm-vc 0/5 AT1/0 triggered, count 1
```

Related Commands

Command	Description
debug interface counters exceptions	Displays a message when a recoverable exceptional condition happens during the computation of the interface packet and data rate statistics.
debug interface counters protocol memory	Displays the memory operations (create and free) of protocol counters on interfaces and debugging messages during memory operations.

debug interface counters exceptions

To display a message when a recoverable exceptional condition happens during the computation of the interface packet and data rate statistics, use the **debug interface counters exceptions** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug interface counters exceptions

no debug interface counters exceptions

Syntax Description

This command has no arguments or keywords.

Command Default

By default, the debugging messages are not enabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.3(4)T	This command was introduced.
12.0(26)S	This command was integrated into Cisco IOS Release 12.0(26)S.

Usage Guidelines

Use the **debug interface counters exceptions** command to debug problems where the packet counter values or rates have unexpected values. The command helps to flag interfaces whose packet counter values have decreased in number. This condition can occur if a packet is counted and then dropped. This command helps you to determine if the input and output rate statistics are adjusted to display a zero value versus an unexpected value. It is also possible for zero values to be displayed if an interface is running at or close to its maximum capacity due to interface statistics being viewed as negative values.

This message is rate limited to one message per minute. If multiple interfaces are having unexpected counter statistic issues, then a message is displayed only for the first interface that experiences a problem within a minute.

Examples

The following is sample output from the **debug interface counters exceptions** command when backward-going counters are detected. The output is self-explanatory.

```
Router# debug interface counters exceptions
IF-4-BACKWARD_COUNTERS: Corrected for backward rx_bytes counters (561759 -> 526385) on
Multilink1
IF-4-BACKWARD_COUNTERS: Corrected for backward tx_bytes counters (288114 -> 268710) on
Multilink1
IF-4-BACKWARD_COUNTERS: Corrected for backward tx_bytes counters (2220 -> 0) on
Virtual-Access4
```

Related Commands

Command	Description
debug interface	Displays the interface descriptor block debugging messages.
debug interface counters protocol memory	Displays the memory operations (create and free) of protocol counters on interfaces and debugging messages during memory operations.

debug interface counters protocol memory

To display the memory operations (create and free) of protocol counters on interfaces and debugging messages during memory operations, use the **debug interface counters protocol memory**command in privileged EXEC mode. To disable the debugging output, use the **no** form of this command.

debug interface counters protocol memory

no debug interface counters protocol memory

Syntax Description

This command has no arguments or keywords.

Command Default

By default, the debugging messages are not enabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.3(4)T	This command was introduced.
12.0(26)S	This command was integrated into Cisco IOS Release 12.0(26)S.

Examples

The following is sample output from the **debug interface counters protocol memory**command. The output is self-explanatory.

```
Router# debug interface counters protocol memory
interface counter protocol memory operations debugging is on
*Jan 11 11:34:08.154: IDB_PROTO: Ethernet0/0 created CDP
*Jan 11 11:35:08.154: IDB_PROTO: Ethernet0/0 reset CDP
```

Related Commands

Command	Description
debug interface	Displays the interface descriptor block debugging messages.
debug interface counters exceptions	Displays a message when a recoverable exceptional condition happens during the computation of the interface packet and data rate statistics.

debug interface states

To display intermediary messages when an interface's state transitions, use the **debug interface states** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug interface states

no debug interface states

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging is disabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.4(11)T	This command was introduced.
12.2(44)S	This command was integrated into Cisco IOS Release 12.2(44)S.

Usage Guidelines

This command helps to debug interface state transition problems and includes the following interface state related message outputs:

- BRIDGE_ADJ--bridging database and Spanning tree protocol (STP) port state adjustment
- CSTATE_REQ--carrier state change request
- CSTATE_TMR--carrier timer state change
- LSTATE_REQ--line protocol state change request
- LSTATE_TMR--line protocol timer state change
- ROUTE_ADJ--route adjustment
- TRANS_ADJ--state transition adjustment

The debug information can be restricted to display state transitions on an interface basis using the **debug condition interface** command.

Caution Because the **debug interface states** command is a global debug command for all the interfaces in the router, in some cases such as with online insertion and removal (OIR) this command generates a substantial amount of output, depending on the number of interfaces hosted on the shared port adapter (SPA) or the line card. Use the **debug condition interface** command instead for debugging an interface state transition problem.

Examples

The following is sample output from the **debug interface states**command when the **shutdown** command is executed on an interface. The output is self-explanatory.

```
Router# debug interface states
interface state transitions debugging is on
Router# debug condition interface fast0/0
Condition 1 set
Router# config terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#
*Sep  1 12:24:46.294: [IDB Fa0/0 UARUYY] LSTATE_REQ: Entry
*Sep  1 12:24:46.294: [IDB Fa0/0 UARUYY] LSTATE_REQ: timers not running
*Sep  1 12:24:46.294: [IDB Fa0/0 UARUYY] LSTATE_REQ: Exit
Router(config)# interface fast0/0
Router(config-if)# shut
Router(config-if)#
*Sep  1 12:24:56.294: [IDB Fa0/0 UARUYY] LSTATE_REQ: Entry
*Sep  1 12:24:56.294: [IDB Fa0/0 UARUYY] LSTATE_REQ: timers not running
*Sep  1 12:24:56.294: [IDB Fa0/0 UARUYY] LSTATE_REQ: Exit
*Sep  1 12:24:57.162: [IDB Fa0/0 UARUYY] CSTATE_REQ: Entry, requested
state: A
*Sep  1 12:24:57.162: [IDB Fa0/0 UARUYY] CSTATE_REQ: starting ctimer (2000)
*Sep  1 12:24:57.162: [IDB Fa0/0 AURUYY] CSTATE_REQ: state assign
*Sep  1 12:24:57.162: [IDB Fa0/0 AURUYY] LSTATE_REQ: Entry
*Sep  1 12:24:57.162: [IDB Fa0/0 AURUYY] LSTATE_REQ: Exit
*Sep  1 12:24:57.162: [IDB Fa0/0 AURUYY] CSTATE_REQ: Exit
*Sep  1 12:24:57.162: [IDB Fa0/0 AURUYY] CSTATE_REQ: Entry, requested
state: A
*Sep  1 12:24:57.162: [IDB Fa0/0 AURUYY] CSTATE_REQ: state assign
*Sep  1 12:24:57.162: [IDB Fa0/0 AURUYY] LSTATE_REQ: Entry
*Sep  1 12:24:57.162: [IDB Fa0/0 AURUYY] LSTATE_REQ: Exit
*Sep  1 12:24:57.162: [IDB Fa0/0 AURUYY] CSTATE_REQ: Exit
*Sep  1 12:24:57.166: [IDB Fa0/0 AURUnY] TRANS_ADJ: Entry
*Sep  1 12:24:57.166: [IDB Fa0/0 AURUnn] TRANS_ADJ: propagating change
to subifs
*Sep  1 12:24:57.170: [IDB Fa0/0 AURUnn] TRANS_ADJ: Exit
*Sep  1 12:24:57.170: [IDB Fa0/0 AURUnn] ROUTE_ADJ: Entry
*Sep  1 12:24:57.170: [IDB Fa0/0 AURUnn] ROUTE_ADJ: Exit
*Sep  1 12:24:57.170: [IDB Fa0/0 AURUnn] BRIDGE_ADJ: Entry
*Sep  1 12:24:57.170: [IDB Fa0/0 AURUnn] BRIDGE_ADJ: Exit
*Sep  1 12:24:59.162: [IDB Fa0/0 AURUnn] CSTATE_TMR: Entry
*Sep  1 12:24:59.162: [IDB Fa0/0 AURUnn] CSTATE_TMR: netidb=Fa0/0,
linestate: n
*Sep  1 12:24:59.162: [IDB Fa0/0 AURUnn] LSTATE_REQ: Entry
*Sep  1 12:24:59.162: [IDB Fa0/0 AURUnn] LSTATE_REQ: timers not running
*Sep  1 12:24:59.162: [IDB Fa0/0 AURUnn] LSTATE_REQ: starting lineproto
timer
*Sep  1 12:24:59.162: [IDB Fa0/0 AURUnn] LSTATE_REQ: Exit
*Sep  1 12:24:59.162: [IDB Fa0/0 AURUnn] CSTATE_TMR: transition detected
*Sep  1 12:24:59.162: %ENTITY_ALARM-6-INFO: ASSERT INFO Fa0/0 Physical
Port Administrative State Down
*Sep  1 12:24:59.162: [IDB Fa0/0 AURUnn] TRANS_ADJ: Entry
*Sep  1 12:24:59.162: [IDB Fa0/0 AURUnn] TRANS_ADJ: Exit
*Sep  1 12:24:59.162: [IDB Fa0/0 AURUnn] CSTATE_TMR: Exit
*Sep  1 12:25:00.162: [IDB Fa0/0 AURUnn] LSTATE_TMR: Entry
*Sep  1 12:25:00.162: [IDB Fa0/0 AURUnn] LSTATE_TMR: not spoofing,
current state: n
*Sep  1 12:25:00.162: [IDB Fa0/0 AURUnn] LSTATE_TMR: informing line
```

```
state transitions
*Sep  1 12:25:00.162: [IDB Fa0/0 AURUnn] TRANS_ADJ: Entry
*Sep  1 12:25:00.162: [IDB Fa0/0 AURUnn] TRANS_ADJ: Exit
*Sep  1 12:25:00.162: [IDB Fa0/0 AURUnn] ROUTE_ADJ: Entry
*Sep  1 12:25:00.162: [IDB Fa0/0 AURUnn] ROUTE_ADJ: Exit
*Sep  1 12:25:00.162: [IDB Fa0/0 AURUnn] LSTATE_TMR: Exit
```

Related Commands

Command	Description
debug condition interface	Limits output for some debug commands on the basis of the interface, VC, or VLAN.

debug interface(vasi)

To display debugging information for the VRF-Aware Service Infrastructure (VASI) interface descriptor block, use the **debug interface** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug interface {**vasileft**| **vasiright**} *number*

no debug interface {**vasileft**| **vasiright**} *number*

Syntax Description

vasileft	Displays information about vasileft interface.
vasiright	Displays information about vasiright interface.
number	Identifier of the VASI interface. The range is from 1 to 256.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS XE Release 2.6	This command was introduced.

Examples

The following is sample output from the **debug interface**command:

```
Router# debug interface vasileft 100
Condition 1 set
```

Related Commands

interface (vasi)	Configures a VASI virtual interface.
debug adjacency (vasi)	Displays debugging information for the VASI adjacency.
debug vasi	Displays debugging information for the VASI.
show vasi pair	Displays the status of a VASI pair.

debug iosd issu

To enable all the debugs inside the IOS issu_iosd and iosvrp_issu_upgrade subsystems, use the **debug iosd issu** command in Privileged EXEC mode. To disable debugging use the **no** form of the command.

debug iosd issu

Command Default

Debugs not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
IOS XE 3.2.0 SE	Command introduced.

Privileged EXEC

Usage Guidelines

No command variables

It's always a good idea to turn on **debug iosd issu** when troubleshooting installer related problems

Related Commands

Command	Description
show version	To display information about the currently loaded software along with hardware and device information, use the **show version** command.

debug ip access-list hash-generation

To display debugging information about access control list (ACL) hash-value generation (for ACL Syslog entries), use the **debug ip access-list hash-generation** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip access-list hash-generation

no debug ip access-list hash-generation

Syntax Description

This command has no arguments or keywords.

Command Default

This command is disabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.4(22)T	This command was introduced.

Usage Guidelines

Use this command when configuring an access control entry (ACE) to view the router-generated hash values for the ACE.

This command displays the input and output for the hash-generation mechanism. The input is the ACE text and ACL name. The output is an MD5 algorithm-derived, 4-byte value.

Examples

The following example shows sample debug output displayed when configuring ACL hash-value generation.

Note: The example in this section shows sample output for a numbered access list. However, you can configure ACL hash-value generation for both numbered and named access lists, and for both standard and extended access lists.

```
Router#
*Aug 9 00:24:31.765: %SYS-5-CONFIG_I: Configured from console by console
Router# debug ip access-list hash-generation
Syslog hash code generation debugging is on
Router# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)# ip access-list logging hash-generation
Router(config)# access-list 101 permit tcp host 10.1.1.1 host 10.1.1.2 log
Router(config)#
*Aug 9 00:25:31.661: %IPACL-HASHGEN: Hash Input: 101 extended permit 6 host 20.1.1.1 host
20.1.1.2 Hash Output: 0xA363BB54
Router(config)# exit
Router#
```

Related Commands

Command	Description
ip access-list logging hash-generation	Enables the generation of hash-values for access control entries in the system messaging logs.
show ip access-list	Displays the contents of all current access lists.

debug ip access-list intstats

To display information about whether or not the interface-level statistics of an access list were created, updated, cleared or deleted successfully, use the **debug ip access-list intstats**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip access-list intstats

no debug ip access-list intstats

Syntax Description

This command has no arguments or keywords.

Command Default

No default behaviors or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.4(6)T	This command was introduced.

Examples

The following is sample output from the **debug ip access-list intstats**command:

```
Router# debug ip access-list intstats
Router# enable
Router# configure terminal
Router(config)#interface e0/0
Router(config-if)#ip access-group 100 in
*Oct 29 08:52:16.763: IPACL-INTSTATS: ACL swsb created
*Oct 29 08:52:16.763: IPACL-INTSTATS: ACL header stats structure created
*Oct 29 08:52:16.763: IPACL-INTSTATS: I/P stats table created
*Oct 29 08:52:16.763: IPACL-INTSTATS: Statsid bitmap created
*Oct 29 08:52:16.763: IPACL-INTSTATS: Done with static ACEs
Router(config-if)#ip access-group 100 out
*Oct 29 08:52:19.435: IPACL-INTSTATS: O/P stats table created
*Oct 29 08:52:19.435: IPACL-INTSTATS: Done with static ACEs
```

debug ip access-list turboacl

To display debugging information about turbo access control lists (ACLs), use the **debug ip access-list turboacl**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip access-list turboacl

no debug ip access-list turboacl

Syntax Description

This command has no arguments or keywords.

Command Default

No default behaviors or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2	This command was introduced.
12.3(3)T	This command was modified to include support for turbo ACLs.
12.3(4)T	This command was integrated into Cisco IOS Release 12.3(4)T.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

The **debug ip access-list turboacl**command is useful for debugging problems associated with turbo ACLs. Turbo ACLs compile the ACLs into a set of lookup tables, while maintaining the first packet matching requirements. Packet headers are used to access these tables in a small, fixed, number of lookups, independent of the existing number of ACL entries.

Examples

The following is sample output from the **debug ip access-list turboacl**command:

```
Router# debug ip access-list turboacl

*Aug 20 00:41:17.843 UTC:Miss at index 73, 19
*Aug 20 00:41:17.843 UTC:Adding dynamic entry, update = 1
*Aug 20 00:41:17.843 UTC:Miss at index 21, 39
*Aug 20 00:41:17.847 UTC:Adding dynamic entry, update = 1
*Aug 20 00:41:17.847 UTC:Miss at index 116, 42
*Aug 20 00:41:17.851 UTC:Adding dynamic entry, update = 1
*Aug 20 00:41:17.851 UTC:Miss at index 119, 28
*Aug 20 00:41:17.851 UTC:Adding dynamic entry, update = 1
*Aug 20 00:41:17.855 UTC:Miss at index 116, 42
*Aug 20 00:41:17.855 UTC:Adding dynamic entry, update = 1
*Aug 20 00:41:17.855 UTC:Miss at index 92, 20
*Aug 20 00:41:17.855 UTC:Adding dynamic entry, update = 1
*Aug 20 00:41:17.855 UTC:Miss at index 119, 28
```

```
*Aug 20 00:41:17.855 UTC:Adding dynamic entry, update = 1
*Aug 20 00:41:17.855 UTC:Miss at index 56, 29
*Aug 20 00:41:17.859 UTC:Adding dynamic entry, update = 1
*Aug 20 00:41:17.859try, update = 1
*Aug 20 00:41:19.959 UTC:Miss at index 29, 41
*Aug 20 00:41:19.959 UTC:Adding dynamic entry, update = 1
*Aug 20 00:41:19.959 UTC:Miss at index 29, 38
```

The table below describes the significant fields shown in the display.

Table 1: debug ip access-list turboacl Field Descriptions

Field	Description
Aug 20 00:41:17.843 UTC	Date and Coordinated Universal Time (UTC) the command was used to debug the turbo ACL.
Miss at index 73, 19	Location in the compiled access list tables where a new packet lookup does not match an existing entry.
Adding dynamic entry, update = 1	Action taken to add a new entry in the compiled access list tables as a result of a packet being processed.

debug ip admission consent

To display authentication proxy consent page information on the router, use the **debug ip admission consent** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip admission consent [**events**| **errors**| **messages**]

no debug ip admission consent

Syntax Description

errors	(Optional) Displays only error messages.
events	(Optional) Displays only event-related messages.
messages	(Optional) Displays only packet-related messages.

Command Default

If an option is not selected, all debug messages are displayed.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.4(15)T	This command was introduced.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.

Examples

```
Router# debug ip admission consent errors

IP Admission Consent Errors debugging is on
Router# debug ip admission consent events

IP Admission Consent Events debugging is on
Router# debug ip admission consent messages

IP Admission Consent Messages debugging is on
Router#
Router# show debugging
IP Admission Consent:
IP Admission Consent Errors debugging is on
IP Admission Consent Events debugging is on
IP Admission Consent Messages debugging is on
```

debug ip admission eapoudp

To display information about Extensible Authentication Protocol over User Datagram Protocol (UDP) (EAPoUDP) network admission control events, use the **debug ip admission eapoudp**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip admission eapoudp

no debug ip admission eapoudp

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging is not enabled.

Command Modes

Privileged EXEC #

Command History

Release	Modification
12.3(8)T	This command was introduced.
12.2(33)SXI	This command was integrated into Cisco IOS Release 12.2(33)SXI.

Examples

The following sample output from the **debug ip admission eapoudp** command shows information about network admission control using EAPoUDP. In the command output, the term “posture” refers to the credentials (for example, antivirus state or version of Cisco IOS software) of the host system.

```
Router# debug ip admission eapoudp

Posture validation session created for client mac= 0001.027c.f364 ip= 10.0.0.1
Total Posture sessions= 1 Total Posture Init sessions= 1
*Apr  9 19:39:45.684: %AP-6-POSTURE_START_VALIDATION: IP=10.0.0.1|
Interface=FastEthernet0/0.420
*Apr  9 19:40:42.292: %AP-6-POSTURE_STATE_CHANGE: IP=10.0.0.1| STATE=POSTURE ESTAB
*Apr  9 19:40:42.292: auth_proxy_posture_parse_aaa_attributes:
CiscoDefined-ACL name= #ACSACL#-IP-HealthyACL-40921e54
Apr  9 19:40:42.957: %AP-6-POSTURE_POLICY: Apply access control list
(xACSACLx-IP-HealthyACL-40921e54) policy for host (10.0.0.1)
```

The fields in the display are self-explanatory.

Related Commands

Command	Description
show ip admission	Displays IP admission control cache entries or the running admission control configuration.

debug ip auth-proxy

To display the authentication proxy configuration information on the router, use the **debug ip auth-proxy**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip auth-proxy {**detailed**| **ftp**| **function-trace**| **object-creation**| **object-deletion**| **telnet**| **timers**}

no debug ip auth-proxy

Syntax Description

detailed	Displays details of the TCP events during an authentication proxy process. The details are generic to all FTP, HTTP, and Telnet protocols.
ftp	Displays FTP events related to the authentication proxy.
function-trace	Displays the authentication proxy functions.
object-creation	Displays additional entries to the authentication proxy cache.
object-deletion	Displays deletion of cache entries for the authentication proxy.
telnet	Displays Telnet-related authentication proxy events.
timers	Displays authentication proxy timer-related events.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(5)T	This command was introduced.
12.3(1)	The **detailed** keyword was added.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

Use the **debug ip auth-proxy** command to display authentication proxy activity.

Note The **function-trace** debugging information provides low-level software information for Cisco technical support representatives. No output examples are provided for this keyword option.

Examples

The following examples illustrate the output of the **debug ip auth-proxy** command. In these examples, debugging is on for object creations, object deletions, HTTP, and TCP.

In this example, the client host at 192.168.201.1 is attempting to make an HTTP connection to the web server located at 192.168.21.1. The HTTP debugging information is on for the authentication proxy. The output shows that the router is setting up an authentication proxy entry for the login request:

```
00:11:10: AUTH-PROXY creates info:
cliaddr - 192.168.21.1, cliport - 36583
seraddr - 192.168.201.1, serport - 80
ip-srcaddr 192.168.21.1
pak-srcaddr 0.0.0.0
```

Following a successful login attempt, the debugging information shows the authentication proxy entries created for the client. In this example, the client is authorized for SMTP (port 25), FTP data (port 20), FTP control (port 21), and Telnet (port 23) traffic. The dynamic access control list (ACL) entries are included in the display.

```
00:11:25:AUTH_PROXY OBJ_CREATE:acl item 61AD60CC

00:11:25:AUTH-PROXY OBJ_CREATE:create acl wrapper 6151C7C8 -- acl item 61AD60CC
00:11:25:AUTH-PROXY Src 192.168.162.216 Port [0]
00:11:25:AUTH-PROXY Dst 192.168.162.220 Port [25]
00:11:25:AUTH_PROXY OBJ_CREATE:acl item 6151C908

00:11:25:AUTH-PROXY OBJ_CREATE:create acl wrapper 6187A060 -- acl item 6151C908
00:11:25:AUTH-PROXY Src 192.168.162.216 Port [0]
00:11:25:AUTH-PROXY Dst 192.168.162.220 Port [20]
00:11:25:AUTH_PROXY OBJ_CREATE:acl item 61A40B88

00:11:25:AUTH-PROXY OBJ_CREATE:create acl wrapper 6187A0D4 -- acl item 61A40B88
00:11:25:AUTH-PROXY Src 192.168.162.216 Port [0]
00:11:25:AUTH-PROXY Dst 192.168.162.220 Port [21]
00:11:25:AUTH_PROXY OBJ_CREATE:acl item 61879550

00:11:25:AUTH-PROXY OBJ_CREATE:create acl wrapper 61879644 -- acl item 61879550
00:11:25:AUTH-PROXY Src 192.168.162.216 Port [0]
00:11:25:AUTH-PROXY Dst 192.168.162.220 Port [23]
```

The next example shows the debug output following a **clear ip auth-proxy cache** command to clear the authentication entries from the router. The dynamic ACL entries are removed from the router.

```
00:12:36:AUTH-PROXY OBJ_DELETE:delete auth_proxy cache 61AD6298
00:12:36:AUTH-PROXY OBJ_DELETE:delete create acl wrapper 6151C7C8 -- acl item 61AD60CC
00:12:36:AUTH-PROXY OBJ_DELETE:delete create acl wrapper 6187A060 -- acl item 6151C908
00:12:36:AUTH-PROXY OBJ_DELETE:delete create acl wrapper 6187A0D4 -- acl item 61A40B88
00:12:36:AUTH-PROXY OBJ_DELETE:delete create acl wrapper 61879644 -- acl item 61879550
```

The following example shows the timer information for a dynamic ACL entry. All times are expressed in milliseconds. The *first laststart* is the time that the ACL entry is created relative to the startup time of the router. The *lastref* is the time of the last packet to hit the dynamic ACL relative to the startup time of the router. The *exptime* is the next expected expiration time for the dynamic ACL. The *delta* indicates the remaining time before the dynamic ACL expires. After the timer expires, the debugging information includes a message indicating that the ACL and associated authentication proxy information for the client have been removed.

```
00:19:51:first laststart 1191112

00:20:51:AUTH-PROXY:delta 54220 lastref 1245332 exptime 1251112
00:21:45:AUTH-PROXY:ACL and cache are removed
```

The following example is sample output with the **detailed** keyword enabled:

```
00:37:50:AUTH-PROXY:proto_flag=5, dstport_index=1
00:37:50: SYN SEQ 245972 LEN 0
00:37:50:dst_addr 192.168.127.2 src_addr 192.168.27.1 dst_port 21 src_port 4347
00:37:50:AUTH-PROXY:auth_proxy_half_open_count++ 1
00:37:50:AUTH-PROXY:proto_flag=5, dstport_index=1
00:37:50: ACK 1820245643 SEQ 245973 LEN 0
00:37:50:dst_addr 192.168.127.2 src_addr 192.168.27.1 dst_port 21 src_port 4347
00:37:50:clientport 4347 state 0
00:37:50:AUTH-PROXY:incremented proxy_proc_count=1
00:37:50:AUTH-PROXY:proto_flag=5, dstport_index=1
00:37:50: ACK 1820245674 SEQ 245973 LEN 0
00:37:50:dst_addr 192.168.127.2 src_addr 192.168.27.1 dst_port 21 src_port 4347
00:37:50:clientport 4347 state 0
00:37:57:AUTH-PROXY:proto_flag=5, dstport_index=1
00:37:57: PSH ACK 1820245674 SEQ 245973 LEN 16
00:37:57:dst_addr 192.168.127.2 src_addr 192.168.27.1 dst_port 21 src_port 4347
00:37:57:clientport 4347 state 0
00:37:57:AUTH-PROXY:proto_flag=5, dstport_index=1
00:37:57: ACK 1820245699 SEQ 245989 LEN 0
00:37:57:dst_addr 192.168.127.2 src_addr 192.168.27.1 dst_port 21 src_port 4347
00:37:57:clientport 4347 state 0
00:38:01:AUTH-PROXY:proto_flag=5, dstport_index=1
00:38:01: PSH ACK 1820245699 SEQ 245989 LEN 16
00:38:01:dst_addr 192.168.127.2 src_addr 192.168.27.1 dst_port 21 src_port 4347
00:38:01:clientport 4347 state 0
00:38:01:AUTH-PROXY:Authenticating user ryan
00:38:01:AUTH-PROXY:Session state is INIT.Not updating stats
00:38:01:AUTH-PROXY:Session state is INIT.Not updating stats
00:38:01:AUTH-PROXY:Sent AAA request successfully
00:38:01:AUTH-PROXY:Sent password successfully
00:38:01:AUTH-PROXY:processing authorization data
00:38:01:AUTH-PROXY:Sending accounting start.unique-id 2
00:38:01:AUTH-PROXY:Session state is INIT.Not updating stats
00:38:01:AUTH-PROXY:Session state is INIT.Not updating stats
00:38:01:AUTH-PROXY:wait complete on watched boolean stat=0
00:38:01:AUTH-PROXY:src ip addr is 192.168.127.2, dstaddr=192.168.27.1
00:38:01: SYN ACK 2072458992 SEQ 4051022445 LEN 0
00:38:01:AUTH-PROXY:src ip addr is 192.168.127.2, dstaddr=192.168.27.1
00:38:01: PSH ACK 2072458992 SEQ 4051022446 LEN 49
00:38:02:AUTH-PROXY:src ip addr is 192.168.127.2, dstaddr=192.168.27.1
00:38:02: ACK 2072459003 SEQ 4051022495 LEN 0
00:38:02:AUTH-PROXY:src ip addr is 192.168.127.2, dstaddr=192.168.27.1
00:38:02: PSH ACK 2072459003 SEQ 4051022495 LEN 33
00:38:02:AUTH-PROXY:src ip addr is 192.168.127.2, dstaddr=192.168.27.1
00:38:02: ACK 2072459014 SEQ 4051022528 LEN 0
00:38:02:AUTH-PROXY:src ip addr is 192.168.127.2, dstaddr=192.168.27.1
00:38:02: PSH ACK 2072459014 SEQ 4051022528 LEN 26
00:38:03:AUTH-PROXY:proto_flag=5, dstport_index=1
00:38:03: ACK 1820245725 SEQ 246005 LEN 0
00:38:03:dst_addr 192.168.127.2 src_addr 192.168.27.1 dst_port 21 src_port 4347
00:38:03:clientport 4347 state 3
7200b#
```

Related Commands

Command	Description
show debug	Displays the debug options set on the router.

debug ip auth-proxy ezvpn

To display information related to proxy authentication behavior for web-based activation, use the **debug ip auth-proxy ezvpn**command in privileged EXEC mode. To turn off debugging, use the **no** form of this command.

debug ip auth-proxy ezvpn

no debug ip auth-proxy ezvpn

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging is not turned on.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.3(14)T	This command was introduced.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2SX	This command is supported in the Cisco IOS 12.2SX family of releases. Support in a specific 12.2SX release is dependent on your feature set, platform, and platform hardware.

Usage Guidelines

Caution Using this command may result in considerable output if simultaneous authentications are taking place.

Examples

The following is output from the **debug ip auth-proxy ezvpn** command. The output displays the proxy authentication behavior of a web-based activation.

```
Router# debug ip auth-proxy ezvpn
*Dec 20 20:25:11.006: AUTH-PROXY: New request received by EzVPN WebIntercept from
10.4.205.205
*Dec 20 20:25:17.150: AUTH-PROXY:GET request received
*Dec 20 20:25:17.150: AUTH-PROXY:Authentication scheme is 401
*Dec 20 20:25:17.362: AUTH-PROXY:Authorization information not present in GET request
*Dec 20 20:25:17.362: AUTH-PROXY: Allocated on credinfo for connect at 0x81EF1A84
*Dec 20 20:25:17.362: AUTH-PROXY: Posting CONNECT request to EzVPN
*
Dec 20 20:25:17.362: EZVPN(tunnel22): Received CONNECT from 10.4.205.205!
*Dec 20 20:25:17.366: EZVPN(tunnel22): Current State: CONNECT_REQUIRED
*Dec 20 20:25:17.366: EZVPN(tunnel22): Event: CONNECT
```

The output in the display is self-explanatory.

Related Commands

Command	Description
xauth userid mode	Specifies how the Cisco Easy VPN Client handles Xauth requests or prompts from the server.

debug ip bgp

To display information related to processing of the Border Gateway Protocol (BGP), use the **debug ip bgp command in**privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip bgp [*ip-address*| **addpath| dampening| events| in| keepalives| out| updates| vpnv4| mpls**]

no debug ip bgp [*ip-address*| **addpath| dampening| events| in| keepalives| out| updates| vpnv4| mpls**]

Cisco 10000 Series Router

debug ip bgp [*ip-address*| **dampening| events| in| keepalives| out| updates| vpnv4| mpls| all| groups| import| ipv4| ipv6**]

no debug ip bgp [*ip-address*| **dampening| events| in| keepalives| out| updates| vpnv4| mpls| all| groups| import| ipv4| ipv6**]

Syntax Description

ip-address	(Optional) The BGP neighbor IP address.
addpath	(Optional) Displays BGP additional path events.
dampening	(Optional) Displays BGP dampening.
events	(Optional) Displays BGP events.
in	(Optional) Displays BGP inbound information.
keepalives	(Optional) Displays BGP keepalives.
out	(Optional) Displays BGP outbound information.
updates	(Optional) Displays BGP updates.
vpnv4	(Optional) Displays Virtual Private Network version 4 (VPNv4) Network Layer Reachability Information (NLRI).
mpls	(Optional) Displays Multiprotocol Label Switching (MPLS) information.
all	(Optional) Displays all address family information.
groups	(Optional) Displays BGP configuration and update groups information.
import	(Optional) Displays BGP import routes to a VPN routing and forwarding (VRF) instance across address family information.

ipv4	(Optional) Displays BGP IPv4 address family information.
ipv6	(Optional) Displays BGP IPv6 address family information.

Command Modes

Privileged EXEC(#)

Command History

Release	Modification
12.0(5)T	This command was introduced.
12.0(21)ST	This command was integrated into Cisco IOS Release 12.0(21)ST. The **mpls** keyword was added.
12.0(22)S	This command was integrated into Cisco IOS Release 12.0(22)S.
12.2(13)T	The **mpls** keyword was added.
12.2(17b)SXA	This command was integrated into Cisco IOS Release 12.2(17b)SXA.
12.0(27)S	The command output was modified to show explicit-null label information.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.
12.2(33)SRE	This command was modified. The **addpath** keyword was added.
12.2(33)XNE	This command was integrated into Cisco IOS Release 12.2(33)XNE.
Cisco IOS XE Release 2.5	This command was integrated into Cisco IOS XE Release 2.5.

Usage Guidelines

Use this command with the **updates** and **mpls** keywords to display explicit-null label information. The optional arguments in, out, keepalives, updates, and events provide verbose output to the debug ip bgp command. The sequence in which the optional arguments are provided affects the behavior of the command. The non peer specific commands override the peer-specific commands.

Examples

Following is the sample output from the **debug ip bgp** command used with vpnv4 keyword:

```
Router# debug ip bgp vpnv4
```

```
03:47:14:vpn:bgp_vpnv4_bnetinit:100:2:10.0.0.0/8
03:47:14:vpn:bnettable add:100:2:10.0.0.0/8
03:47:14:vpn:bestpath_hook route_tag_change for vpn2:10.0.0.0/255.0.0.0(ok)
03:47:14:vpn:bgp_vpnv4_bnetinit:100:2:10.0.0.0/8
03:47:14:vpn:bnettable add:100:2:10.0.0.0/8
03:47:14:vpn:bestpath_hook route_tag_change for vpn2:10.0.0.0/255.0.0.0(ok)
03:47:14:vpn:bgp_vpnv4_bnetinit:100:2:10.0.0.0/8
03:47:14:vpn:bnettable add:100:2:10.0.0.0/8
03:47:14:vpn:bestpath_hook route_tag_chacle ip bgp *nge for vpn2:10.0.0.0/255.0.0.0(ok)
```

The following example shows sample output, including the explicit-null label, from the d**ebug ip bgp updates** and the **debug ip bgp mpls** commands:

```
Router# debug ip bgp updates
BGP updates debugging is on
Router# debug ip bgp mpls
BGP MPLS labels debugging is on

Router#
01:33:53: BGP(0): route 10.10.10.10/32 up
01:33:53: BGP(0): nettable_walker 10.10.10.10/32 route sourced locally
01:33:53: BGP: adding MPLS label to 10.10.10.10/32
01:33:53: BGP: check on 10.10.10.10/8 in LDP - ok
01:33:53: BGP: label imp-null allocated via LDP
01:33:53: BGP-IPv4: send exp-null label for 10.10.10.10/32
01:33:53: BGP-IPv4: Send prefix 10.10.10.10/32, label exp-null    !explicit-null label being
 sent
01:33:53: BGP(0): 10.10.10.11 send UPDATE (format) 10.10.10.10/32, next 10.10.10.12, metric
 0, path , mpls label 0 !label value is 0
01:33:53: BGP(0): updgrp 1 - 10.10.10.12 enqueued 1 updates, average/maximum size (bytes)
61/61
```

Following example shows a sample output from the debug ip bgp command when various arguments are provided in a particular sequence:

```
Router# debug ip bgp 209.165.200.225
Router# debug ip bgp 209.165.200.225 updates
Router# debug ip bgp keepalives
Router# debug ip bgp events
Router# debug ip bgp in
Router# debug ip bgp out

Router# show debug
IP routing:
  BGP debugging is on (outbound) for address family: IPv4 Unicast
  BGP events debugging is on
  BGP keepalives debugging is on
  BGP updates debugging is on (outbound) for address family: IPv4 Unicast
```

The behavior of the command changes when the arguments are provided in a different sequence

```
Router# debug ip bgp keepalives
Router# debug ip bgp events
Router# debug ip bgp in
Router# debug ip bgp out
Router# debug ip bgp 209.165.200.225
Router# debug ip bgp 209.165.200.225 updates

Router# show debug
IP routing:
  BGP debugging is on for neighbor 209.165.200.225 for address family: IPv4 Unicast
  BGP events debugging is on for neighbor 209.165.200.225
  BGP keepalives debugging is on for neighbor 209.165.200.225 for address family: IPv4
Unicast
  BGP updates debugging is on for neighbor 209.165.200.225 for address family: IPv4 Unicast
```

debug ip bgp groups

To display information related to the processing of Border Gateway Protocol (BGP) update-groups, use the **debug ip bgp update** privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip bgp groups [*index-group*| *ip-address*]

no debug ip bgp groups

Syntax Description

index-group	(Optional) Specifies that update-group debugging information for the corresponding index number will be displayed. The range of update-group index numbers is from 1 to 4294967295.
ip-address	(Optional) Specifies that update-group debugging information for a single peer will be displayed.

Command Default

No information about BGP update-groups is displayed.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(24)S	This command was introduced.
12.2(18)S	This command was integrated into Cisco IOS Release 12.2(18)S.
12.3(4)T	This command was integrated into Cisco IOS Release 12.3(4)T.
12.2(27)SBC	This command was integrated into Cisco IOS Release 12.2(27)SBC.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

The output of this command displays information about update-group calculations and the addition and removal of update-group members. Information about peer-groups, peer-policy, and peer-session templates will also be displayed in the output of this command as neighbor configurations change.

Note The output of this command can be very verbose. This command should not be deployed in a production network unless you are troubleshooting a problem.

When a change to outbound policy occurs, the router automatically recalculates update-group memberships and applies the changes by triggering an outbound soft reset after a 1-minute timer expires. This behavior is designed to provide the network operator with time to change the configuration if a mistake is made. You can manually enable an outbound soft reset before the timer expires by entering the **clear ip bgp** *ip-address* **soft out** command.

Note In Cisco IOS Release 12.0(25)S, 12.3(2)T, and prior releases the update group recalculation delay timer is set to 3 minutes.

Examples

The following sample output from the **debug ip bgp groups** command shows that peering has been established with neighbor 10.4.9.8 and update-group calculations are occurring for this member:

```
Router# debug ip bgp groups

5w4d: BGP-DYN(0): Comparing neighbor 10.4.9.8 flags 0x0 cap 0x0 and updgrp 1 fl0
5w4d: BGP-DYN(0): Created update-group(0) flags 0x0 cap 0x0 from neighbor 10.4.0
5w4d: BGP-DYN(0): Adding  neighbor 10.4.9.8 flags 0x0 cap 0x0, to update-group 0
5w4d: %BGP-5-ADJCHANGE: neighbor 10.4.9.8 Up
```

The following sample output from the **debug ip bgp groups** command shows the recalculation of update-groups after the **clear ip bgp groups** command was issued:

```
Router# debug ip bgp groups

5w4d: %BGP-5-ADJCHANGE: neighbor 10.4.9.5 Down User reset
5w4d: BGP-DYN(0): Comparing neighbor 10.4.9.5 flags 0x0 cap 0x0 and updgrp 2 fl0
5w4d: BGP-DYN(0): Update-group 2 flags 0x0 cap 0x0 policies same as 10.4.9.5 fl0
5w4d: %BGP-5-ADJCHANGE: neighbor 10.4.9.8 Down User reset
5w4d: BGP-DYN(0): Comparing neighbor 10.4.9.8 flags 0x0 cap 0x0 and updgrp 2 fl0
5w4d: BGP-DYN(0): Update-group 2 flags 0x0 cap 0x0 policies same as 10.4.9.8 fl0
5w4d: %BGP-5-ADJCHANGE: neighbor 10.4.9.21 Down User reset
5w4d: BGP-DYN(0): Comparing neighbor 10.4.9.21 flags 0x0 cap 0x0 and updgrp 1 f0
5w4d: BGP-DYN(0): Update-group 1 flags 0x0 cap 0x0 policies same as 10.4.9.21 f0
5w4d: %BGP-5-ADJCHANGE: neighbor 10.4.9.5 Up
5w4d: %BGP-5-ADJCHANGE: neighbor 10.4.9.21 Up
5w4d: %BGP-5-ADJCHANGE: neighbor 10.4.9.8 Up
```

The table below describes the significant fields shown in the display.

Table 2: debug ip bgp groups Field Descriptions

Field	Description
%BGP-5-ADJCHANGE:	A BGP neighbor has come Up or gone Down. The IP address of the neighbor is specified in the output string.
BGP-DYN(0):	This line is displayed when a neighbor adjacency is established. The BGP dynamic update group algorithm analyzes the policies of the new neighbor and then adds the neighbor to the appropriate BGP update group.

Related Commands

Command	Description
clear ip bgp	Resets a BGP connection or session.
clear ip bgp update-group	Clears BGP update-group member sessions.
show ip bgp replication	Displays BGP update-group replication statistics.
show ip bgp update-group	Displays information about BGP update-groups.

debug ip bgp igp-metric ignore

To display information related to the system ignoring the Interior Gateway Protocol (IGP) metric during best path selection, use the **debug ip bgp igp-metric ignore**command in privileged EXEC mode. To disable such debugging output, use the **no** form of the command.

debug ip bgp igp-metric ignore

no debug ip bgp igp-metric ignore

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS XE Release 3.4S	This command was introduced.

Usage Guidelines

You might use this command if the path you expected to be chosen as the best path at the shadow RR was not chosen as such. That could be because the **bgp bestpath igp-metric ignore** command makes the best path algorithm choose the same best path as the primary RR if they are not co-located.

Examples

The following example turns on debugging of events related to the system ignoring the IGP metric during bestpath selection:

```
Router# debug ip bgp igp-metric ignore
```

Related Commands

Command	Description
bgp bestpath igp-metric ignore	Specifies that the system ignore the Interior Gateway Protocol (IGP) metric during best path selection.

debug ip bgp import

To display debugging information related to importing IPv4 prefixes from the BGP global routing table into a VRF table or exporting from a VRF table into the BGP global table, use the **debug ip bgp import** command in privileged EXEC mode. To disable the display of such debugging information, use the **no** form of this command.

debug ip bgp import {**events**| **updates** [*access-list*| *expanded-access-list*]}

no debug ip bgp import {**events**| **updates** [*access-list*| *expanded-access-list*]}

Syntax Description

events	Displays messages related to IPv4 prefix import events.
updates	Displays messages related to IPv4 prefix import updates.
access-list	(Optional) Number of the access list used to filter debugging messages. The range is from 1 to 199.
expanded-access-list	(Optional) Number of the expanded access list used to filter debugging messages. The range is from 1300 to 2699.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.0(29)S	This command was introduced.
12.2(25)S	This command was integrated into Cisco IOS Release 12.2(25)S.
12.3(14)T	This command was integrated into Cisco IOS Release 12.3(14)T.
12.2(27)SBC	This command was integrated into Cisco IOS Release 12.2(27)SBC.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.
15.2(4)S	This command was modified. The output now includes information for the BGP Support for IP Prefix Export from a VRF to the Global Table feature.
Cisco IOS XE Release 3.7S	This command was modified. The output now includes information for the BGP Support for IP Prefix Export from a VRF to the Global Table feature.

Usage Guidelines

Use this command to display debugging information related to the BGP Support for IP Prefix Import from Global Table into a VRF Table feature or the BGP Support for IP Prefix Export from a VRF Table into Global Table feature. The former feature provides the capability to import IPv4 unicast prefixes from the global routing table into a Virtual Private Network (VPN) routing/forwarding (VRF) instance table using an import route map. The latter feature provides the capability to export IPv4 or IPv6 prefixes from a VRF table into the global table using an export route map.

Examples

The following example configures IPv4 prefix import debugging messages for both import events and import updates to be displayed on the console of the router:

```
Router# debug ip bgp import events
```

```
BGP import events debugging is on
Router# debug ip bgp import updates
BGP import updates debugging is on for access list 3
00:00:50: %BGP-5-ADJCHANGE: neighbor 10.2.2.2 Up
00:01:06: BGP: reevaluate IPv4 Unicast routes in VRF academic
00:01:06: BGP: 0 routes available (limit: 1000)
00:01:06: BGP: import IPv4 Unicast routes to VRF academic
00:01:06: BGP(2)-VRF(academic): import pfx 100:1:10.30.1.0/24 via 10.2.2.2
00:01:06: BGP: accepted 8 routes (limit: 1000)
00:01:06: BGP: reevaluate IPv4 Multicast routes in VRF multicast
00:01:06: BGP: 0 routes available (limit: 2)
00:01:06: BGP: import IPv4 Multicast routes to VRF multicast
00:01:06: %BGP-4-AFIMPORT: IPv4 Multicast prefixes imported to multicast vrf reached the
limit 2
00:01:06: BGP: accepted 2 routes (limit: 2)
00:01:06: BGP: reevaluate IPv4 Unicast routes in VRF BLUE
00:01:06: BGP: 0 routes available (limit: 1000)
00:01:06: BGP: import IPv4 Unicast routes to VRF BLUE
00:01:06: BGP: accepted 3 routes (limit: 1000)
```

The table below describes the significant fields shown in the display.

Table 3: debug ip bgp import Field Descriptions

Field	Description
BGP: accepted 2 routes (limit: 2)	Number of routes imported into the VRF, and the default or user-defined prefix import limit.
BGP: reevaluate IPv4 Unicast routes in VRF BLUE	Prefix was imported during BGP convergence and is being reevaluated for the next scan cycle.
BGP: 0 routes available (limit: 1000)	Number of routes available from the import source, and the default or user-defined prefix import limit.
BGP: import IPv4 Unicast routes to VRF BLUE	Import map and prefix type (unicast or multicast) that is being imported into the specified VRF.

The following is a sample debug message for the IP prefix export from a VRF table to global table:

```
Device# debug ip bgp import events
```

```
*Jul 12 10:06:48.357: BGP GBL-IMP: vpn1:VPNv4 Unicast:base 1:1:192.168.4.0/24
-> global:IPv4 Unicast:base Creating importing net.
  4.4.4.4 (metric 11) from 4.4.4.4 (4.4.4.4)
   Origin IGP, metric 0, localpref 100, valid, internal, best
   Extended Community: RT:1:1
   mpls labels in/out nolabel/16
```

Related Commands

Command	Description
clear ip bgp	Resets a BGP connection.
export map (VRF table to global table)	Exports IP prefixes from a VRF table to the global routing table based on a route map.
import map	Imports IP prefixes from the global routing table to a VRF table based on a route map.

debug ip bgp range

To display debugging information related to Border Gateway Protocol (BGP) dynamic subnet range neighbors, use the **debug ip bgp range** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip bgp range [detail]

no debug ip bgp range

Syntax Description

detail	(Optional) Specifies that detailed debugging information about BGP dynamic subnet range neighbors will be displayed.

Command Default

No debugging information about BGP dynamic subnet range neighbors is displayed.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(33)SXH	This command was introduced.
15.0(1)S	This command was integrated into Release 15.0(1)S.
Cisco IOS XE Release 3.1S	This command was integrated into Cisco IOS XE Release 3.1S.
15.2(4)S	This command was integrated into Cisco IOS Release 15.2(4)S.

Usage Guidelines

The output of this command displays information about the identification and creation of BGP dynamic subnet range neighbors. BGP dynamic neighbors are configured using a range of IP addresses and BGP peer groups. After a subnet range is configured for a BGP peer group, and a TCP session is initiated for an IP address in the subnet range, a new BGP neighbor is dynamically created as a member of that group. The new BGP neighbor will inherit any configuration or templates for the group.

Examples

The following output shows that the **debug ip bgp range** command has been entered and a BGP neighbor at 192.168.3.2 has been dynamically created using the subnet range 192.168.0.0/16. This new neighbor is a member of the peer group named group192.

```
Router# debug ip bgp range
bgprange_debug = 1, sense = 1
BGP dynamic Range debugging is on
!
```

```
*Mar 26 20:05:13.251: BGP:DN: Created a new neighbor *192.168.3.2
in range 192.168.0.0/16, peer-group group192,count = 1
```

The following sample output from the **debug ip bgp range detail** command shows more detailed debugging of the addition of dynamic BGP neighbors:

```
Router# debug ip bgp range detail
bgprange_debug = 1, sense = 1
BGP dynamic Range debugging is on with detail (Dynamic Range neighbors details only)
!
*Mar 26 20:09:12.311: BGP:DN: ACCEPT an OPEN from 192.168.1.2 valid range
0x32123D8:192.168.0.0/16,tcb 0x32114C0
!
*Mar 26 20:09:12.331: BGP: 192.168.1.2 passive open to 192.168.1.1
*Mar 26 20:09:12.331: BGP:DN: ACCEPTED an OPEN from 192.168.1.2 valid range
0x32123D8:192.168.0.0/16,tcb 0x3494040
!
*Mar 26 20:09:12.331: BGP:DN: Created a new neighbor *192.168.1.2
in range 192.168.0.0/16, peer-group group192,count = 2
```

The table below describes the significant field shown in the display.

Table 4: debug ip bgp range Field Descriptions

Field	Description
BGP:DN:	A potential dynamic BGP neighbor has been identified as opening a TCP session with an IP address in a subnet associated with a BGP peer group. BGP accepts the session and creates a new neighbor. The new neighbor becomes a member of the peer group associated with its subnet range.

Related Commands

Command	Description
bgp listen	Configures BGP dynamic neighbor parameters.
clear ip bgp peer-group	Clears BGP peer group member sessions.
show ip bgp peer-group	Displays information about BGP peer groups.

debug ip bgp sso

To display Border Gateway Protocol (BGP)-related stateful switchover (SSO) events or debugging information for BGP-related interactions between the active Route Processor (RP) and the standby RP, use the **debug ip bgp sso**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip bgp sso {**events**| **transactions**} **[detail]**

no debug ip bgp sso {**events**| **transactions**} **[detail]**

Syntax Description

events	Displays BGP-related SSO failures.
transactions	Displays debugging information for failed BGP-related interactions between the active RP and the standby RP.
detail	(Optional) Displays detailed debugging information about successful BGP-related SSO operations and successful BGP-related interactions between the active and the standby RP.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(28)SB	This command was introduced.
12.2(33)SRB1	This command was integrated into Cisco IOS Release 12.2(33)SRB1.
15.0(1)S	This command was integrated into Cisco IOS Release 15.0(1)S.
Cisco IOS XE 3.1S	This command was integrated into Cisco IOS XE Release 3.1S.

Usage Guidelines

The **debug ip bgp sso**command is used to display BGP-related SSO events or debugging information for BGP-related interactions between the active RP and the standby RP. This command is useful for monitoring or troubleshooting BGP sessions on a provider edge (PE) router during an RP switchover or during a planned In-Service Software Upgrade (ISSU).

Examples

The following is sample output from the **debug ip bgp sso** command with the **events** keyword. The following output indicates that the 10.34.32.154 BGP session is no longer SSO capable.

```
*Mar 28 02:29:43.526: BGPSSO: 10.34.32.154 reset SSO and decrement count
```

Tip Use the **show ip bgp vpnv4 all neighbors**command to display the reason that the SSO-capable BGP session has been disabled.

The following is sample output from the **debug ip bgp sso** command with the **transactions** keyword. The following output shows an SSO notification indicating that the SSO capability is pending for 602 BGP neighbors. This notification is generated as the state between the active and standby RP is being synchronized during the bulk synchonization phase of SSO initialization. During this phase, the Transmission Control Blocks (TCBs) must be synchronized with the TCBs on the standby RP before SSO initialization is complete.

```
*Mar 28 02:32:12.102: BGPSSO: tcp sso notify pending for 602 nbrs
```

debug ip bgp updates

To display information about the processing of Border Gateway Protocol (BGP) updates, use the **debug ip bgp updates** command in privileged EXEC mode. To disable the display of BGP update information, use the **no** form of this command.

debug ip bgp updates [*access-list*| *expanded-access-list*] [**in**| **out**] **[events] [refresh]**

no debug ip bgp updates [*access-list*| *expanded-access-list*] [**in**| **out**] **[events] [refresh]**

Syntax Description

access-list	(Optional) Number of access list used to filter debugging messages. The range that can be specified is from 1 to 199.
expanded-access-list	(Optional) Number of expanded access lists used to filter debugging messages. The range that can be specified is from 1300 to 2699.
in	(Optional) Specifies debugging messages for inbound BGP update information.
out	(Optional) Specifies debugging messages for outbound BGP update information.
events	(Optional) Specifies debugging messages for BGP update events.
refresh	(Optional) Specifies debugging messages for BGP update refresh.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.0(5)T	This command was introduced.
12.2(18)S	This command was integrated into Cisco IOS Release 12.2(18)S.
12.2(27)SBC	This command was integrated into Cisco IOS Release 12.2(27)SBC.
12.2(31)SB	This command was modified. The **refresh** keyword was added.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Examples

The following is sample output from the **debug ip bgp updates** command. The output shows that the BGP session was cleared between neighbor 10.4.9.21 and the local router 10.4.9.4. There are no field description tables for this command because the debugging output from this command depends on the subsequent commands that are entered.

```
Router# debug ip bgp updates
5w2d: %SYS-5-CONFIG_I: Configured from console by console
5w2d: BGP: 10.4.9.21 went from Idle to Active
5w2d: BGP: 10.4.9.21 open active, delay 7032ms
5w2d: BGP: 10.4.9.21 open active, local address 10.4.9.4
5w2d: BGP: 10.4.9.21 went from Active to OpenSent
5w2d: BGP: 10.4.9.21 sending OPEN, version 4, my as: 101
5w2d: BGP: 10.4.9.21 send message type 1, length (incl. header) 45
5w2d: BGP: 10.4.9.21 rcv message type 1, length (excl. header) 26
5w2d: BGP: 10.4.9.21 rcv OPEN, version 4
5w2d: BGP: 10.4.9.21 rcv OPEN w/ OPTION parameter len: 16
5w2d: BGP: 10.4.9.21 rcvd OPEN w/ optional parameter type 2 (Capability) len 6
5w2d: BGP: 10.4.9.21 OPEN has CAPABILITY code: 1, length 4
5w2d: BGP: 10.4.9.21 OPEN has MP_EXT CAP for afi/safi: 1/1
5w2d: BGP: 10.4.9.21 rcvd OPEN w/ optional parameter type 2 (Capability) len 2
5w2d: BGP: 10.4.9.21 OPEN has CAPABILITY code: 128, length 0
5w2d: BGP: 10.4.9.21 OPEN has ROUTE-REFRESH capability(old) for all address-fams
5w2d: BGP: 10.4.9.21 rcvd OPEN w/ optional parameter type 2 (Capability) len 2
5w2d: BGP: 10.4.9.21 OPEN has CAPABILITY code: 2, length 0
5w2d: BGP: 10.4.9.21 OPEN has ROUTE-REFRESH capability for all address-families
5w2d: BGP: 10.4.9.21 went from OpenSent to OpenConfirm
5w2d: BGP: 10.4.9.21 went from OpenConfirm to Established
5w2d: %BGP-5-ADJCHANGE: neighbor 10.4.9.21 Up
5w2d: BGP(0): 10.4.9.21 computing updates, afi 0, neighbor version 0, table ver0
5w2d: BGP(0): 10.4.9.21 update run completed, afi 0, ran for 0ms, neighbor vers1
5w2d: BGP(0): 10.4.9.21 initial update completed
```

The following is sample output from the **debug ip bgp updates out**command. The output shows that the local router is sending updates with the cost community:

```
Router# debug ip bgp updates out
*Mar 15 01:41:23.515:BGP(0):10.0.0.5 computing updates, afi 0, neighbor version 0, table
version 64, starting at 0.0.0.0
*Mar 15 01:41:23.515:BGP(0):10.0.0.5 send UPDATE (format) 0.0.0.0/0, next 10.0.0.2, metric
0, path , extended community Cost:igp:1:100
*Mar 15 01:41:23.515:BGP(0):10.0.0.5 send UPDATE (format) 10.2.2.0/24, next 10.20.20.10,
metric 0, path 10, extended community Cost:igp:8:22
*Mar 15 01:41:23.515:BGP(0):10.0.0.5 send UPDATE (format) 10.13.13.0/24, next 10.0.0.8,
metric 0, path
```

The following is sample output from the **debug ip bgp updates in**command. The output shows that the local router is receiving updates with the cost community:

```
Router# debug ip bgp updates in
*Jan  6 01:27:09.111:BGP(2):10.0.0.8 rcvd UPDATE w/ attr:nexthop 10.0.0.8, origin ?,
localpref 100, metric 0, path 10, extended community RT:100:1 Cost:igp:10:10
Cost:igp:11:11
```

debug ip bgp vpnv4 checkpoint

To display the events for the Virtual Routing and Forwarding (VRF) checkpointing system between the active and standby Route Processors, use the debug ip bgp vpnv4 checkpoint command in privileged EXEC mode. To disable the display of these events, use the **no** form of this command.

debug ip bgp vpnv4 checkpoint

no debug ip bgp vpnv4 checkpoint

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(25)S	This command was introduced.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB and implemented on the Cisco 10000 series router.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.

Examples

The following example shows command output on the active Route Processor:

```
Router# debug ip bgp vpnv4 checkpoint
3d18h: %HA-5-SYNC_NOTICE: Config sync started.
3d18h: vrf-nsf: vrf vpn2 tableid 1 send OK
3d18h: vrf-nsf: vrf tableid bulk sync complete msg send OK
3d18h: vrf-nsf: CF send ok
3d18h: vrf-nsf: CF send ok
3d18h: %HA-5-SYNC_NOTICE: Config sync completed.
3d18h: %HA-5-SYNC_NOTICE: Standby has restarted.
3d18h: %HA-5-MODE: Operating mode is sso, configured mode is sso.
```

Related Commands

Command	Description
debug ip bgp vpnv4 nsf	Displays the nonstop forwarding events for the VRF table-id synchronization subsystem between the active and standby route processors.

debug ip bgp vpnv4 nsf

To display the nonstop forwarding events for the VRF table-id synchronization subsystem between the active and standby Route Processors, use the debug ip bgp vpnv4 nsf command in privileged EXEC mode. To disable the display of these events, use the **no** form of this command.

debug ip bgp vpnv4 nsf

no debug ip bgp vpnv4 nsf

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(25)S	This command was introduced.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB and implemented on the Cisco 10000 series router.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.

Examples

The following example shows the command output on the active Route Processor:

```
Router# debug ip bgp vpnv4 nsf
MPLS VPN NSF Processing debugging is on
Router(config)# ip vrf vpn3
3d18h: vrf-nsf: vrf vpn3 tableid 2 send rpc OK
Router(config-vrf)# no ip vrf vpn3
% IP addresses from all interfaces in VRF vpn3 have been removed
3d18h: vrf-nsf:  rx vrf tableid delete complete msg, tid = 2, name = vpn3
```

The following example shows the command output on the standby Route Processor:

```
Router# debug ip bgp vpnv4 nsf
MPLS VPN NSF Processing debugging is on
00:05:21: vrf-nsf: rx vrf tableid rpc msg, tid = 2, name = vpn3
% IP addresses from all interfaces in VRF vpn3 have been removed
00:06:22: vrf-nsf: vrf vpn3 tableid 2 , delete complete, send OK
```

Related Commands

Command	Description
debug ip bgp vpnv4 checkpoint	Display the events for the VRF checkpointing system between the active and standby Route Processors.

debug ip bgp vpnv4 unicast

To display debugging messages for Virtual Private Network version 4 (VPNv4) unicast routes, use the **debug ip bgp vpnv4 unicast** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip bgp vpnv4 unicast {**checkpoint| csc| import| keepalives| labelmode| updates**}

no debug ip bgp vpnv4 unicast {**checkpoint| csc| import| keepalives| labelmode| updates**}

Syntax Description

checkpoint	Displays virtual routing and forwarding (VRF) nonstop forwarding (NSF) checkpoint messages and events.
csc	Displays VRF processing messages for a Carrier Supporting Carrier (CSC) VPN.
import	Displays VRF import processing messages.
keepalives	Displays Border Gateway Protocol (BGP) keepalives.
labelmode	Displays VRF label mode processing.
updates	Displays BGP updates processing for Unicast VPNv4 address family.

Command Default

Debugging of VPNv4 unicast routes is not enabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.0(5)T	This command was introduced.
XE Release 2.2	The **labelmode** keyword was added.
12.2(33)SRD	This command was integrated into Cisco IOS Release 12.2(33)SRD.

Examples

The following example enables debugging of MPLS VPN label mode processing:

```
Router# debug ip bgp vpnv4 unicast labelmode
MPLS VPN Label mode processing debugging is on
```

```
Router# config terminal

Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# mpls label mode all-vrfs protocol bgp-vpnv4 per-vrf
% This command is an unreleased and unsupported feature
Router(config)#
*Oct 18 11:35:01.159: vpn: changing the label mode (Enable: per-vrf) for all-vrfs
*Oct 18 11:35:01.459: vpn: label mode change, bnet walk complete.
*Oct 18 11:35:01.459: BGP: VPNv4 Unicast label mode changed
Router(config)#^Z
Router#
*Oct 18 11:35:21.995: %SYS-5-CONFIG_I: Configured from console by console
Router# show debug

Tag VPN:
  MPLS VPN Label mode processing debugging is on
Router#
```

Related Commands

Command	Description
show ip vrf detail	Displays assigned label mode for the VRF.

debug ip bgp vpnv6 unicast

To display debugging messages for Virtual Private Network version 6 (VPNv6) unicast routes, use the **debug ip bgp vpnv6 unicast** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip bgp vpnv6 unicast {**csc**| **import**| **keepalives**| **labelmode**| **topology**| **updates**}

no debug ip bgp vpnv6 unicast {**csc**| **import**| **keepalives**| **labelmode**| **topology**| **updates**}

Syntax Description

csc	Displays VPN routing and forwarding (VRF) processing messages for a Carrier Supporting Carrier (CSC) VPN.
import	Displays VRF import processing messages.
keepalives	Displays Border Gateway Protocol (BGP) keepalives.
labelmode	Displays VRF label mode processing.
topology	Displays the routing topology instance.
updates	Displays BGP updates processing for the unicast VPNv6 address family.

Command Default

Debugging of VPNv6 unicast routes is not enabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(33)SRD	This command was introduced.

Examples

The following example enables debugging of MPLS VPN label mode processing:

```
Router# debug ip bgp vpnv6 unicast labelmode
MPLS VPN Label mode processing debugging is on
Router# config terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# mpls label mode vrf vpn1 protocol bgp-vpnv6 per-vrf
% Command accepted but obsolete, unreleased or unsupported; see documentation.
Router(config)#
6d03h: vpn: changing the label mode (Enable: per-vrf) for vrf vpn1, address family ipv6
6d03h: vpn: setting pervrfaggr label 18 for vrf vpn1:2001:DB8:1:2::/96
```

```
6d03h: vpn: setting pervrfaggr label 18 for vrf vpn1:2001:DB8:2::1/128
6d03h: vpn: pervrfaggr, withdraw and free local label 19 for vpn1:2001:DB8:CE1::1/128
6d03h: vpn: setting pervrfaggr label 18 for vrf vpn1:2001:DB8:CE1::1/128
6d03h: vpn: label mode change, bnet walk complete.
6d03h: BGP: VPNv6 Unicast label mode changed
Router(config)# end
```

Related Commands

Command	Description
show vrf detail	Displays assigned label mode for the VRF.

debug ip casa affinities

To display debugging messages for affinities, use the **debug ip casa affinities**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip casa affinities

no debug ip casa affinities

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging for affinities is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(5)T	This command was introduced.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Examples

The following is sample output from the **debug ip casa affinities**command:

```
Router# debug ip casa affinities
16:15:36:Adding fixed affinity:
16:15:36:    10.10.1.1:54787 -> 10.10.10.10:23 proto = 6
16:15:36:Updating fixed affinity:
16:15:36:    10.10.1.1:54787 -> 10.10.10.10:23 proto = 6
16:15:36:    flags = 0x2, appl addr = 10.10.3.2, interest = 0x5/0x100
16:15:36:    int ip:port = 10.10.2.2:1638, sequence delta = 0/0/0/0
16:15:36:Adding fixed affinity:
16:15:36:    10.10.10.10:23 -> 10.10.1.1:54787 proto = 6
16:15:36:Updating fixed affinity:
16:15:36:    10.10.10.10:23 -> 10.10.1.1:54787 proto = 6
16:15:36:    flags = 0x2, appl addr = 0.0.0.0, interest = 0x3/0x104
16:15:36:    int ip:port = 10.10.2.2:1638, sequence delta = 0/0/0/0
```

The table below describes the significant fields shown in the display.

Table 5: debug ip casa affinities Field Descriptions

Field	Description
Adding fixed affinity	Adding a fixed affinity to affinity table.
Updating fixed affinity	Modifying a fixed affinity table with information from the services manager.

Field	Description
flags	Bit field indicating actions to be taken on this affinity.
fwd addr	Address to which packets will be directed.
interest	Services manager that is interested in packets for this affinity.
int ip:port	Services manager port to which interest packets are sent.
sequence delta	Used to adjust TCP sequence numbers for this affinity.

debug ip casa packets

To display debugging messages for packets, use the **debug ip casa packets**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip casa packets

no debug ip casa packets

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging for packets is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(5)T	This command was introduced.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Examples

The following is sample output from the **debug ip casa packets**command:

```
Router# debug ip casa packets
16:15:36:Routing CASA packet - TO_MGR:
16:15:36:    10.10.1.1:55299 -> 10.10.10.10:23 proto = 6
16:15:36:    Interest Addr:10.10.2.2   Port:1638
16:15:36:Routing CASA packet - FWD_PKT:
16:15:36:    10.10.1.1:55299 -> 10.10.10.10:23 proto = 6
16:15:36:    Fwd Addr:10.10.3.2
16:15:36:Routing CASA packet - TO_MGR:
16:15:36:    10.10.10.10:23 -> 10.10.1.1:55299 proto = 6
16:15:36:    Interest Addr:10.10.2.2   Port:1638
16:15:36:Routing CASA packet - FWD_PKT:
16:15:36:    10.10.10.10:23 -> 10.10.1.1:55299 proto = 6
16:15:36:    Fwd Addr:0.0.0.0
16:15:36:Routing CASA packet - TICKLE:
16:15:36:    10.10.10.10:23 -> 10.10.1.1:55299 proto = 6
16:15:36:    Interest Addr:10.10.2.2   Port:1638  Interest Mask:SYN
16:15:36:    Fwd Addr:0.0.0.0
16:15:36:Routing CASA packet - FWD_PKT:
16:15:36:    10.10.1.1:55299 -> 10.10.10.10:23 proto = 6
16:15:36:    Fwd Addr:10.10.3.2
```

The table below describes the significant fields shown in the display.

Table 6: debug ip casa packets Field Descriptions

Field	Description
Routing CASA packet - TO_MGR	Forwarding Agent is routing a packet to the services manager.
Routing CASA packet - FWD_PKT	Forwarding Agent is routing a packet to the forwarding address.
Routing CASA packet - TICKLE	Forwarding Agent is signaling services manager while allowing the packet in question to take the appropriate action.
Interest Addr	Services manager address.
Interest Port	Port on the services manager where packet is sent.
Fwd Addr	Address to which packets matching the affinity are sent.
Interest Mask	Services manager that is interested in packets for this affinity.

debug ip casa wildcards

To display debugging messages for wildcards, use the **debug ip casa wildcards**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip casa wildcards

no debug ip casa wildcards

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging for wildcards is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(5)T	This command was introduced.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Examples

The following is sample output from the **debug ip casa wildcards**command:

```
Router# debug ip casa wildcards
16:13:23:Updating wildcard affinity:
16:13:23:    10.10.10.10:0 -> 0.0.0.0:0 proto = 6
16:13:23:    src mask = 255.255.255.255, dest mask = 0.0.0.0
16:13:23:    no frag, not advertising
16:13:23:    flags = 0x0, appl addr = 0.0.0.0, interest = 0x8107/0x8104
16:13:23:    int ip:port = 10.10.2.2:1638, sequence delta = 0/0/0/0
16:13:23:Updating wildcard affinity:
16:13:23:    0.0.0.0:0 -> 10.10.10.10:0 proto = 6
16:13:23:    src mask = 0.0.0.0, dest mask = 255.255.255.255
16:13:23:    no frag, advertising
16:13:23:    flags = 0x0, appl addr = 0.0.0.0, interest = 0x8107/0x8102
16:13:23    int ip:port = 10.10.2.2:1638, sequence delta = 0/0/0/0
```

The table below describes the significant fields shown in the display.

Table 7: debug ip casa wildcards Field Descriptions

Field	Description
src mask	Source of connection.
dest mask	Destination of connection.
no frag, not advertising	Not accepting IP fragments.

Field	Description
flags	Bit field indicating actions to be taken on this affinity.
fwd addr	Address to which packets matching the affinity will be directed.
interest	Services manager that is interested in packets for this affinity.
int ip: port	Services manager port to which interest packets are sent.
sequence delta	Used to adjust sequence numbers for this affinity.

debug ip cef

To troubleshoot various Cisco Express Forwarding events, use the **debug ip cef** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip cef {**drops** [**rpf** [*access-list*]] [*access-list*]| **receive** [*access-list*]| **events** [*access-list*]| **interface**| **dialer**}

no debug ip cef {**drops** [**rpf** [*access-list*]] [*access-list*]| **receive** [*access-list*]| **events** [*access-list*]| **interface**| **dialer**}

Specific to Interprocess Communication (IPC) Records

debug ip cef {**ipc**| **interface-ipc**| **prefix-ipc** [*access-list*]}

no debug ip cef {**ipc**| **interface-ipc**| **prefix-ipc** [*access-list*]}

Cisco 10000 Series Routers Only

debug ip cef {**drops** [**rpf** [*access-list*]] [*access-list*]| **receive** [*access-list*]| **events** [*access-list*]}

no debug ip cef {**drops** [**rpf** [*access-list*]] [*access-list*]| **receive** [*access-list*]| **events** [*access-list*]}

Cisco 10000 Series Routers Only--Specific to IPC Records

debug ip cef ipc

no debug ip cef ipc

Syntax Description

drops	Records dropped packets.
rpf	(Optional) Records the result of the Reverse Path Forwarding (RPF) check for packets.
access-list	(Optional) Limits debugging collection to packets that match the list.
receive	Records packets that are ultimately destined to the router and packets destined to a tunnel endpoint on the router. If the decapsulated tunnel is IP, the packets are Cisco Express Forwarding switched; otherwise the packets are process switched.
events	Records general Cisco Express Forwarding events.
interface	Records IP Cisco Express Forwarding interface events.
dialer	Records IP Cisco Express Forwarding interface events for dialer interfaces.

ipc	Records information related to IPC in Cisco Express Forwarding. Possible types of events are the following: • IPC messages received out of sequence • Status of resequenced messages • Status of buffer space for IPC messages • Transmission status of IPC messages • Throttle requests sent from a line card to the Route Processor
interface-ipc	Records IPC updates related to interfaces. Possible reporting includes an interface coming up or going down and updates to fibhwidb and fibidb.
prefix-ipc	Records updates related to IP prefix information. Possible updates include the following: • Debugging of IP routing updates in a line card • Reloading of a line card with a new table • Updates related to exceeding the maximum number of routes • Control messages related to Forwarding Information Base (FIB) table prefixes

Command Default

This command is disabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
11.2GS	This command was introduced.
11.1CC	Support for multiple platforms was added.
12.0(5)T	The **rpf** keyword was added.
12.2(4)T	The **dialer** keyword was added.
12.2(25)S	This command was integrated into Cisco IOS Release 12.2(25)S.

Release	Modification
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB and implemented on the Cisco 10000 series routers.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.
12.4(20)T	This command was integrated into Cisco IOS Release 12.4(20)T.

Usage Guidelines

This command gathers additional information for the handling of Cisco Express Forwarding interface, IPC, or packet events.

Note For packet events, we recommend that you use an access control list (ACL) to limit the messages recorded.

Examples

The following is sample output from the **debug ip cef rpf**commandfor a packet that is dropped when it fails the RPF check. IP address 172.17.249.252 is the source address, and Ethernet 2/0/0 is the input interface.

```
Router# debug ip cef drops rpf
IP CEF drops for RPF debugging is on
00:42:02:CEF-Drop:Packet from 172.17.249.252 via Ethernet2/0/0 -- unicast rpf check
```

The following is sample output for Cisco Express Forwarding packets that are not switched using information from the FIB table but are received and sent to the next switching layer:

```
Router# debug ip cef receive
IP CEF received packets debugging is on
00:47:52:CEF-receive:Receive packet for 10.1.104.13
```

The table below describes the significant fields shown in the display.

Table 8: debug ip cef receive Field Descriptions

Field	Description
CEF-Drop:Packet from 172.17.249.252 via Ethernet2/0/0 -- unicast rpf check	A packet from IP address 172.17.249.252 is dropped because it failed the RPF check.
CEF-receive:Receive packet for 10.1.104.13	Cisco Express Forwarding has received a packet addressed to the router.

The following is sample output from the **debug ip cef dialer**commandfor a legacy dialer:

```
Router# debug ip cef dialer
00:19:50:CEF-Dialer (legacy):add link to 10.10.10.2 via Dialer1 through BRI0/0:1
00:19:50:CEF-Dialer:adjacency added:0x81164850
00:19:50:CEF-Dialer:adjacency found:0x81164850; fib->count:1
00:19:50:CEF-Dialer:setup loadinfo with 1 paths
```

The following is sample output from the **debug ip cef dialer** command for a dialer profile:

```
Router# debug ip cef dialer
00:31:44:CEF-Dialer (profile dynamic encap (not MLP)):add link to 10.10.10.2 via Dialer1
through Dialer1
00:31:44:CEF-Dialer:adjacency added:0x81164850
00:31:44:CEF-Dialer:adjacency found:0x81164850; fib->count:1
```

The table below describes the significant fields shown in the display.

Table 9: debug ip cef dialer Field Descriptions

Field	Description
CEF-Dialer (legacy):add link to 10.10.10.2 via Dialer1 through BRI0/0:1	A link was added to IP address 10.10.10.2 for legacy Dialer1 through physical interface BRI0/0:1.
CEF-Dialer (profile dynamic encap (not MLP)):add link to 10.10.10.2 via Dialer1 through Dialer1	A link was added to IP address 10.10.10.2 for dialer profile Dialer1 through Dialer1.

Related Commands

Command	Description
ip cef	Enables Cisco Express Forwarding on the RPC card.
show ip cef	Displays entries in the FIB or displays a summary of the FIB.

debug ip cef accounting non-recursive

To troubleshoot Cisco Express Forwarding accounting records, use the **debug ip cef accounting non-recursive**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip cef accounting non-recursive

no debug ip cef accounting non-recursive

Syntax Description

This command has no arguments or keywords.

Command Default

This command is disabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
11.1CC	This command was introduced.
12.2(25)S	This command was integrated into Cisco IOS Release 12.2(25)S.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB and implemented on the Cisco 10000 series routers.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.
12.4(20)T	This command was integrated into Cisco IOS Release 12.4(20)T.

Usage Guidelines

This command records accounting events for nonrecursive prefixes when the **ip cef accounting non-recursive** command is enabled in global configuration mode.

Examples

The following is sample output from the **debug ip cef accounting non-recursive** command:

```
Router# debug ip cef accounting non-recursive
03:50:19:CEF-Acct:tmstats_binary:Beginning generation of tmstats
ephemeral file (mode binary)
03:50:19:CEF-Acct:snapshoting loadinfo 0x63FF2000
03:50:19:CEF-Acct:snapshoting loadinfo 0x63FF1EA0
03:50:19:CEF-Acct:snapshoting loadinfo 0x63FF17C0
03:50:19:CEF-Acct:snapshoting loadinfo 0x63FF1D40
03:50:19:CEF-Acct:snapshoting loadinfo 0x63FF1A80
03:50:19:CEF-Acct:snapshoting loadinfo 0x63FF0740
03:50:19:CEF-Acct:snapshoting loadinfo 0x63FF08A0
```

```
03:50:19:CEF-Acct:snapshoting loadinfo 0x63FF0B60
03:50:19:CEF-Acct:snapshoting loadinfo 0x63FF0CC0
03:50:19:CEF-Acct:snapshoting loadinfo 0x63FF0F80
03:50:19:CEF-Acct:snapshoting loadinfo 0x63FF10E0
03:50:19:CEF-Acct:snapshoting loadinfo 0x63FF1240
03:50:19:CEF-Acct:snapshoting loadinfo 0x63FF13A0
03:50:19:CEF-Acct:snapshoting loadinfo 0x63FF1500
03:50:19:CEF-Acct:snapshoting loadinfo 0x63FF1920
03:50:19:CEF-Acct:snapshoting loadinfo 0x63FF0E20
03:50:19:CEF-Acct:snapshoting loadinfo 0x63FF1660
03:50:19:CEF-Acct:snapshoting loadinfo 0x63FF05E0
03:50:19:CEF-Acct:snapshoting loadinfo 0x63FF0A00
03:50:19:CEF-Acct:snapshoting loadinfo 0x63FF1BE0
03:50:19:CEF-Acct:snapshoting loadinfo 0x63FF0480
03:50:19:CEF-Acct:tmstats_binary:aggregation complete, duration 0 seconds
03:50:21:CEF-Acct:tmstats_binary:writing 45 bytes
03:50:24:CEF-Acct:tmstats_binary:writing 45 bytes
03:50:24:CEF-Acct:tmstats_binary:writing 45 bytes
03:50:27:CEF-Acct:tmstats_binary:writing 45 bytes
03:50:29:CEF-Acct:tmstats_binary:writing 45 bytes
03:50:32:CEF-Acct:tmstats_binary:writing 45 bytes
03:50:35:CEF-Acct:tmstats_binary:writing 45 bytes
03:50:38:CEF-Acct:tmstats_binary:writing 45 bytes
03:50:41:CEF-Acct:tmstats_binary:writing 45 bytes
03:50:45:CEF-Acct:tmstats_binary:writing 45 bytes
03:50:48:CEF-Acct:tmstats_binary:writing 45 bytes
03:50:49:CEF-Acct:tmstats_binary:writing 45 bytes
03:50:52:CEF-Acct:tmstats_binary:writing 45 bytes
03:50:55:CEF-Acct:tmstats_binary:writing 45 bytes
03:50:57:CEF-Acct:tmstats_binary:writing 45 bytes
03:50:57:CEF-Acct:tmstats_binary:writing 45 bytes
03:50:57:CEF-Acct:tmstats_binary:writing 45 bytes
03:50:57:CEF-Acct:tmstats_binary:writing 45 bytes
03:50:57:CEF-Acct:tmstats_binary:writing 45 bytes
03:50:57:CEF-Acct:tmstats_binary:writing 45 bytes
03:50:57:CEF-Acct:tmstats_binary:writing 45 bytes
03:50:57:CEF-Acct:tmstats_binary:tmstats file written, status 0
```

The table below describes the significant fields shown in the display.

Table 10: debug ip cef accounting non-recursive Field Descriptions

Field	Description
Beginning generation of tmstats ephemeral file (mode binary)	Tmstats file is being created.
CEF-Acct:snapshoting loadinfo 0x63FF2000	Baseline counters are being written to the tmstats file for each nonrecursive prefix.
CEF-Acct:tmstats_binary:aggregation complete, duration 0 seconds	Tmstats file creation is complete.
CEF-Acct:tmstats_binary:writing 45 bytes	Nonrecursive accounting statistics are being updated to the tmstats file.
CEF-Acct:tmstats_binary:tmstats file written, status 0	Update of the tmstats file is complete.

Related Commands

Command	Description
debug ip cef	Troubleshoots various Cisco Express Forwarding events.
ip cef accounting	Enables Cisco Express Forwarding network accounting.
show ip cef	Displays entries or a summary of the FIB table.

debug ip cef fragmentation

To report fragmented IP packets when Cisco Express Forwarding is enabled, use the **debug ip cef fragmentation**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command:

debug ip cef fragmentation

no debug ip cef fragmentation

Syntax Description

This command has no arguments or keywords.

Command Default

This command is disabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.0(14)S	This command was introduced.
12.2(2)T	This command was integrated into Cisco IOS Release 12.2(2)T.
12.2(25)S	This command was integrated into Cisco IOS Release 12.2(25)S.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB and implemented on the Cisco 10000 series routers.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.
12.4(20)T	This command was integrated into Cisco IOS Release 12.4(20)T.

Usage Guidelines

This command is used to troubleshoot fragmentation problems when Cisco Express Forwarding switching is enabled.

Examples

The following is sample output from the **debug ip cef fragmentation**command:

```
Router# debug ip cef fragmentation
00:59:45:CEF-FRAG:no_fixup path:network_start 0x5397CF8E datagramstart 0x5397CF80 data_start
 0x397CF80 data_block 0x397CF40 mtu 1000 datagramsize 1414 data_bytes 1414
00:59:45:CEF-FRAG:send frag:datagramstart 0x397CF80 datagramsize 442 data_bytes 442
00:59:45:CEF-FRAG:send frag:datagramstart 0x38BC266 datagramsize 1006 data_bytes 1006
00:59:45:CEF-FRAG:no_fixup path:network_start 0x5397C60E datagramstart 0x5397C600 data_start
 0x397C600 data_block 0x397C5C0 mtu 1000 datagramsize 1414 data_bytes 1414
```

```
00:59:45:CEF-FRAG:send frag:datagramstart 0x397C600 datagramsize 442 data_bytes 442
00:59:45:CEF-FRAG:send frag:datagramstart 0x38BC266 datagramsize 1006 data_bytes 1006
```

The table below describes the significant fields shown in the display.

Table 11: debug ip cef fragmentation Field Descriptions

Field	Description
no_fixup path	A packet is being fragmented in the no_fixup path.
network_start 0x5397CF8E	Memory address of the IP packet.
datagramstart 0x5397CF80	Memory address of the encapsulated IP packet.
data_start 0x397CF80	For particle systems, the memory address where data starts for the first packet particle.
data_block 0x397C5C0	For particle systems, the memory address of the first packet particle data block.
mtu 1000	Maximum transmission unit of the output interface.
datagramsize 1414	Size of the encapsulated IP packet.
data_bytes 1414	For particle systems, the sum of the particle data bytes that make up the packet.
send frag	Fragment is being forwarded.

Related Commands

Command	Description
debug ip cef	Troubleshoots various Cisco Express Forwarding events.

debug ip cef hash

To record Cisco Express Forwarding load sharing hash algorithm events, use the **debug ip cef hash**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip cef hash

no debug ip cef hash

Syntax Description

This command has no arguments or keywords.

Command Default

This command is disabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.0(12)S	This command was introduced.
12.1(5)T	This command was integrated into Cisco IOS Release 12.1(5)T.
12.2(25)S	This command was integrated into Cisco IOS Release 12.2(25)S.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB. This command is not supported on the Cisco 10000 series routers.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.This command is not supported on the Cisco 7600 router.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.
12.4(20)T	This command was integrated into Cisco IOS Release 12.4(20)T.

Usage Guidelines

Use this command when changing the load sharing algorithm to display the hash table details.

Examples

The following is sample output from the **debug ip cef hash**command with IP Cisco Express Forwarding load algorithm tunnel information:

```
Router# debug ip cef hash
01:15:06:%CEF:ip cef load-sharing algorithm tunnel 0
01:15:06:%CEF:Load balancing algorithm:tunnel
01:15:06:%CEF:Load balancing unique id:1F2BA5F6
01:15:06:%CEF:Destroyed load sharing hash table
01:15:06:%CEF:Sending hash algorithm id 2, unique id 1F2BA5F6 to slot 255
```

The following lines show IP Cisco Express Forwarding load algorithm universal information:

```
01:15:28:%CEF:ip cef load-sharing algorithm universal 0
01:15:28:%CEF:Load balancing algorithm:universal
01:15:28:%CEF:Load balancing unique id:062063A4
01:15:28:%CEF:Creating load sharing hash table
01:15:28:%CEF:Hash table columns for valid max_index:
01:15:28:12: 9  7  7 4  4 10  0  7 10  4  5  0  4  7  8  4
01:15:28:15: 3 10 10  4 10  4  0  7  1  7 14  6 13 13 11 13
01:15:28:16: 1  3  7 12  4 14  8  7 10  4  1 12  8 15  4  8
01:15:28:%CEF:Sending hash algorithm id 3, unique id 062063A4 to slot 255
```

The table below describes the significant fields shown in the display.

Table 12: debug ip cef hash Field Descriptions

Field	Description
ip cef load-sharing algorithm tunnel 0	Echo of the user command.
Load balancing algorithm:tunnel	Load sharing algorithm is set to tunnel.
Load balancing unique id:1F2BA5F6	ID field in the command is usually 0. In this instance, the router chose a pseudo random ID of 1F2BA5F6.
Destroyed load sharing hash table	Purge the existing hash table.
Sending hash algorithm id 2, unique id 1F2BA5F6 to slot 255	Algorithm is being distributed.
Creating load sharing hash table	Hash table is being created.
Hash table columns for valid max_index:	Generated hash table.

Related Commands

Command	Description
debug ip cef	Troubleshoots various Cisco Express Forwarding events.
debug ip cef rrhash	Records Cisco Express Forwarding removal of receive hash events.

debug ip cef rrhash

To record Cisco Express Forwarding removal of receive hash events, use the **debug ip cef rrhash**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip cef rrhash

no debug ip cef rrhash

Syntax Description

This command has no arguments or keywords.

Command Default

This command is disabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(2)T	This command was introduced.
12.2(25)S	This command was integrated into Cisco IOS Release 12.2(25)S.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB. This command is not supported on the Cisco 10000 series routers.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA. This command is not supported on the Cisco 7600 routers.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.
12.4(20)T	This command was integrated into Cisco IOS Release 12.4(20)T.

Usage Guidelines

Use this command to verify the removal of receive hash events when you are shutting down or deleting an interface.

Examples

The following is sample output from the **debug ip cef rrhash**command:

```
Router# debug ip cef rrhash
00:27:15:CEF:rrhash/check:found 10.1.104.7 on down idb [ok to delete]
00:27:15:CEF:rrhash/check:found 10.1.104.0 on down idb [ok to delete]
00:27:15:CEF:rrhash/check:found 10.1.104.255 on down idb [ok to delete]
00:27:15:CEF:rrhash/check:found 10.1.104.7 on down idb [ok to delete]
00:27:15:CEF:rrhash/check:found 10.1.104.7 on down idb [ok to delete]
00:27:15:CEF:rrhash/check:found 10.1.104.0 on down idb [ok to delete]
00:27:15:CEF:rrhash/check:found 10.1.104.255 on down idb [ok to delete]
00:27:15:CEF:rrhash/check:found 10.1.104.7 on down idb [ok to delete]
```

The table below describes the significant fields shown in the display.

Table 13: debug ip cef rrhash Field Descriptions

Field	Description
rrhash/check	Verify address is on the receive list.
found 10.1.104.7 on down idb [ok to delete]	Found a valid address on the receive list for a shutdown interface that can be deleted.

Related Commands

Command	Description
debug ip cef	Troubleshoots various Cisco Express Forwarding events.
debug ip cef hash	Records Cisco Express Forwarding removal of receive hash events.

debug ip cef subblock

To troubleshoot Cisco Express Forwarding subblock events, use the **debug ip cef subblock**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip cef subblock [**id** {**all**| **hw** *hw-id*| **sw** *sw-id*}] [**xdr** {**all**| **control**| **event**| **none**| **statistic**}]

no debug ip cef subblock [**id** {**all**| **hw** *hw-id*| **sw** *sw-id*}] [**xdr** {**all**| **control**| **event**| **none**| **statistic**}]

Syntax Description

id	(Optional) Subblock types.
all	(Optional) All subblock types.
hw *hw-id*	(Optional) Hardware subblock and identifier.
sw *sw-id*	(Optional) Software subblock and identifier.
xdr	(Optional) Exernal Data Representation (XDR) message types.
control	(Optional) All XDR message types.
event	(Optional) Event XDR messages only.
none	(Optional) No XDR messages.
statistic	(Optional) Statistic XDR messages.

Command Default

This command is disabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.0S	This command was introduced.
12.2(2)T	This command was integrated into Cisco IOS Release 12.2(2)T.
12.2(25)S	This command was integrated into Cisco IOS Release 12.2(25)S.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB and implemented on the Cisco 10000 series routers.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Release	Modification
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.
12.4(20)T	This command was integrated into Cisco IOS Release 12.4(20)T.

Usage Guidelines

This command is used to record Cisco Express Forwarding subblock messages and events.

Examples

The following is sample output from the **debug ip cef subblock**command:

```
Router# debug ip cef subblock
00:28:12:CEF-SB:Creating unicast RPF subblock for FastEthernet6/0
00:28:12:CEF-SB:Linked unicast RPF subblock to FastEthernet6/0.
00:28:12:CEF-SB:Encoded unit of unicast RPF data (length 16) for FastEthernet6/0
00:28:12:CEF-SB:Sent 1 data unit to slot 6 in 1 XDR message
```

Examples

The following is sample output from the **debug ip cef subblock**command:

```
Router# debug ip cef subblock
00:28:12:CEF-SB:Creating unicast RPF subblock for FastEthernet6/0/0
00:28:12:CEF-SB:Linked unicast RPF subblock to FastEthernet6/0/0.
00:28:12:CEF-SB:Encoded unit of unicast RPF data (length 16) for FastEthernet6/0/0
00:28:12:CEF-SB:Sent 1 data unit to slot 6 in 1 XDR message
```

The table below describes the significant fields shown in the display.

Table 14: debug ip cef subblock Field Descriptions

Field	Description
Creating unicast RPF subblock for FastEthernet6/0/0	Creating an Unicast Reverse Path Forwarding (Unicast RPF) interface descriptor subblock.
Linked unicast RPF subblock to FastEthernet6/0/0	Linked the subblock to the specified interface.
Encoded unit of unicast RPF data (length 16) for FastEthernet6/0/0	Encoded the subblock information in an XDR.
Sent 1 data unit to slot 6 in 1 XDR message	Sent the XDR message to a line card through the IPC.

Related Commands

Command	Description
debug ip cef	Troubleshoots various Cisco Express Forwarding events.

debug ip cef table

To enable the collection of events that affect entries in the Cisco Express Forwarding tables, use the **debug ip cef table**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip cef table [*access-list*| **consistency-checkers**]

no debug ip cef table [*access-list*| **consistency-checkers**]

Syntax Description

access-list	(Optional) Controls collection of consistency checker parameters from specified lists.
consistency-checkers	(Optional) Sets consistency checking characteristics.

Command Default

This command is disabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
11.2GS	This command was introduced.
11.1CC	Support was added for multiple platforms.
12.0(15)S	The **consistency-checkers** keyword was added.
12.2(2)T	This command was integrated into Cisco IOS Release 12.2(2)T.
12.2(25)S	This command was integrated into Cisco IOS Release 12.2(25)S.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB and implemented on the Cisco 10000 series routers.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.
12.4(20)T	This command was integrated into Cisco IOS Release 12.4(20)T.

Usage Guidelines

This command is used to record Cisco Express Forwarding table events related to the Forwarding Information Base (FIB) table. Possible types of events include the following:

- Routing updates that populate the FIB table
- Flushing of the FIB table
- Adding or removing of entries to the FIB table
- Table reloading process

Examples

The following is sample output from the **debug ip cef table** command:

```
Router# debug ip cef table
01:25:46:CEF-Table:Event up, 10.1.1.1/32 (rdbs:1, flags:1000000)
01:25:46:CEF-IP:Checking dependencies of 0.0.0.0/0
01:25:47:CEF-Table:attempting to resolve 10.1.1.1/32
01:25:47:CEF-IP:resolved 10.1.1.1/32 via 10.1.104.1 to 10.1.104.1 Ethernet2/0/0
01:26:02:CEF-Table:Event up, default, 0.0.0.0/0 (rdbs:1, flags:400001)
01:26:02:CEF-IP:Prefix exists - no-op change
```

Examples

The following is sample output from the **debug ip cef table** command:

```
Router# debug ip cef table
01:25:46:CEF-Table:Event up, 10.1.1.1/32 (rdbs:1, flags:1000000)
01:25:46:CEF-IP:Checking dependencies of 0.0.0.0/0
01:25:47:CEF-Table:attempting to resolve 10.1.1.1/32
01:25:47:CEF-IP:resolved 10.1.1.1/32 via 10.1.104.1 to 10.1.104.1  GigabitEthernet2/0/0
01:26:02:CEF-Table:Event up, default, 0.0.0.0/0 (rdbs:1, flags:400001)
01:26:02:CEF-IP:Prefix exists - no-op change
```

The table below describes the significant fields shown in the display.

Table 15: debug ip cef table Field Descriptions

Field	Description
CEF-Table	Indicates a table event.
Event up, 10.1.1.1/32	IP prefix 10.1.1.1/32 is being added.
rdbs:1	Event is from routing descriptor block 1.
flags:1000000	Indicates the network descriptor block flags.
CEF-IP	Indicates a Cisco Express Forwarding IP event.
Checking dependencies of 0.0.0.0/0	Resolves the next hop dependencies for 0.0.0.0/0.
attempting to resolve 10.1.1.1/32	Resolves the next hop dependencies.
resolved 10.1.1.1/32 via 10.1.104.1 to 10.1.104.1 Ethernet2/0/0	Next hop to IP prefix 10.1.1.1/32 is set and is added to the table.
Event up, default, 0.0.0.0/0 Prefix exists - no-op change	Indicates no table change is necessary for 0.0.0.0/32.

Related Commands

Command	Description
cef table consistency-check	Enables Cisco Express Forwarding consistency checker table values by type and parameter.
clear cef table	Clears the Cisco Express Forwarding tables.
clear ip cef inconsistency	Clears Cisco Express Forwarding inconsistency statistics and records found by the Cisco Express Forwarding consistency checkers.
debug cef	Enables the display of information about Cisco Express Forwarding events.
debug ip cef	Troubleshoots various Cisco Express Forwarding events.
show cef table consistency-check	Displays Cisco Express Forwarding consistency checker table values.
show ip cef inconsistency	Displays Cisco Express Forwarding IP prefix inconsistencies.

debug ip ddns update

To enable debugging for Dynamic Domain Name System (DDNS) updates, use the **debug ip ddns update**command in privileged EXEC mode. To disable the debugging, use the **no** form of this command.

debug ip ddns update

no debug ip ddns update

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(8)YA	This command was introduced.
12.3(14)T	This command was integrated into Cisco IOS Release 12.3(14)T.

Examples

Use the **debug ip ddns update** command to verify that your configurations are working properly. The following sample configurations are shown for demonstration of possible debug output that could display for each configuration.

Examples

The following scenario has a client configured for IETF DDNS updating of address (A) Resource Records (RRs) during which a Dynamic Host Configuration Protocol (DHCP) server is expected to update the pointer (PTR) RR. The DHCP client discovers the domain name system (DNS) server to update using an Start of Authority (SOA) RR lookup since the IP address to the server to update is not specified. The DHCP client is configured to include an fully qualified domain name (FQDN) DHCP option and notifies the DHCP server that it will be updating the A RRs.

```
!DHCP Client Configuration
ip ddns update method testing
 ddns
interface Ethernet1
 ip dhcp client update dns
 ip ddns update testing
 ip address dhcp
end
!DHCP Server Configuration
ip dhcp pool test
 network 10.0.0.0 255.0.0.0
 update dns
!Debug Output Enabled
Router# debug ip ddns update
00:14:39: %DHCP-6-ADDRESS_ASSIGN: Interface Ethernet1 assigned DHCP address 10.0.0.4, mask
 255.0.0.0, hostname canada_reserved
00:14:39: DYNDNSUPD: Adding DNS mapping for canada_reserved.hacks <=> 10.0.0.4
00:14:39: DYNDNSUPD: Sleeping for 3 seconds waiting for interface Ethernet1 configuration
to settle
00:14:42: DHCPC: Server performed PTR update
```

```
00:14:42: DDNS: Enqueuing new DDNS update 'canada_reserved.hacks' <=> 10.0.0.4
00:14:42: DDNS: Zone name for 'canada_reserved.hacks' is 'hacks'
00:14:42: DDNS: Dynamic Update 1: (sending to server 10.19.192.32)
00:14:42: DDNS:   Zone = hacks
00:14:42: DDNS:   Prerequisite: canada_reserved.hacks not in use
00:14:42: DDNS:   Update: add canada_reserved.hacks IN A 10.0.0.4
00:14:42: DDNS: Dynamic DNS Update 1 (A) for host canada_reserved.hacks returned 0 (NOERROR)
00:14:42: DDNS: Update of 'canada_reserved.hacks' <=> 10.0.0.4 finished
00:14:42: DYNDNSUPD: Another update completed (total outstanding=0)
```

Examples

The following scenario has the client configured for IETF DDNS updating of both A and DNS RRs and requesting that the DHCP server update neither. The DHCP client discovers the DNS server to update using an SOA RR lookup since the IP address to the server to update is not specified. The DHCP client is configured to include an FQDN DHCP option that instructs the DHCP server to not update either A or PTR RRs.

```
!DHCP Client Configuration
ip dhcp-client update dns server none
ip ddns update method testing
 ddns both
interface Ethernet1
 ip ddns update testing
 ip address dhcp
end
!DHCP Server Configuration
ip dhcp pool test
 network 10.0.0.0 255.0.0.0
 update dns
!Debug Output Enabled
Router# debug ip ddns update
00:15:33: %DHCP-6-ADDRESS_ASSIGN: Interface Ethernet1 assigned DHCP address 10.0.0.5, mask
 255.0.0.0, hostname canada_reserved
00:15:33: DYNDNSUPD: Adding DNS mapping for canada_reserved.hacks <=> 10.0.0.5
00:15:33: DYNDNSUPD: Sleeping for 3 seconds waiting for interface Ethernet1 configuration
to settle
00:15:36: DDNS: Enqueuing new DDNS update 'canada_reserved.hacks' <=> 10.0.0.5
00:15:36: DDNS: Zone name for '10.0.0.11.in-addr.arpa.' is '10.in-addr.arpa'
00:15:36: DDNS: Dynamic Update 1: (sending to server 10.19.192.32)
00:15:36: DDNS:   Zone = 10.in-addr.arpa
00:15:36: DDNS:   Prerequisite: 10.0.0.11.in-addr.arpa. not in use
00:15:36: DDNS:   Update: add 10.0.0.11.in-addr.arpa. IN PTR canada_reserved.hacks
00:15:36: DDNS: Dynamic DNS Update 1 (PTR) for host canada_reserved.hacks returned 0 (NOERROR)
00:15:36: DDNS: Zone name for 'canada_reserved.hacks' is 'hacks'
00:15:36: DDNS: Dynamic Update 1: (sending to server 10.19.192.32)
00:15:36: DDNS:   Zone = hacks
00:15:36: DDNS:   Prerequisite: canada_reserved.hacks not in use
00:15:36: DDNS:   Update: add canada_reserved.hacks IN A 10.0.0.5
00:15:36: DDNS: Dynamic DNS Update 1 (A) for host canada_reserved.hacks returned 0 (NOERROR)
00:15:36: DDNS: Update of 'canada_reserved.hacks' <=> 10.0.0.5 finished
00:15:36: DYNDNSUPD: Another update completed (total outstanding=0)
```

Examples

The following scenario has the client configured for IETF DDNS updating of both A and DNS RRs and requesting that the DHCP server update neither. The DHCP client explicitly specifies the server to update. The DHCP client is configured to include an FQDN DHCP option that instructs the DHCP server not to update either A or PTR RRs. The configuration is performed using the **ip dhcp client update dns** command. The DHCP server is configured to override the client request and update both A and PTR RR anyway.

```
!DHCP Client Configuration
ip dhcp client update dns server none
ip ddns update method testing
 ddns both
interface Ethernet1
 ip dhcp client update dns server none
 ip ddns update testing
 ip address dhcp
```

```
end
!DHCP Server Configuration
ip dhcp pool test
 network 10.0.0.0 255.0.0.0
 update dns both override
!Debug Output Enabled on DHCP Client
Router# debug ip ddns update
00:16:30: %DHCP-6-ADDRESS_ASSIGN: Interface Ethernet1 assigned DHCP address 10.0.0.6, mask
 255.0.0.0, hostname canada_reserved
00:16:30: DYNDNSUPD: Adding DNS mapping for canada_reserved.hacks <=> 10.0.0.6
00:16:30: DYNDNSUPD: Sleeping for 3 seconds waiting for interface Ethernet1 configuration
to settle
00:16:33: DHCPC: Server performed both updates
```

Examples

The following scenario has the client configured for IETF DDNS updating of both A and DNS RRs and requesting the DHCP server to update neither. The DHCP client is configured to include an FQDN DHCP option which instructs the DHCP server not to update either A or PTR RRs. The DHCP server is configured to allow the client to update whatever RR it chooses.

```
!DHCP Client Configuration
ip dhcp client update dns server non
ip ddns update method testing
 ddns both
interface Ethernet1
 ip dhcp client update dns server none
 ip ddns update testing host 172.19.192.32
 ip address dhcp
end
!DHCP Server Configuration
ip dhcp pool test
 network 10.0.0.0 255.0.0.0
 update dns
!Debug Output Enabled on DHCP Client
Router# debug ip ddns update
00:17:52: %DHCP-6-ADDRESS_ASSIGN: Interface Ethernet1 assigned DHCP address 10.0.0.7, mask
 255.0.0.0, hostname canada_reserved
00:17:52: DYNDNSUPD: Adding DNS mapping for canada_reserved.hacks <=> 10.0.0.6
00:17:52: DYNDNSUPD: Sleeping for 3 seconds waiting for interface Ethernet1 configuration
to settle
00:17:55: DDNS: Enqueuing new DDNS update 'canada_reserved.hacks' <=> 10.0.0.7
00:17:55: DYNDNSUPD: Adding DNS mapping for canada_reserved.hacks <=> 10.0.0.7 server
10.19.192.32
00:17:55: DDNS: Enqueuing new DDNS update 'canada_reserved.hacks' <=> 10.0.0.7 server
10.19.192.32
00:17:55: DDNS: Zone name for '7.0.0.11.in-addr.arpa.' is '11.in-addr.arpa'
00:17:55: DDNS: Dynamic Update 1: (sending to server 10.19.192.32)
00:17:55: DDNS:   Zone = 11.in-addr.arpa
00:17:55: DDNS:   Prerequisite: 10.0.0.11.in-addr.arpa. not in use
00:17:55: DDNS:   Update: add 10.0.0.11.in-addr.arpa. IN PTR canada_reserved.hacks
00:17:55: DDNS: Zone name for '10.0.0.11.in-addr.arpa.' is '10.in-addr.arpa'
00:17:55: DDNS: Using server 10.19.192.32
00:17:55: DDNS: Dynamic Update 1: (sending to server 10.19.192.32)
00:17:55: DDNS:   Zone = 11.in-addr.arpa
00:17:55: DDNS:   Prerequisite: 10.0.0.11.in-addr.arpa. not in use
00:17:55: DDNS:   Update: add 10.0.0.11.in-addr.arpa. IN PTR canada_reserved.hacks
00:17:55: DDNS: Dynamic DNS Update 1 (PTR) for host canada_reserved.hacks returned 0 (NOERROR)
00:17:55: DDNS: Dynamic DNS Update 1 (PTR) for host canada_reserved.hacks returned 6
(YXDOMAIN)
00:17:55: DDNS: Dynamic Update 2: (sending to server 10.19.192.32)
00:17:55: DDNS:   Zone = 11.in-addr.arpa
00:17:55: DDNS:   Update: delete 10.0.0.11.in-addr.arpa. all PTR RRs
00:17:55: DDNS:   Update: add 10.0.0.11.in-addr.arpa. IN PTR canada_reserved.hacks
00:17:55: DDNS: Dynamic DNS Update 2 (PTR) for host canada_reserved.hacks returned 0 (NOERROR)
00:17:55: DDNS: Zone name for 'canada_reserved.hacks' is 'hacks'
00:17:55: DDNS: Dynamic Update 1: (sending to server 10.19.192.32)
00:17:55: DDNS:   Zone = hacks
00:17:55: DDNS:   Prerequisite: canada_reserved.hacks not in use
00:17:55: DDNS:   Update: add canada_reserved.hacks IN A 10.0.0.7
00:17:55: DDNS: Dynamic DNS Update 1 (A) for host canada_reserved.hacks returned 0 (NOERROR)
```

```
00:17:55: DDNS: Update of 'canada_reserved.hacks' <=> 10.0.0.7 finished
00:17:55: DYNDNSUPD: Another update completed (total outstanding=1)
00:17:55: DDNS: Zone name for 'canada_reserved.hacks' is 'hacks'
00:17:55: DDNS: Using server 10.19.192.32
00:17:55: DDNS: Dynamic Update 1: (sending to server 10.19.192.32)
00:17:55: DDNS:   Zone = hacks
00:17:55: DDNS:   Prerequisite: canada_reserved.hacks not in use
00:17:55: DDNS:   Update: add canada_reserved.hacks IN A 10.0.0.7
00:17:55: DDNS: Dynamic DNS Update 1 (A) for host canada_reserved.hacks returned 6 (YXDOMAIN)
00:17:55: DDNS: Dynamic Update 2: (sending to server 10.19.192.32)
00:17:55: DDNS:   Zone = hacks
00:17:55: DDNS:   Update: delete canada_reserved.hacks all A RRs
00:17:55: DDNS:   Update: add canada_reserved.hacks IN A 10.0.0.7
00:17:55: DDNS: Dynamic DNS Update 2 (A) for host canada_reserved.hacks returned 0 (NOERROR)
00:17:55: DDNS: Update of 'canada_reserved.hacks' <=> 10.0.0.7 finished
00:17:55: DYNDNSUPD: Another update completed (total outstanding=0)
```

Examples

In the following scenario, the debug output displays the internal host table updates when the default domain name is hacks. The update method named test specifies that the internal Cisco IOS software host table should be updated. Configuring the update method as "test" should be used when the address on the Ethernet interface 0/0 changes. The hostname is configured for the update on this interface.

```
!Cisco IOS Software Configuration
ip domain name hacks
ip ddns update method test
 internal
interface ethernet0/0
 ip ddns update test hostname test2
 ip addr dhcp
!Debug Output Enabled
Router# debug ip ddns update
*Jun 4 03:11:10.591: %DHCP-6-ADDRESS_ASSIGN: Interface Ethernet0/0 assigned DHCP address
10.0.0.5, mask 255.0.0.0, hostname test2
*Jun 4 03:11:10.591: DYNDNSUPD: Adding DNS mapping for test2.hacks <=> 10.0.0.5
*Jun 4 03:11:10.591: DYNDNSUPD: Adding internal mapping test2.hacks <=> 10.0.0.5
```

Using the **show hosts** command displays the newly added host table entry.

```
Router# show hosts
Default domain is hacks
Name/address lookup uses domain service
Name servers are 255.255.255.255
Codes: UN - unknown, EX - expired, OK - OK, ?? - revalidate
       temp - temporary, perm - permanent
       NA - Not Applicable None - Not defined
Host                      Port Flags      Age Type   Address(es)
test2.hacks               None (perm, OK) 0   IP    10.0.0.5
```

Shutting down the interface removes the host table entry.

```
interface ethernet0/0
 shutdown
*Jun 4 03:14:02.107: DYNDNSUPD: Removing DNS mapping for test2.hacks <=> 10.0.0.5
*Jun 4 03:14:02.107: DYNDNSUPD: Removing mapping test2.hacks <=> 10.0.0.5
```

Using the **show hosts** command confirms that the entry has been removed.

```
Router# show hosts
Default domain is hacks
Name/address lookup uses domain service
Name servers are 255.255.255.255
Codes: UN - unknown, EX - expired, OK - OK, ?? - revalidate
       temp - temporary, perm - permanent
       NA - Not Applicable None - Not defined
Host                      Port Flags      Age Type   Address(es)
```

Examples

In the following scenario, the debug output shows the HTTP-style DDNS updates. The sample configuration defines a new IP DDNS update method named dyndns that configures a URL to use when adding or changing an address. No URL has been defined for use when removing an address since DynDNS.org does not use such a URL for free accounts. A maximum update interval of 28 days has been configured, which specifies that updates should be sent at least every 28 days. Configuring the new “dyndns” update method should be used for Ethernet interface 1.

```
!DHCP Client Configuration
ip ddns update method dyndns
 http
    add http://test:test@<s>/nic/update?system=dyndns&hostname=<h>&myip=<a>
    interval max 28 0 0 0
interface ethernet1
 ip ddns update hostname test.dyndns.org
 ip ddns update dyndns host members.dyndns.org
 ip addr dhcp
!Debugging Enabled
Router# debug ip ddns update
00:04:35: %DHCP-6-ADDRESS_ASSIGN: Interface Ethernet1 assigned DHCP address 10.32.254.187,
 mask 255.255.255.240, hostname test.dyndns.org
00:04:35: DYNDNSUPD: Adding DNS mapping for test.dyndns.org <=> 10.32.254.187 server
63.208.196.94
00:04:35: DYNDNSUPD: Sleeping for 3 seconds waiting for interface Ethernet1 configuration
to settle
00:04:38: HTTPDNS: Update add called for test.dyndns.org <=> 10.32.254.187
00:04:38: HTTPDNS: Update called for test.dyndns.org <=> 10.32.254.187
00:04:38: HTTPDNS: init
00:04:38: HTTPDNSUPD: Session ID = 0x7
00:04:38: HTTPDNSUPD: URL =
'http://test:test@63.208.196.94/nic/update?system=dyndns&hostname=test.dyndns.org&myip=10.32.254.187'
00:04:38: HTTPDNSUPD: Sending request
00:04:40: HTTPDNSUPD: Response for update test.dyndns.org <=> 10.32.254.187
00:04:40: HTTPDNSUPD: DATA START
good 10.32.254.187
00:04:40: HTTPDNSUPD: DATA END, Status is Response data received, successfully
00:04:40: HTTPDNSUPD: Call returned SUCCESS for update test.dyndns.org <=> 10.32.254.187
00:04:40: HTTPDNSUPD: Freeing response
00:04:40: DYNDNSUPD: Another update completed (outstanding=0, total=0)
00:04:40: HTTPDNSUPD: Clearing all session 7 info
!28 days later, the automatic update happens.
00:05:39: DYNDNSUPD: Adding DNS mapping for test.dyndns.org <=> 10.32.254.187 server
63.208.196.94
00:05:39: HTTPDNS: Update add called for test.dyndns.org <=> 10.32.254.187
00:05:39: HTTPDNS: Update called for test.dyndns.org <=> 10.32.254.187
00:05:39: HTTPDNS: init
00:05:39: HTTPDNSUPD: Session ID = 0x8
00:05:39: HTTPDNSUPD: URL =
'http://test:test@63.208.196.94/nic/update?system=dyndns&hostname=test.dyndns.org&myip=10.32.254.187'
00:05:39: HTTPDNSUPD: Sending request
00:05:39: HTTPDNSUPD: Response for update test.dyndns.org <=> 10.32.254.187
00:05:39: HTTPDNSUPD: DATA START
nochg 10.32.254.187
00:05:39: HTTPDNSUPD: DATA END, Status is Response data received, successfully
00:05:39: HTTPDNSUPD: Call returned SUCCESS for update test.dyndns.org <=> 10.32.254.187
00:05:39: HTTPDNSUPD: Freeing response
00:05:39: DYNDNSUPD: Another update completed (outstanding=0, total=0)
00:05:39: HTTPDNSUPD: Clearing all session 8 info
```

The table below describes the significant fields shown in the output.

Table 16: debug ip ddns update Field Descriptions

Field	Description
HTTPDNSUPD	Reflects the method of update. In this case, the update method is HTTP.
HTTPDNSUPD: URL =	URL that is used to update the DNS.

Related Commands

Command	Description
debug dhcp	Displays debugging information about the DHCP client and monitors the status of DHCP packets.
debug ip dhcp server	Enables DHCP server debugging.
host (host-list)	Specifies a list of hosts that will receive DDNS updates of A and PTR RRs.
ip ddns update hostname	Enables a host to be used for DDNS updates of A and PTR RRs.
ip ddns update method	Specifies a method of DDNS updates of A and PTR RRs and the maximum interval between the updates.
ip dhcp client update dns	Enables DDNS updates of A RRs using the same hostname passed in the hostname and FQDN options by a client.
ip dhcp-client update dns	Enables DDNS updates of A RRs using the same hostname passed in the hostname and FQDN options by a client.
ip dhcp update dns	Enables DDNS updates of A and PTR RRs for most address pools.
ip host-list	Specifies a list of hosts that will receive DDNS updates of A and PTR RRs.
show ip ddns update	Displays information about the DDNS updates.
show ip ddns update method	Displays information about the DDNS update method.
show ip dhcp server pool	Displays DHCP server pool statistics.
show ip host-list	Displays the assigned hosts in a list.

Command	Description
update dns	Dynamically updates a DNS with A and PTR RRs for some address pools.

debug ip dfp agent

To display debugging messages for the Dynamic Feedback Protocol (DFP) agent subsystem, use the **debug ip dfp**command in user EXEC or privileged EXEC mode. To stop debugging output, use the **no** form of this command.

debug ip dfp agent

no debug ip dfp agent

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values.

Command Modes

User EXEC or privileged EXEC mode

Command History

Release	Modification
12.1(8a)E	This command was introduced.
12.2(14)S	This command was integrated into Cisco IOS Release 12.2(14)S.
12.3(4)T	This command was integrated into Cisco IOS Release 12.3(4)T.
12.2(18)SXD	This command was integrated into Cisco IOS Release 12.2(18)SXD.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

This command displays debugging messages for the DFP agent subsystem.

See the following caution before using debug commands:

Caution Because debugging output is assigned a high priority in the CPU process, it can render the system unusable. For this reason, use debug commands only to troubleshoot specific problems or during troubleshooting sessions with Cisco technical support staff. Moreover, it is best to use debug commands during periods of lower network flows and fewer users. Debugging during these periods reduces the effect these commands have on other users on the system.

Examples

The following example configures a DFP agent debugging session:

```
Router# debug ip dfp agent
DFP debugging is on
```

The following example stops all debugging:

```
Router# no debug all
All possible debugging has been turned off
```

debug ip dhcp server

To enable Cisco IOS Dynamic Host Configuration Protocol (DHCP) server debugging, use the **debug ip dhcp server** command in privileged EXEC mode. To disable DHCP server debugging, use the **no** form of this command.

debug ip dhcp server {**events**| **packets**| **linkage**| **class**}

no debug ip dhcp server {**events**| **packets**| **linkage**| **class**}

Syntax Description

events	Reports server events, such as address assignments and database updates.
packets	Decodes DHCP receptions and transmissions.
linkage	Displays database linkage information, such as parent-child relationships in a radix tree.
class	Displays DHCP class-based information.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(1)T	This command was introduced.
12.2(13)ZH	The **class** keyword was added.
12.3(4)T	The **class** keyword was integrated into Cisco IOS Release 12.3(4)T.
12.3(11)T	The output was enhanced to show the static mappings.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Examples

The following example shows a combination of DHCP server events and decoded receptions and transmissions:

```
Router# debug ip dhcp server events
```

```
Router# debug ip dhcp server packets
```

```
DHCPD:DHCPDISCOVER received from client 0b07.1134.a029 through relay 10.1.0.253.
DHCPD:assigned IP address 10.1.0.3 to client 0b07.1134.a029.
DHCPD:Sending DHCPOFFER to client 0b07.1134.a029 (10.1.0.3).
DHCPD:unicasting BOOTREPLY for client 0b07.1134.a029 to relay 10.1.0.253.
DHCPD:DHCPREQUEST received from client 0b07.1134.a029.
DHCPD:Sending DHCPACK to client 0b07.1134.a029 (10.1.0.3).
```

```
DHCPD:unicasting BOOTREPLY for client 0b07.1134.a029 to relay 10.1.0.253.
DHCPD:checking for expired leases.
```

The following example shows database linkage information:

```
Router# debug ip dhcp server linkage

DHCPD:child pool:10.1.0.0 / 255.255.0.0 (subnet10.1)
DHCPD:parent pool:10.0.0.0 / 255.0.0.0 (net10)
DHCPD:child pool:10.0.0.0 / 255.0.0.0 (net10)
DHCPD:pool (net10) has no parent.
DHCPD:child pool:10.1.0.0 / 255.255.0.0 (subnet10.1)
DHCPD:parent pool:10.0.0.0 / 255.0.0.0 (net10)
DHCPD:child pool:10.0.0.0 / 255.0.0.0 (net10)
DHCPD:pool (net10) has no parent.
```

The following example shows when a DHCP class is removed:

```
Router# debug ip dhcp server class
DHCPD:deleting class CLASS1
```

The following example shows the debug output when the configured pattern does not match:

```
Router# debug ip dhcp server class

DHCPD:Searching for a match to 'relay-information
0106000 4000202020208000600009e80b8800' in class CLASS1
DHCPD:Searching for a match to 'relay-information 01060004000202020208000600009e80b8800' in
 class CLASS1
DHCPD:Searching for a match to 'relay-information 0106000
```

The following example shows the debug output when you unconfigure a DHCP pattern in a DHCP class and then configure the pattern in the DHCP class:

```
Router# debug ip dhcp server class

DHCPD:pattern 'relay-information 123456' removed from class CLASS1
DHCPD:Added pattern 'relay-information 010600040002020202 0800060009e80b8800' for class
CLASS1
```

The following example shows the debug output when the configured pattern does match:

```
Router# debug ip dhcp server class

DHCPD:Searching for a match to 'relay-information
0106000 4000202020208000600009e80b8800' in class CLASS1
DHCPD:input pattern 'relay-information 010600040002020202 0800060009e80b8800' matches class
 CLASS1
DHCPD:input matches class CLASS1
```

The following example shows the debug output when static mappings are configured:

```
Router# debug ip dhcp server
Loading abc/static_pool from 10.19.192.33 (via Ethernet0): !
[OK - 333 bytes]
*May 26 23:14:21.259: DHCPD: contacting agent tftp://10.19.192.33/abc/static_pool (attempt
 0)
*May 26 23:14:21.467: DHCPD: agent tftp://10.19.192.33/abc/static_pool is responding.
*May 26 23:14:21.467: DHCPD: IFS is ready.
*May 26 23:14:21.467: DHCPD: reading bindings from
tftp://10.19.192.33/abc/static_pool.
*May 26 23:14:21.707: DHCPD: read 333 / 1024 bytes.
*May 26 23:14:21.707: DHCPD: parsing text line "*time* Apr 22 2002 11:31 AM"
*May 26 23:14:21.707: DHCPD: parsing text line ""
*May 26 23:14:21.707: DHCPD: parsing text line
!IP address Type Hardware address Lease expiration.
*May 26 23:14:21.707: DHCPD: parsing text line
"10.9.9.1/24 id 0063.6973.636f.2d30.3036.302e.3437"
*May 26 23:14:21.707: DHCPD: creating binding for 10.9.9.1
*May 26 23:14:21.707: DHCPD: Adding binding to radix tree (10.9.9.1)
*May 26 23:14:21.707: DHCPD: Adding binding to hash tree
*May 26 23:14:21.707: DHCPD: parsing text line
```

```
"10.9.9.4 id 0063.7363.2d30.3036.302e.3762.2e39.3634.632d"
*May 26 23:14:21.711: DHCPD: creating binding for 10.9.9.4
*May 26 23:14:21.711: DHCPD: Adding binding to radix tree (10.9.9.4)
*May 26 23:14:21.711: DHCPD: Adding binding to hash tree
*May 26 23:14:21.711: DHCPD: parsing text line "Infinite"
*May 26 23:14:21.711: DHCPD: parsing text line ""
*May 26 23:14:21.711: DHCPD: parsing text line
!IP address Interface-index Lease expiration VRF.
*May 26 23:14:21.711: DHCPD: parsing text line "*end*"
*May 26 23:14:21.711: DHCPD: read static bindings from tftp://10.19.192.33/smith/static_pool.
```

Related Commands

Command	Description
debug dhcp	Displays debugging information about the DHCP client and monitors the status of DHCP packets.
debug ip ddns update	Enables debugging for DDNS updates.
host (host-list)	Specifies a list of hosts that will receive DDNS updates of A and PTR RRs.
ip ddns update hostname	Enables a host to be used for DDNS updates of A and PTR RRs.
ip ddns update method	Specifies a method of DDNS updates of A and PTR RRs and the maximum interval between the updates.
ip dhcp client update dns	Enables DDNS updates of A RRs using the same hostname passed in the hostname and FQDN options by a client on an interface.
ip dhcp-client update dns	Enables DDNS updates of A RRs using the same hostname passed in the hostname and FQDN options by a client.
ip dhcp update dns	Enables DDNS updates of A and PTR RRs for most address pools.
ip host-list	Specifies a list of hosts that will receive DDNS updates of A and PTR RRs.
show ip ddns update	Displays information about the DDNS updates.
show ip ddns update method	Displays information about the DDNS update method.
show ip dhcp server pool	Displays DHCP server pool statistics.
show ip host-list	Displays the assigned hosts in a list.
update dns	Dynamically updates a DNS with A and PTR RRs for some address pools.

debug ip dhcp server redundancy

To display debugging information about DHCP server and relay agent redundancy events, use the **debug ip dhcp server redundancy**command in privileged EXEC mode. To disable the display of debugging output, use the **no** form of this command.

debug ip dhcp server redundancy

no debug ip dhcp server redundancy

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging output is disabled for DHCP server and relay agent redundancy events.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(31)SB2	This command was introduced.

Usage Guidelines

Use this command with caution. Many bindings being synchronized between the active and standby Route Processor (RP) can trigger a large amount of debugging output.

Examples

The following example displays debug messages regarding DHCP server and relay agent redundancy events. The last line (and only that line) is output when the **debug ip dhcp server redundancy** command is enabled. The line indicates that a binding update message has been sent to the standby for the IP address 10.0.0.2 in the pool named "test."

```
Router# debug ip dhcp server redundancy
*Mar 22 10:32:21: DHCPD: assigned IP address 10.0.0.2 to client
0063.6973.636f.2d30.3030.342e.3465.6130.2e30.3831.632d.4661.312f.302e.31.
*Mar 22 10:32:21:  DHCPD: lease time = 3600
*Mar 22 10:32:21: DHCPD: dhcpd_lookup_route: host = 10.0.0.2
*Mar 22 10:32:21: DHCPD: dhcpd_lookup_route: index = 0
*Mar 22 10:32:21: DHCPD: dhcpd_create_and_hash_route: host = 10.0.0.2
*Mar 22 10:32:21: DHCPD: dhcpd_create_and_hash_route index = 0
*Mar 22 10:32:21: DHCPD: dhcpd_add_route: lease = 3600
*Mar 22 10:32:21: DHCPD: dynamic sync completed for 10.0.0.2 in pool test
```

Related Commands

Command	Description
debug dhcp redundancy	Displays debugging information about DHCP proxy client redundancy events.

debug ip dhcp server snmp

To enable DHCP server Simple Network Management Protocol (SNMP) debugging, use the **debug ip dhcp server snmp** command in privileged EXEC mode. To disable DHCP server SNMP debugging, use the **no** form of this command.

debug ip dhcp server snmp

no debug ip dhcp server snmp

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(33)SRC	This command was introduced.

Examples

The following example shows how to enable debugging and display DHCP server SNMP debugging events:

```
Router# debug ip dhcp server snmp
00:18:01: DHCPD SNMP: pool 'pool1' 'high' utilization trap is ignored
00:18:18: DHCPD SNMP: pool 'pool1' 'low' utilization trap is ignored
00:20:46: DHCPD SNMP: subnet 4.1.1.0 'high' utilization trap is ignored
00:21:03: DHCPD SNMP: subnet 4.1.1.0 'low' utilization trap is ignored
00:18:01: DHCPD SNMP: subnet trap is not enabled
00:37:32: DHCPD SNMP: pool trap is not enabled
00:37:57: DHCPD SNMP: interface trap is not enabled
00:27:27: DHCPD SNMP: duplicate trap is not enabled
```

debug ip dns name-list

To enable debugging output for Domain Name System (DNS) name list events, use the **debug ip dns name-list** command in privileged EXEC mode. To disable debugging output for DNS name list events, use the **no** form of this command.

debug ip dns name-list

no debug ip dns name-list

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging output is disabled for DNS name lists.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.4(9)T	This command was introduced.

Usage Guidelines

This command enables the writing of DNS name list event messages to system message logging (syslog) output. A DNS name list event can be either of the following:

- The addition or removal of a DNS name list entry (a hostname pattern and action to perform on an incoming DNS query for a hostname that matches the pattern). To add or remove a DNS name list entry, use the **ip dns name-list** command.
- The removal of a DNS name list.

Note The addition of a DNS name list is reported as an addition of a name list entry.

To display which debugging options are enabled (DNS name list, DNS view, or DNS view list), use the **show debugging** command. To display the syslog history statistics and buffer contents, use the **show logging** command. To display a particular DNS name list or all configured name lists, use the **show ip dns name-list** command.

Examples

The following sample output from the **debug ip dns name-list** command shows the hostname pattern www.example.com being added to DNS name list 1 as a permit clause. Next, the hostname patterns www.example1.com and www.example2.com are added to DNS name list 2 as deny clauses and permit clauses, respectively. Finally, the hostname pattern www.example1.com is removed from DNS name list 2.

```
Router# debug ip dns name-list
```

```
DNS Name-list debugging is on
.
.
.
Router# show debugging

DNS Name-list debugging is on
.
.
.
Router# show logging

.
.
.
*May 16 14:54:44.326: DNS_NAMELIST: adding permit 'WWW.EXAMPLE' to name-list 1
*May 16 14:54:44.910: DNS_NAMELIST: adding deny 'WWW.EXAMPLE1.COM' to name-list 2
*May 16 14:54:45.202: DNS_NAMELIST: adding permit 'WWW.EXAMPLE2.COM' to name-list 2
*May 16 19:32:20.881: DNS_NAMELIST: removing 'WWW.EXAMPLE1.COM' from name-list 2
```

Related Commands

Command	Description
ip dns name-list	Defines a list of pattern-matching rules in which each rule permits or denies the use of a DNS view list member to handle a DNS query based on whether the query hostname matches the specified regular expression.
show debugging	Displays the state of each debugging option.
show ip dns name-list	Displays a particular DNS name list or all configured name lists.
show logging	Displays the contents of logging buffers.

debug ip dns view

To enable debugging output for Domain Name System (DNS) view events, use the **debug ip dns view** command in privileged EXEC mode. To disable debugging output for a DNS view, use the **no** form of this command.

debug ip dns view

no debug ip dns view

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging output is disabled for DNS views.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.4(9)T	This command was introduced.

Usage Guidelines

This command enables the writing of DNS view event messages to system message logging (syslog) output. A DNS view event can be any of the following:

- The addition or removal of a DNS view definition.
- The addition or removal of a DNS forwarding name server setting for a DNS view.
- The addition or removal of a DNS resolver setting for a DNS view.
- The enabling or disabling of logging of a syslog message each time a DNS view is used.

To display which debugging options are enabled (DNS name list, DNS view, or DNS view list), use the **show debugging** command. To show the syslog history statistics and buffer contents, use the **show logging** command.

Examples

The following sample output from the **debug ip dns view** command shows the default DNS view being configured:

```
Router# debug ip dns view

DNS View debugging is on
.
.
.
Router# show debugging

DNS View debugging is on
.
.
```

```
.
Router# show logging

.
.
.
DNS_VIEW: creating view view1
DNS_VIEW: Clearing logging in view default
DNS_VIEW: Setting domain lookup in view default
DNS_VIEW: Setting domain name to cisco.com in view default
DNS_VIEW: Setting  domain list example1.com in view default
DNS_VIEW: Setting  domain list example1.com example2.com in view default
DNS_VIEW: Setting  domain list example1.com example2.com example3.com in view default
DNS_VIEW: Setting domain multicast to 192.0.2.10 in view default
DNS_VIEW: Setting domain lookup in view default
DNS_VIEW: Setting domain timeout to 7 in view default
DNS_VIEW: Setting domain retry to 7 in view default
DNS_VIEW: Setting  domain name-server 192.0.2.204 192.0.2.205 in view default
DNS_VIEW: Setting  domain name-server 192.0.2.204 192.0.2.205 192.0.2.206 in view default
DNS_VIEW: Setting domain name-server interface FastEthernet0/1 in view default
DNS_VIEW: Setting domain round-robin to 4 in view default
DNS_VIEW: Setting dns forwarding in view default
DNS_VIEW: Setting  dns forwarder 192.0.2.11 in view default
DNS_VIEW: Setting  dns forwarder 192.0.2.11 192.0.2.12 in view default
DNS_VIEW: Setting  dns forwarder 192.0.2.11 192.0.2.12 192.0.2.13 in view default
```

Related Commands

Command	Description
ip dns view	Enters DNS view configuration mode for the specified DNS view so that the logging setting, forwarding parameters, and resolving parameters can be configured for the view.
show debugging	Displays the state of each debugging option.
show logging	Displays the contents of logging buffers.

debug ip dns view-list

To enable debugging output for Domain Name System (DNS) view list events, use the **debug ip dns view-list** command in privileged EXEC mode. To disable debugging output for a DNS view list, use the **no** form of this command.

debug ip dns view-list

no debug ip dns view-list

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging output is disabled for DNS view lists.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.4(9)T	This command was introduced.

Usage Guidelines

This command enables the writing of DNS view list event messages to system message logging (syslog) output. A DNS view list event can be any of the following:

- The addition or removal of a DNS view list definition. To add or remove a DNS view list definition, use the **ip dns view-list** command.
- The addition or removal of a DNS view list member (a DNS view and the relative order in which it is to be checked in the view list) to or from a DNS view list. To add or remove a DNS view list member, use the **view** command.
- The setting or clearing of a DNS view list assignment as the default view list (using the **ip dns server view-group** command) or to an interface (using the **ip dns view-group** command).

To show which debugging options are enabled (DNS name list, DNS view, or DNS view list), use the **show debugging** command. To show the syslog history statistics and buffer contents, use the **show logging** command.

Examples

The following sample output from the **debug ip dns vies-list** command shows the addition of the DNS view list definition named userlist5. Next, five DNS views are added as members of the DNS view list.

```
Router# debug ip dns view-list

DNS View-list debugging is on
.
.
.
Router# show debugging
```

```
DNS View-list debugging is on
.
.
.
Router# show logging

*May 16 23:31:17.491: DNS_VIEWLIST: creating view-list userlist5
*May 16 23:31:17.711: DNS_VIEWLIST: adding member user1 vrf vpn101 order 10 to view-list
userlist5
*May 16 23:31:18.583: DNS_VIEWLIST: adding member user2 vrf vpn102 order 20 to view-list
userlist5
*May 16 23:31:19.851: DNS_VIEWLIST: adding member user3 vrf vpn103 order 30 to view-list
userlist5
*May 16 23:31:21.007: DNS_VIEWLIST: adding member user4 vrf vpn204 order 45 to view-list
userlist5
*May 16 23:31:22.199: DNS_VIEWLIST: adding member default order 60 to view-list userlist5
```

Related Commands

Command	Description
ip dns server view-group	Specifies the DNS view list to use to determine which DNS view to use handle incoming queries that arrive on an interface not configured with a DNS view list.
ip dns view-group	Specifies the DNS view list to use to determine which DNS view to use to handle incoming DNS queries that arrive on a specific interface.
ip dns view-list	Enters DNS view list configuration mode so that DNS views can be added to or removed from the ordered list of DNS views.
show debugging	Displays the state of each debugging option.
show logging	Displays the contents of logging buffers.
view	Enters DNS view list member configuration mode so that usage restrictions can be configured for the view list member.

debug ip domain

To enable Domain Name System (DNS) debugging and view DNS debugging information, use the **debug ip domain** command in privileged EXEC mode. To disable DNS debugging, use the **no** form of this command.

debug ip domain

no debug ip domain

Syntax Description

This command has no arguments or keywords.

Note Use the **debug ip domain** command form to enable DNS debugging and view basic DNS debugging information. To view more DNS debugging options such as DNS server response debugging and so on, use the question mark (?) online help function.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS XE Release 3.13S	This command was introduced.
15.4(3)S	This command was integrated into Cisco IOS Release 15.4(3)S.

Examples

The following is sample output from the **debug ip domain** command:

```
Device> enable
Device# debug ip domain

Domain Name System debugging is on
Device#
*Jul 18 09:16:19.546: DNS: Incoming UDP query (id#8168)
*Jul 18 09:16:19.547: DNS: Type 1 DNS query (id#8168) for host 'abc.google.com' from
209.165.200.230(27106)
*Jul 18 09:16:19.547: DNS: Servicing request using view default
*Jul 18 09:16:19.547: search_nametype_index: abc.google.com
*Jul 18 09:16:19.547: search_nametype_index: found abc.google.com for abc.google.com
*Jul 18 09:16:19.547: search_nametype_index: abc.google.com
*Jul 18 09:16:19.547: search_nametype_index: found abc.google.com for abc.google.com
*Jul 18 09:16:19.547: search_nametype_index: google.com
*Jul 18 09:16:19.547: search_nametype_index: com
*Jul 18 09:16:19.547: search_nametype_index: abc.google.com
*Jul 18 09:16:19.547: search_nametype_index: found abc.google.com for abc.google.com
*Jul 18 09:16:19.547: DNS: Reply to client 209.165.200.230/27106 query A
*Jul 18 09:16:19.547: DNS: Finished processing query (id#8168) in 0.001 secs
*Jul 18 09:16:19.547: DNS: Sending response to 209.165.200.230/27106, len 48
```

Related Commands

Command	Description
debug ip domain replies	Enables DNS server response debugging and displays debugging information for DNS server responses to clients.
ip dns server	Enables the DNS server on a device.
ip dns server view-group	Specifies the default DNS server view list for a device.

debug ip domain replies

To enable debugging for Domain Name System (DNS) server responses to clients and view debugging information for DNS server responses to clients, use the **debug ip domain replies** command in privileged EXEC mode. To disable DNS server response debugging, use the **no** form of this command.

debug ip domain replies [**detail**]

no debug ip domain replies [**detail**]

Syntax Description

detail	(Optional) Displays detailed debugging information for DNS server responses to clients.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS XE Release 3.13S	This command was introduced.
15.4(3)S	This command was integrated into Cisco IOS Release 15.4(3)S.

Examples

The following is sample output from the **debug ip domain replies** command:

```
Device> enable
Device# debug ip domain replies

Domain Name System Reply debugging is on


*Jul 18 09:17:22.868: DNS: Finished processing query (id#34422) in 0.000 secs
*Jul 18 09:17:23.663: DNS: Finished processing query (id#51171) in 0.000 secs
*Jul 18 09:17:23.665: DNS: Finished processing query (id#46198) in 0.000 secs
```

Examples

Sample Output for Detailed DNS Response Debugging

```
Device> enable
Device# debug ip domain replies detail

Domain Name System Reply debugging is on (detailed)


*Jul 18 09:17:58.635: DNS: Send reply from internal information:
*Jul 18 09:17:58.635: DOM: id=47025, response, opcode=0, aa=0, tc=0, rd=1, ra=1
*Jul 18 09:17:58.635:      rcode=0, qdcount=1, ancount=1, nscount=0, arcount=0
*Jul 18 09:17:58.635:      query name is abc.google.com, qtype=1, class=1
*Jul 18 09:17:58.635: Answer section:
*Jul 18 09:17:58.635:    Name='abc.google.com'
```

```
*Jul 18 09:17:58.635:    RR type=1, class=1, ttl=10, data length=4
*Jul 18 09:17:58.635:      IP=12.12.12.12
*Jul 18 09:17:58.635: Authority section:
*Jul 18 09:17:58.635: Additional record section:
*Jul 18 09:17:58.635: DNS: Finished processing query (id#47025) in 0.001 secs


*Jul 18 09:17:58.637: DNS: Send reply from internal information:
*Jul 18 09:17:58.637: DOM: id=25881, response, opcode=0, aa=0, tc=0, rd=1, ra=1
*Jul 18 09:17:58.637:      rcode=0, qdcount=1, ancount=1, nscount=0, arcount=0
*Jul 18 09:17:58.637:      query name is abc.google.com, qtype=1, class=1
*Jul 18 09:17:58.637: Answer section:
*Jul 18 09:17:58.637:    Name='abc.google.com'
*Jul 18 09:17:58.637:    RR type=1, class=1, ttl=10, data length=4
*Jul 18 09:17:58.637:      IP=12.12.12.12
*Jul 18 09:17:58.637: Authority section:
*Jul 18 09:17:58.637: Additional record section:
*Jul 18 09:17:58.637: DNS: Finished processing query (id#25881) in 0.001 secs


*Jul 18 09:17:58.638: DNS: Send reply from internal information:
*Jul 18 09:17:58.638: DOM: id=41387, response, opcode=0, aa=0, tc=0, rd=1, ra=1
*Jul 18 09:17:58.638:      rcode=0, qdcount=1, ancount=1, nscount=0, arcount=0
*Jul 18 09:17:58.638:      query name is abc.google.com, qtype=1, class=1
*Jul 18 09:17:58.638: Answer section:
*Jul 18 09:17:58.638:    Name='abc.google.com'
*Jul 18 09:17:58.638:    RR type=1, class=1, ttl=10, data length=4
*Jul 18 09:17:58.638:      IP=12.12.12.12
*Jul 18 09:17:58.638: Authority section:
*Jul 18 09:17:58.638: Additional record section:
*Jul 18 09:17:58.638: DNS: Finished processing query (id#41387) in 0.000 secs
```

Related Commands

Command	Description
debug ip domain	Enables DNS debugging and displays DNS debugging information.
ip dns server	Enables the DNS server on a device.
ip dns server view-group	Specifies the default DNS server view list for a device.

debug ip drp

To display Director Response P rotocol (DRP) information, use the **debug ip drp** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip drp

no debug ip drp

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

The **debug ip drp** command is used to debug the director response agent used by the Distributed Director product. The Distributed Director can be used to dynamically respond to Domain Name System (DNS) queries with the IP address of the "best" host based on various criteria.

Examples

The following is sample output from the **debug ip drp**command. This example shows the packet origination, the IP address that information is routed to, and the route metrics that were returned.

```
Router# debug ip drp
DRP: received v1 packet from 172.69.232.8, via Ethernet0
DRP: RTQUERY for 172.69.58.94 returned internal=0, external=0
```

The table below describes the significant fields shown in the display.

Table 17: debug ip drp Field Descriptions

Field	Description
DRP: received v1 packet from 172.69.232.8, via Ethernet0	Router received a version 1 DRP packet from the IP address shown, via the interface shown.
DRP: RTQUERY for 172.69.58.94	DRP packet contained two Route Query requests. The first request was for the distance to the IP address 171.69.113.50.
internal	If nonzero, the metric for the internal distance of the route that the router uses to send packets in the direction of the client. The internal distance is the distance within the autonomous system of the router.
external	If nonzero, the metric for the Border Gateway Protocol (BGP) or external distance used to send packets to the client. The external distance is the distance outside the autonomous system of the router.

debug ip dvmrp

Note The **debug ip dvmrp**command is not available in 12.2(33)SRB, 15.0(1)M, and later 12.2SR, 15.0M, and T releases.

To display information on Distance Vector Multiprotocol Routing Protocol (DVMRP) packets received and sent, use the **debug ip dvmrp** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip dvmrp [**detail** [*access-list*] [**in**| **out**]]

no debug ip dvmrp [**detail** [*access-list*] [**in**| **out**]]

Syntax Description

detail	(Optional) Enables a more detailed level of output and displays packet contents.
access-list	(Optional) Causes the **debug ip dvmrp** command to restrict output to one access list.
in	(Optional) Causes the **debug ip dvmrp** command to output packets received in DVMRP reports.
out	(Optional) Causes the **debug ip dvmrp** command to output packets sent in DVMRP reports.

Command Modes

Privileged EXEC

Usage Guidelines

Use the debug ip dvmrp detail command with care. This command generates a substantial amount of output and can interrupt other activity on the router when it is invoked.

Examples

The following is sample output from the **debug ip dvmrp** command:

```
Router# debug ip dvmrp
DVMRP: Received Report on Ethernet0 from 172.19.244.10
DVMRP: Received Report on Ethernet0 from 172.19.244.11
DVMRP: Building Report for Ethernet0 224.0.0.4
DVMRP: Send Report on Ethernet0 to 224.0.0.4
DVMRP: Sending IGMP Reports for known groups on Ethernet0
DVMRP: Received Report on Ethernet0 from 172.19.244.10
DVMRP: Received Report on Tunnel0 from 192.168.199.254
DVMRP: Received Report on Tunnel0 from 192.168.199.254
DVMRP: Received Report on Tunnel0 from 192.168.199.254
DVMRP: Received Report on Tunnel0 from 192.168.199.254
DVMRP: Received Report on Tunnel0 from 192.168.199.254
DVMRP: Received Report on Tunnel0 from 192.168.199.254
DVMRP: Building Report for Tunnel0 224.0.0.4
```

```
DVMRP: Send Report on Tunnel0 to 192.168.199.254
DVMRP: Send Report on Tunnel0 to 192.168.199.254
DVMRP: Send Report on Tunnel0 to 192.168.199.254
DVMRP: Send Report on Tunnel0 to 192.168.199.254
DVMRP: Radix tree walk suspension
DVMRP: Send Report on Tunnel0 to 192.168.199.254
```

The following lines show that the router received DVMRP routing information and placed it in the mroute table:

```
DVMRP: Received Report on Ethernet0 from 172.19.244.10
DVMRP: Received Report on Ethernet0 from 172.19.244.11
```

The following lines show that the router is creating a report to send to another DVMRP router:

```
DVMRP: Building Report for Ethernet0 224.0.0.4
DVMRP: Send Report on Ethernet0 to 224.0.0.4
```

The table below provides a list of internet multicast addresses supported for host IP implementations.

Table 18: Internet Multicast Addresses

Address	Description	RFC
224.0.0.0	Base address (reserved)	RFC 1112
224.0.0.1	All systems on this subnet	RFC 1112
224.0.0.2	All routers on this subnet	
224.0.0.3	Unassigned	
224.0.0.4	DVMRP routers	RFC 1075
224.0.0.5	OSPFIGP all routers	RFC 1583

The following lines show that a protocol update report has been sent to all known multicast groups. Hosts use Internet Group Management Protocol (IGMP) reports to communicate with routers and to request to join a multicast group. In this case, the router is sending an IGMP report for every known group to the host, which is running mrouted. The host then responds as though the router were a host on the LAN segment that wants to receive multicast packets for the group.

```
DVMRP: Sending IGMP Reports for known groups on Ethernet0
```

The following is sample output from the **debug ip dvmrp detail** command:

```
Router# debug ip dvmrp detail

DVMRP: Sending IGMP Reports for known groups on Ethernet0
DVMRP: Advertise group 224.2.224.2 on Ethernet0
DVMRP: Advertise group 224.2.193.34 on Ethernet0
DVMRP: Advertise group 224.2.231.6 on Ethernet0
DVMRP: Received Report on Tunnel0 from 192.168.199.254
DVMRP: Origin 150.166.53.0/24, metric 13, distance 0
DVMRP: Origin 150.166.54.0/24, metric 13, distance 0
DVMRP: Origin 150.166.55.0/24, metric 13, distance 0
DVMRP: Origin 150.166.56.0/24, metric 13, distance 0
DVMRP: Origin 150.166.92.0/24, metric 12, distance 0
DVMRP: Origin 150.166.100.0/24, metric 12, distance 0
DVMRP: Origin 150.166.101.0/24, metric 12, distance 0
DVMRP: Origin 150.166.142.0/24, metric 8, distance 0
```

```
DVMRP: Origin 150.166.200.0/24, metric 12, distance 0
DVMRP: Origin 150.166.237.0/24, metric 12, distance 0
DVMRP: Origin 150.203.5.0/24, metric 8, distance 0
```

The following lines show that this group is available to the DVMRP router. The mrouted process on the host will forward the source and multicast information for this group through the DVMRP cloud to other members.

```
DVMRP: Advertise group 224.2.224.2 on Ethernet0
```

The following lines show the DVMRP route information:

```
DVMRP: Origin 150.166.53.0/24, metric 13, distance 0
DVMRP: Origin 150.166.54.0/24, metric 13, distance 0
```

The metric is the number of hops the route has covered, and the distance is the administrative distance.

debug ip eigrp

To display information on Enhanced Interior Gateway Routing Protocol (EIGRP) protocol packets, use the **debug ip eigrp** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip eigrp [**vrf** *vrf-name*]

no debug ip eigrp [**vrf** *vrf-name*]

Syntax Description

vrf *vrf-name*	(Optional) Restricts output to a specific VRF.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(21)S	This command was modified. The **vrf** *vrf-name* keyword and argument were added.

Usage Guidelines

This command helps you analyze the packets that are sent and received on an interface. Because the **debug ip eigrp** command generates a substantial amount of output, only use it when traffic on the network is light.

Examples

The following is sample output from the **debug ip eigrp** command:

```
Router# debug ip eigrp
IP-EIGRP: Processing incoming UPDATE packet
IP-EIGRP: Ext 192.168.3.0 255.255.255.0 M 386560 - 256000 130560 SM 360960 - 256000 104960
IP-EIGRP: Ext 192.168.0.0 255.255.255.0 M 386560 - 256000 130560 SM 360960 - 256000 104960
IP-EIGRP: Ext 192.168.3.0 255.255.255.0 M 386560 - 256000 130560 SM 360960 - 256000 104960
IP-EIGRP: 172.69.43.0 255.255.255.0, - do advertise out Ethernet0/1
IP-EIGRP: Ext 172.69.43.0 255.255.255.0 metric 371200 - 256000 115200
IP-EIGRP: 192.135.246.0 255.255.255.0, - do advertise out Ethernet0/1
IP-EIGRP: Ext 192.135.246.0 255.255.255.0 metric 46310656 - 45714176 596480
IP-EIGRP: 172.69.40.0 255.255.255.0, - do advertise out Ethernet0/1
IP-EIGRP: Ext 172.69.40.0 255.255.255.0 metric 2272256 - 1657856 614400
IP-EIGRP: 192.135.245.0 255.255.255.0, - do advertise out Ethernet0/1
IP-EIGRP: Ext 192.135.245.0 255.255.255.0 metric 40622080 - 40000000 622080
IP-EIGRP: 192.135.244.0 255.255.255.0, - do advertise out Ethernet0/1
```

The table below describes the significant fields shown in the display.

Table 19: debug ip eigrp Field Descriptions

Field	Description
IP-EIGRP:	Indicates that this is an IP EIGRP message.

Field	Description
Ext	Indicates that the following address is an external destination rather than an internal destination, which would be labeled as Int.
M	Displays the computed metric, which includes the value in the SM field and the cost between this router and the neighbor. The first number is the composite metric. The next two numbers are the inverse bandwidth and the delay, respectively.
SM	Displays the metric as reported by the neighbor.

The following example shows how to turn on debugging output for a specific VRF in an EIGRP instance:

```
Router# debug ip eigrp vrf red
EIGRP-IPv4 Route Event debugging is on
```

Related Commands

Command	Description
vrf definition	Defines a virtual routing and forwarding instance.

debug ip eigrp notifications

To display Enhanced Interior Gateway Routing Protocol (EIGRP) events and notifications in the console of the router, use the **debug ip eigrp notifications** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip eigrp notifications

no debug ip eigrp notifications

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(15)T	This command was introduced.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.
15.0(1)M	This command was integrated into Cisco IOS Release 15.0(1)M.

Usage Guidelines

The output of the debug ip eigrp notifications command displays EIGRP events and notifications.

Examples

The following example output shows that the NSF-aware router has received the restart notification. The NSF-aware router will now wait for end of transmission (EOT) to be sent from the restarting neighbor (NSF-capable).

```
Router# debug ip eigrp notifications
*Oct  4 11:39:18.092:EIGRP:NSF:AS2. Rec RS update from 135.100.10.1,
00:00:00. Wait for EOT.
*Oct  4 11:39:18.092:%DUAL-5-NBRCHANGE:IP-EIGRP(0) 2:Neighbor
135.100.10.1 (POS3/0) is up:peer NSF restarted
```

debug ip error

To display IP errors, use the **debug ip error** command in privileged EXEC mode. To disable debugging errors, use the **no** form of this command.

debug ip error *access-list-number* **[detail] [dump]**

no debug ip error

Syntax Description

access-list-number	(Optional) The IP access list number that you can specify. If the datagram is not permitted by that access list, the related debugging output (or IP error) is suppressed. Standard, extended, and expanded access lists are supported. The range of standard and extended access lists is from 1 to 199. The range of expanded access lists is from 1300 to 2699.
detail	(Optional) Displays detailed IP error debugging information.
dump	(Hidden) Displays IP error debugging information along with raw packet data in hexadecimal and ASCII forms. This keyword can be enabled with individual access lists and also with the **detail** keyword. **Note** The **dump** keyword is not fully supported and should be used only in collaboration with Cisco Technical Support. See the caution notes below, in the usage guidelines, for more specific information.

Command Default

No default behavior or values.

Command Modes

Privileged EXEC

Usage Guidelines

This command is used for IP error debugging. The output displays IP errors which are locally detected by this router.

> **Caution** Enabling this command will generate output only if IP errors occur. However, if the router starts to receive many packets that contain errors, substantial output may be generated and severely affect system performance. This command should be used with caution in production networks. It should only be enabled when traffic on the IP network is low, so other activity on the system is not adversely affected. Enabling the **detail** and **dump** keywords use the highest level of system resources of the available configuration options for this command, so a high level of caution should be applied when enabling either of these keywords.

> **Caution** The **dump** keyword is not fully supported and should be used only in collaboration with Cisco Technical Support. Because of the risk of using significant CPU utilization, the dump keyword is hidden from the user and cannot be seen using the "?" prompt. The length of the displayed packet information may exceed the actual packet length and include additional padding bytes that do not belong to the IP packet. Also note that the beginning of a packet may start at different locations in the dump output depending on the specific router, interface type, and packet header processing that may have occurred before the output is displayed.

Examples

The following is sample output from the **debug ip error**command:

```
Router# debug ip error

IP packet errors debugging is on
04:04:45:IP:s=10.8.8.1 (Ethernet0/1), d=10.1.1.1, len 28, dispose ip.hopcount
```

The IP error in the above output was caused when the router attempted to forward a packet with a time-to-live (TTL) value of 0. The "ip.hopcount" traffic counter is incremented when a packet is dropped because of an error. This error is also displayed in the output of the **show ip traffic** command by the "bad hop count" traffic counter.

The table below describes the significant fields shown in the display.

Table 20: debug ip error Field Descriptions

Field	Description
IP:s=10.8.8.1 (Ethernet0/1)	The packet source IP address and interface.
d=10.1.1.1, len 28	The packet destination IP address and prefix length.
dispose ip.hopcount	This traffic counter increments when an IP packet is dropped because of an error.

The following is sample output from the **debug ip error**command enabled with the **detail** keyword:

```
Router# debug ip error detail
```

IP packet errors debugging is on (detailed)

1d08h:IP:s=10.0.19.100 (Ethernet0/1), d=10.1.1.1, len 28, dispose udp.noport

1d08h: UDP src=41921, dst=33434

1d08h:IP:s=10.0.19.100 (Ethernet0/1), d=10.2.2.2, len 28, dispose ip.hopcount

```
1d08h:    UDP src=33691, dst=33434
```

The detailed output includes layer 4 information in addition to the standard output. The IP error in the above output was caused when the router received a UDP packet when no application was listening to the UDP port. The "udp.noport" traffic counter is incremented when the router drops a UDP packet because of this error. This error is also displayed in the output of the **show ip traffic** command by the "no port" traffic counter under "UDP statistics."

The table below describes the significant fields shown in the display.

Table 21: debug ip error detail Field Descriptions

Field	Description
IP:s=10.0.19.100 (Ethernet0/1)	The IP packet source IP address and interface.
d=10.1.1.1, len 28	The IP packet destination and prefix length.
dispose udp.noport	The traffic counter that is incremented when a UDP packet is dropped because of this error.

The following is sample output from the **debug ip error**command enabled with the **detail** and **dump** keywords:

```
Router# debug ip error detail dump
IP packet errors debugging is on (detailed) (dump)
1d08h:IP:s=10.0.19.100 (Ethernet0/1), d=10.1.1.1, len 28, dispose udp.noport
1d08h:    UDP src=37936, dst=33434
03D72360:                      0001 42AD4242            ..B-BB
03D72370:0002FCA5 DC390800 4500001C 30130000  ..|%\9..E...0...
03D72380:01116159 0A001364 0A010101 9430829A  ..aY...d.....0..
03D72390:0008C0AD                             ..@-
1d08h:IP:s=10.0.19.100 (Ethernet0/1), d=10.2.2.2, len 28, dispose ip.hopcount
1d08h:    UDP src=41352, dst=33434
03C01600:                      0001 42AD4242            ..B-BB
03C01610:0002FCA5 DC390800 4500001C 302A0000  ..|%\9..E...0*..
03C01620:01116040 0A001364 0A020202 A188829A  ..`@...d....!...
03C01630:0008B253                             ..2S
```

Note The **dump** keyword is not fully supported and should be used only in collaboration with Cisco Technical Support. See the caution in the usage guidelines section of this command reference page for more specific information.

The output from the **debug ip error** command, when the **dump** keyword is enabled, provides raw packet data in hexadecimal and ASCII forms. This addtional output is displayed in addition to the standard output. The dump keyword can be used with all of the available configuration options of this command.

The table below describes the significant fields shown in the display.

Table 22: debug ip error detail dump Field Descriptions

Field	Description
IP:s=10.0.19.100 (Ethernet0/1)	The IP packet source IP address and interface.

Field	Description
d=10.1.1.1, len 28	The IP packet destination and prefix length.
dispose udp.noport	The traffic counter that is incremented when a UDP packet is dropped because of this error.

Related Commands

Command	Description
show ip traffic	Displays statistics about IP traffic.

debug ip flow cache

To enable debugging output for NetFlow cache, use the **debug ip flow cache** command in user EXEC or privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip flow cache

no debug ip flow cache

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging output for NetFlow data export is disabled.

Command Modes

User EXEC Privileged EXEC

Command History

Release	Modification
12.0(1)	This command was introduced.
12.3(1)	Debugging output for NetFlow v9 data export was added.
12.3(7)T	Debugging output for NetFlow for IPv6 was added.
12.2(30)S	This command was integrated into Cisco IOS Release 12.2(30)S.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.

Examples

The following is sample output from the **debug ip flow export** command:

```
Router# debug ip flow cache
IP Flow cache allocation debugging is on
Router# show ipv6 flow
IP packet size distribution (0 total packets):
   1-32   64   96  128  160  192  224  256  288  320  352  384  416  448  480
   .000 .000 .000 .000 .000 .000 .000 .000 .000 .000 .000 .000 .000 .000 .000
    512  544  576 1024 1536 2048 2560 3072 3584 4096 4608
   .000 .000 .000 .000 .000 .000 .000 .000 .000 .000 .000
IP Flow Switching Cache, 0 bytes
  0 active, 0 inactive, 0 added
  0 ager polls, 0 flow alloc failures
  Active flows timeout in 30 minutes
  Inactive flows timeout in 15 seconds
SrcAddress                                  InpIf    DstAddress
                OutIf    Prot SrcPrt DstPrt Packets
c7200-vxr-2#
```

```
000037: 01:56:26: IPFLOW: Allocating Sub-Flow cache, without hash flags.
000038: 01:56:26: IPFLOW: Sub-Flow table enabled.
000039: 01:56:26: IPFLOW: Sub-Flow numbers are:
    24 sub-flows per chunk, 0 hashflag len,
    1 chunks allocated, 12 max chunks,
    24 allocated records, 24 free records, 960 bytes allocated
000040: 01:56:26: IPFLOW: Sub-Flow cache removed
```

Related Commands

Command	Description
export destination	Enables the exporting of information from NetFlow aggregation caches.
ip flow-aggregation cache	Enables NetFlow aggregation cache schemes.
ip flow-export	Enables the exporting of information in NetFlow cache entries.
ipv6 flow-aggregation cache	Enables NetFlow aggregation cache schemes for IPv6 configurations.
ipv6 flow export	Enables the exporting of information in NetFlow cache entries for IPv6 NetFlow configurations.
show ip cache flow aggregation	Displays the NetFlow aggregation cache configuration.
show ip flow export	Display the statistics for NetFlow data export.

debug ip flow export

To enable debugging output for NetFlow data export, use the **debug ip flow export** command in user EXEC or privileged EXEC mode. To disable debugging output for NetFlow data export, use the **no** form of this command.

debug ip flow export

no debug ip flow export

Syntax Description

This command has no keywords or arguments.

Command Default

Debugging output for NetFlow data export is disabled.

Command Modes

User EXEC Privileged EXEC

Command History

Release	Modification
12.0(1)	This command was introduced.
12.3(1)	Debugging output for NetFlow v9 data export was added.
12.3(7)T	This command was modified so that NetFlow v9 data is collected for both IPv4 and IPv6.
12.2(18)S	This command was integrated into Cisco IOS Release 12.2(18)S.
12.2(30)S	This command was integrated into Cisco IOS Release 12.2(30)S.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(18)SXF	This command was integrated into Cisco IOS Release 12.2(18)SXF.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Examples

The following is sample output from the **debug ip flow export** command:

```
Router# debug ip flow export
IP Flow export mechanism debugging is on
*Mar 6 22:56:21.627:IPFLOW:Sending export pak to 2001::FFFE/64 port 9999
*Mar 6 22:56:21.627:IPFLOW:Error sending export packet:Adjacency failure
```

Related Commands

Command	Description
export destination	Enables the exporting of information from NetFlow aggregation caches.
ipv6 flow-aggregation cache	Enables NetFlow aggregation cache schemes for IPv6.
ipv6 flow-export	Enables the exporting of information in NetFlow cache entries.
show ip cache flow aggregation	Displays the NetFlow accounting aggregation cache statistics.
show ip flow export	Displays the statistics for NetFlow data export.
show ipv6 flow export	Displays the statistics for NetFlow data export for IPv6.

debug ip ftp

To activate the debugging option to track the transactions submitted during an FTP session, use the **debug ip ftp** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip ftp

no debug ip ftp

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

The **debug ip ftp** command is useful for debugging problems associated with FTP.

While configuring the ftp password, only encryption types 0 and 7 are allowed. Other encryption types will invoke an "Invalid encryption type" error.

If encryption type 7 has been chosen, the cli will check if the supplied password is encrypted (encrypted by Cisco proprietary algorithm). If the supplied password is found to be Cisco-encrypted, it will be configured. Otherwise the error "Invalid encrypted password" will be shown. The option 7 expects a Cisco-encrypted password to be supplied in the cli.

While configuring the ftp password, if encryption type 0 has been chosen, the cli will encrypt the password as long as the "service password-encryption" is enabled.

Examples

The following is an example of the **debug ip ftp**command:

```
Router# debug ip ftp
FTP transactions debugging is on
```

The following is sample output from the **debug ip ftp** command:

```
FTP: 220 ProFTPD 1.2.0pre8 Server (DFW Nostrum FTP Server) [defiant.dfw.nostrum.com]
Dec 27 22:12:09.133: FTP: ---> USER router
Dec 27 22:12:09.133: FTP: 331 Password required for router.
Dec 27 22:12:09.137: FTP: ---> PASS WQHK5JY2
Dec 27 22:12:09.153: FTP: 230 Anonymous access granted, restrictions apply.
Dec 27 22:12:09.153: FTP: ---> TYPE I
Dec 27 22:12:09.157: FTP: 200 Type set to I.
Dec 27 22:12:09.157: FTP: ---> PASV
.
.
.
.
.
.
.
.
.
.
.
.
.
.
```

```
Dec 27 22:12:09.173: FTP: ---> QUIT
Dec 27 22:12:09.181: FTP: 221 Goodbye.
```

debug ip http all through debug ip rsvp

debug ip http all

To enable debugging output for all HTTP processes on the system, use the **debug ip http all**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip http all

no debug ip http all

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(15)T	This command was introduced.
12.2(31)SB2	This command was integrated into Cisco IOS Release 12.2(31)SB2.
12.2(33)SRC	This command was integrated into Cisco IOS Release 12.2(33)SRC.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.
12.2(33)SB	This command was integrated into Cisco IOS Release 12.2(33)SB.

Usage Guidelines

Use this command to enable debugging messages for all HTTP processes and activity. Issuing this command is equivalent to issuing the following commands:

- **debug ip http authentication**
- **debug ip http ezsetup**
- **debug ip http ssi**
- **debug ip http token**
- **debug ip http transaction**
- **debug ip http url**

Examples

For sample output and field descriptions of this command, see the documentation of the commands listed in the "Usage Guidelines" section.

Related Commands

Command	Description
debug ip http authentication	Enables debugging output for all processes for HTTP server and client access.
debug ip http ezsetup	Displays the configuration changes that occur during the EZ Setup process.
debug ip http ssi	Displays SSI translations and SSI ECHO command execution.
debug ip http token	Displays individual tokens parsed by the HTTP server.
debug ip http transaction	Displays HTTP server transaction processing.
debug ip http url	Displays the URLs accessed from the router.

debug ip http authentication

To troubleshoot HTTP authentication problems, use the **debug ip http authentication** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip http authentication

no debug ip http authentication

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(15)T	This command was introduced.
12.2(31)SB2	This command was integrated into Cisco IOS Release 12.2(31)SB2.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.

Usage Guidelines

Use this command to display the authentication method the router attempted and authentication-specific status messages.

Examples

The following is sample output from the **debug ip http authentication** command:

```
Router# debug ip http authentication
Authentication for url '/' '/' level 15 privless '/'
Authentication username = 'local15' priv-level = 15 auth-type = local
```

The table below describes the significant fields shown in the display.

Table 23: debug ip http authentication Field Descriptions

Field	Description
Authentication for url	Provides information about the URL in different forms.
Authentication username	Identifies the user.
priv-level	Indicates the user privilege level.

Field	Description
auth-type	Indicates the authentication method.

Related Commands

Command	Description
debug ip http all	Displays authentication processes for all HTTP server processes on the system.
debug ip http ezsetup	Displays the configuration changes that occur during the EZ Setup process.
debug ip http ssi	Displays SSI translations and SSI ECHO command execution.
debug ip http token	Displays individual tokens parsed by the HTTP server.
debug ip http transaction	Displays HTTP server transaction processing.
debug ip http url	Displays the URLs accessed from the router.

debug ip http client

To enable debugging output for the HTTP client, use the **debug ip http client** command in privileged EXEC mode. To disable debugging output for the HTTP client, use the **no** or **undebug** form of this command.

debug ip http client {**all**| **api**| **cache**| **error**| **main**| **msg**| **socket**}

no debug ip http client {**all**| **api**| **cache**| **error**| **main**| **msg**| **socket**}

undebug ip http client {**all**| **api**| **cache**| **error**| **main**| **msg**| **socket**}

Syntax Description

all	Enables debugging for all HTTP client elements.
api	Enables debugging output for the HTTP client application interface (API).
cache	Enables debugging output for the HTTP client cache.
error	Enables debugging output for HTTP communication errors.
main	Enables debugging output specific to the Voice XML (VXML) applications interacting with the HTTP client.
msg	Enables debugging output of HTTP client messages.
socket	Enables debugging output specific to the HTTP client socket.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.3(2)T	This command was introduced.
12.2(31)SB2	This command was integrated into Cisco IOS Release 12.2(31)SB2.
12.2(33)SRC	This command was integrated into Cisco IOS Release 12.2(33)SRC.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.
12.2(33)SB	This command was integrated into Cisco IOS Release 12.2(33)SB.

Usage Guidelines

Use this command to display transactional information for the HTTP client for debugging purposes.

Examples

The following example shows sample debugging output for a failed **copy** transfer operation when the host name resolution fails:

```
Router# debug ip http client all
2w4d: Cache ager called
Router# copy http://www.example.com/index.html flash:index.html

Destination filename [index.html]?
Erase flash: before copying? [confirm] no

Translating "www.example.com"
% Bad IP address for host www.example.com
%Error opening http://www.example.com/index.html (I/O error)
Router#
2w4d: http_client_request:
2w4d: httpc_setup_request:
2w4d: http_client_process_request:
2w4d: HTTPC: Host name resolution failed for www.example.com
2w4d: http_transaction_free:
```

2w4d: http_transaction_free: freed httpc_transaction_t

The following example shows sample debugging output for a failed **copy** transfer operation when the source file is not available:

```
Router# copy http://example.com/hi/file.html flash:/file.html
Destination filename [file.html]?
%Error opening http://example.com/hi/file.html (No such file or directory)
Router#
2w4d: http_client_request:
2w4d: httpc_setup_request:
2w4d: http_client_process_request:
2w4d: httpc_request:Dont have the credentials
Thu, 17 Jul 2003 07:05:25 GMT  http://209.168.200.225/hi/file.html ok
        Protocol = HTTP/1.1
        Content-Type = text/html; charset=iso-8859-1
        Date = Thu, 17 Jul 2003 14:24:29 GMT
2w4d: http_transaction_free:
2w4d: http_transaction_free:freed httpc_transaction_t
2w4d: http_client_abort_request:
2w4d: http_client_abort_request:Bad Transaction Id
Router#
```

The table below describes the significant fields shown in the display.

Table 24: debug ip http client Field Descriptions

Field	Description
2w4d:	In the examples shown, the string “2w4d” is the timestamp configured on the system. Indicates two weeks and four days since the last system reboot. • The time-stamp format is configured using the **service timestamps debug** global configuration mode command.

Field	Description
HTTPC: or httpc	Indicates the HTTP client in Cisco IOS software.
httpc_request:Dont have the credentials	Indicates that this HTTP client request did not supply any authentication information to the server. The authentication information consists of a username and password combination. The message is applicable to both HTTP and HTTPS.
Thu, 17 Jul 2003 07:05:25 GMT http://209.168.200.225/hi/file.html ok	The "ok" in this line indicates that there were no internal errors relating to processing this HTTP client transaction by the HTTP client. In other words, the HTTP client was able to send the request and receive some response. **Note** The "ok" value in this line does not indicate file availability ("200: OK" message or "404: File Not Found" message).

Related Commands

Command	Description
copy	Copies a file from any supported remote location to a local file system, or from a local file system to a remote location, or from a local file system to a local file system.
ip http client connection	Configures the HTTP client connection.
ip http client password	Configures a password for all HTTP client connections.
ip http client proxy-server	Configures an HTTP proxy server.
ip http client source-interface	Configures a source interface for the HTTP client.
ip http client username	Configures a login name for all HTTP client connections.
service timestamps	Configures the time-stamping format for debugging or system logging messages.
show ip http client connection	Displays a report about HTTP client active connections.

Command	Description
show ip http client history	Displays the URLs accessed by the HTTP client.
show ip http client session-module	Displays a report about sessions that have registered with the HTTP client.

debug ip http client cookie

To debug the HTTP client cookie, use the **debug ip http client cookie**command in privileged EXEC mode. To disable this debugging activity, use the **no** form of this command.

debug ip http client cookie

no debug ip http client cookie

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.4(20)T	This command was introduced.

Examples

The following is sample output from the **debug ip http client cookie** command:

```
Device# debug ip http client cookie
ClientCookie: Receiving Set-Cookie cookie1=1 domain=172.16.0.2 path=/cwmp-1-0/testacs
flags=264 expire=Mon,30-Jun-2008 05:51:27 GMT now=48686D74
ClientCookie2: Receiving Set-Cookie2 cookie1= 1  domain=172.16.0.2 path=/cwmp-1-0/ flags=256
 expire=60 port=0 now=48686E1A comment= commentURL=
```

debug ip http ezsetup

To display the configuration changes that occur during the EZ Setup process, use the **debug ip http ezsetup**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip http ezsetup

no debug ip http ezsetup

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(2)T	This command was introduced.
12.2(31)SB2	This command was integrated into Cisco IOS Release 12.2(31)SB2.

Usage Guidelines

Use this command to verify the EZ Setup actions without changing the configuration of the router.

EZ Setup is a form you fill out to perform basic router configuration from most HTML browsers.

Examples

The following sample output from the **debug ip http ezsetup**command shows the configuration changes for the router when the EZ Setup form has been submitted:

```
Router# debug ip http ezsetup
service timestamps debug
service timestamps log
service password-encryption
!
hostname router-name
!
enable secret router-pw
line vty 0 4
password router-pw
!
interface ethernet 0
 ip address 172.69.52.9 255.255.255.0
 no shutdown
 ip helper-address 172.31.2.132
ip name-server 172.31.2.132
isdn switch-type basic-5ess
username Remote-name password Remote-chap
interface bri 0
 ip unnumbered ethernet 0
 encapsulation ppp
 no shutdown
 dialer map ip 192.168.254.254 speed 56 name Remote-name Remote-number
 isdn spid1 spid1
 isdn spid2 spid2
 ppp authentication chap callin
```

```
 dialer-group 1
!
ip classless
access-list 101 deny udp any any eq snmp
access-list 101 deny udp any any eq ntp
access-list 101 permit ip any any
dialer-list 1 list 101
ip route 0.0.0.0 0.0.0.0 192.168.254.254
ip route 192.168.254.254 255.255.255.255 bri 0
logging buffered
snmp-server community public RO
ip http server
ip classless
ip subnet-zero
!
end
```

Related Commands

Command	Description
debug ip http all	Displays authentication processes for all HTTP server processes on the system.
debug ip http authentication	Displays authentication processes for HTTP server and client access.
debug ip http ssi	Displays SSI translations and SSI ECHO command execution.
debug ip http token	Displays individual tokens parsed by the HTTP server.
debug ip http transaction	Displays HTTP server transaction processing.
debug ip http url	Displays the URLs accessed from the router.

debug ip http secure-all

To generate the following output, use the **debug ip http secure-all** command in privileged EXEC mode:

- The debugging information generated by the **debug ip http secure-session** command
- The debugging information generated by the **debug ip http secure-state**command
- Debugging information for each call to the SSL driver, for use primarily by Cisco support personnel

To disable this debugging, use the **no** form of this command.

debug ip http secure-all

no debug ip http secure-all

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.1(11b)E	This command was introduced.
12.2(14)S	This command was integrated into Cisco IOS Release 12.2(14)S.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

This command generates the following:

- The debugging information generated by the **debug ip http secure-session** command. See the **debug ip http secure-session** command page for example debugging output.
- The debugging information generated by the **debug ip http secure-state** command. See the **debug ip http secure-state**command page for example debugging output.
- Debugging information for each call to the SSL driver, for use primarily by Cisco support personnel

Examples

The following example generates the following output:

- The debugging information generated by the **debug ip http secure-session** command
- The debugging information generated by the **debug ip http secure-state** command

- Debugging information for each call to the SSL driver

```
Router# debug ip http secure-all
```

Related Commands

Command	Description
debug ip http secure-session	Generates debugging information about each new secure HTTPS session when it is created.
debug ip http secure-state	Generates debugging information each time the secure HTTPS server changes state.

debug ip http secure-session

To generate debugging information about each new secure HTTPS session when it is created, use the **debug ip http secure-session command** in privileged EXEC mode. To disable this debugging, use the **no** form of this command.

debug ip http secure-session

no debug ip http secure-session

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.1(11b)E	This command was introduced.
12.2(14)S	This command was integrated into Cisco IOS Release 12.2(14)S.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

This command generates debugging information about each new HTTPS session when it is created. When a new HTTPS session is created, debugging information is generated in the following format:

```
HTTPS SSL Session Established/Handshake done - Peer 10.0.0.1
state = SSL negotiation finished successfully
SessionInfo: Digest=RC4-MD5 SSLVer=SSLv3 KeyEx=RSA Auth=RSA Cipher=RC4(128) Mac=MD5
```

The SessionInfo fields provide the following information about the session:

- **Digest--** digest mechanism
- **SSLVer--** SSL or TSL version
- **KeyEx--** key exchange mechanism
- **Auth--** authentication mechanism
- **Cipher--** encryption algorithm
- **Mac--** Message Authentication Code algorithm

Examples

The following example generates debugging information about each new HTTPS session when it is created:

```
debug ip http secure-session
```

Related Commands

Command	Description
debug ip http secure-all	Enables all other debugging **ip http secure-** *x* commands and generates debugging information for each call to the HTTPS server driver.
debug ip http secure-state	Generates debugging information each time the HTTPS server changes state.

debug ip http secure-state

To generate debugging output each time the Secure HTTP (HTTPS) feature changes state, use the **debug ip http secure-state command** in privileged EXEC mode. To disable this debugging, use the **no** form of this command.

debug ip http secure-state

no debug ip http secure-state

Syntax Description

This command has no keywords or arguments.

Command Default

Disabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.1(11b)E	This command was introduced.
12.2(14)S	This command was integrated into Cisco IOS Release 12.2(14)S.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

This command generates debugging information each time the Secure HTTP (HTTPS) feature changes state. When the Secure HTTP (HTTPS) feature changes state, debugging information is generated in the following format:

```
HTTPS SSL State  Change - Peer 10.0.0.1
Old State = SSLv3 read finished A, New State = SSL negotiation finished successfully
```

Examples

The following example generates debugging information each time the Secure HTTP (HTTPS) feature changes state:

```
debug ip http secure-state
```

Related Commands

Command	Description
debug ip http secure-all	Enables all other debugging **ip http secure-** *x* commands and generates debugging information for each call to the HTTPS server driver.

Command	Description
debug ip http secure-state	Generates debugging information each time the HTTPS server changes state.

debug ip http ssi

To display information about the HTML SSI EXEC command or HTML SSI ECHO command, use the **debug ip http ssi** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip http ssi

no debug ip http ssi

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(2)T	This command was introduced.
12.2(31)SB2	This command was integrated into Cisco IOS Release 12.2(31)SB2.

Examples

The following is sample output from the **debug ip http ssi** command:

```
Router# debug ip http ssi
HTML: filtered command ‘exec cmd="show users"’
HTML: SSI command ‘exec’
HTML: SSI tag ‘cmd’ = "show users"
HTML: Executing CLI ‘show users’ in mode ‘exec’ done
```

The following line shows the contents of the SSI EXEC command:

```
HTML: filtered command ‘exec cmd="show users"’
```

The following line indicates the type of SSI command that was requested:

```
HTML: SSI command ‘exec’
```

The following line shows the *show users* argumentassigned to the **tag** *command*:

```
HTML: SSI tag ’cmd’ = "show users"
```

The following line indicates that the **show users** command is being executed in EXEC mode:

```
HTML: Executing CLI ‘show users’ in mode ‘exec’ done
```

Related Commands

Command	Description
debug ip http all	Displays authentication processes for all HTTP server processes on the system.

Command	Description
debug ip http authentication	Displays authentication processes for HTTP server and client access.
debug ip http ezsetup	Displays the configuration changes that occur during the EZ Setup process.
debug ip http token	Displays individual tokens parsed by the HTTP server.
debug ip http transaction	Displays HTTP server transaction processing.
debug ip http url	Displays the URLs accessed from the router.

debug ip http ssl error

To enable debugging messages for the secure HTTP (HTTPS) web server and client, use the **debug ip http ssl error** command in privileged EXEC mode. To disable debugging messages for the HTTPS web server and client, use the **no** form of this command.

debug ip http ssl error

no debug ip http ssl error

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging message output is disabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(15)T	This command was introduced.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.
12.2(33)SB	This command was integrated into Cisco IOS Release 12.2(33)SB.

Usage Guidelines

This command displays output for debugging purposes related to the HTTPS server and HTTPS client. HTTPS services use the Secure Socket Layer (SSL) protocol, version 3.0, for encryption.

Examples

The following is sample debugging output from the **debug ip http ssl error** command:

```
Router# 000030:00:08:01:%HTTPS:Key pair generation failed
Router# 000030:00:08:10:%HTTPS:Failed to generate self-signed cert
Router# 000030:00:08:15:%HTTPS:SSL handshake fail
Router# 000030:00:08:21:%HTTPS:SSL read fail, uninitialized hndshk ctxt
Router# 000030:00:08:25:%HTTPS:SSL write fail, uninitialized hndshk ctxt
```

The table below describes the debug messages shown above.

Table 25: debug ip http ssl error Field Descriptions

Field	Description
%HTTPS:Key pair generation failed	The RSA key pair generation failed.
%HTTPS:Failed to generate self-signed cert	The HTTPS server or client failed to generate a self-signed certificate.
%HTTPS:SSL handshake fail	SSL connection handshake failed.
%HTTPS:SSL read fail, uninitialized hndshk ctxt	A read operation failed for SSL with an unitialized handshake context

Related Commands

Command	Description
ip http secure-server	Enables the secure HTTP (HTTPS) server.

debug ip http token

To display individual tokens parsed by the HTTP server, use the **debug ip http token**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip http token

no debug ip http token

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(2)T	This command was introduced.
12.2(31)SB2	This command was integrated into Cisco IOS Release 12.2(31)SB2.

Usage Guidelines

Use the **debug ip http token** command to display low-level HTTP server parsings. To display high-level HTTP server parsings, use the **debug ip http transaction** command.

Examples

The following is part of sample output from the **debug ip http token** command. In this example, the browser accessed the router's home page http://router-name/. The output gives the token parsed by the HTTP server and its length.

```
Router# debug ip http token
HTTP: token len 3: 'GET'
HTTP: token len 1: ' '
HTTP: token len 1: '/'
HTTP: token len 1: ' '
HTTP: token len 4: 'HTTP'
HTTP: token len 1: '/'
HTTP: token len 1: '1'
HTTP: token len 1: '.'
HTTP: token len 1: '0'
HTTP: token len 2: '\15\12'
HTTP: token len 7: 'Referer'
HTTP: token len 1: ':'
HTTP: token len 1: ' '
HTTP: token len 4: 'http'
HTTP: token len 1: ':'
HTTP: token len 1: '/'
HTTP: token len 1: '/'
HTTP: token len 3: 'www'
HTTP: token len 1: '.'
HTTP: token len 3: 'thesite'
HTTP: token len 1: '.'
HTTP: token len 3: 'com'
HTTP: token len 1: '/'
HTTP: token len 2: '\15\12'
```

```
HTTP: token len 10: 'Connection'
HTTP: token len 1: ':'
HTTP: token len 1: ' '
HTTP: token len 4: 'Keep'
HTTP: token len 1: '-'
HTTP: token len 5: 'Alive'
HTTP: token len 2: '\15\12'
HTTP: token len 4: 'User'
HTTP: token len 1: '-'
HTTP: token len 5: 'Agent'
HTTP: token len 1: ':'
HTTP: token len 1: ' '
HTTP: token len 7: 'Mozilla'
HTTP: token len 1: '/'
HTTP: token len 1: '2'
HTTP: token len 1: '.'
.
.
.
```

Related Commands

Command	Description
debug ip http all	Displays authentication processes for all HTTP server processes on the system.
debug ip http authentication	Displays authentication processes for HTTP server and client access.
debug ip http ezsetup	Displays the configuration changes that occur during the EZ Setup process.
debug ip http ssi	Displays SSI translations and SSI ECHO command execution.
debug ip http transaction	Displays HTTP server transaction processing.
debug ip http url	Displays the URLs accessed from the router.

debug ip http transaction

To display HTTP server transaction processing, use the **debug ip http transaction**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip http transaction

no debug ip http transaction

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(2)T	This command was introduced.
12.2(31)SB2	This command was integrated into Cisco IOS Release 12.2(31)SB2.

Usage Guidelines

Use the **debug ip http transaction** command to display what the HTTP server is parsing at a high level. To display what the HTTP server is parsing at a low level, use the **debug ip http toke n** command.

Examples

The following is sample output from the **debug ip http transaction** command. In this example, the browser accessed the router's home page http://router-name/.

```
Router# debug ip http transaction
HTTP: parsed uri '/'
HTTP: client version 1.1
HTTP: parsed extension Referer
HTTP: parsed line http://www.company.com/
HTTP: parsed extension Connection
HTTP: parsed line Keep-Alive
HTTP: parsed extension User-Agent
HTTP: parsed line Mozilla/2.01 (X11; I; FreeBSD 2.1.0-RELEASE i386)
HTTP: parsed extension Host
HTTP: parsed line router-name
HTTP: parsed extension Accept
HTTP: parsed line image/gif, image/x-xbitmap, image/jpeg, image/
HTTP: parsed extension Authorization
HTTP: parsed authorization type Basic
HTTP: received GET ''
```

The table below describes the significant fields shown in the display.

Table 26: debug ip http transaction Field Descriptions

Field	Description
HTTP: parsed uri '/'	Uniform resource identifier that is requested.

Field	Description
HTTP: client version 1.1	Client HTTP version.
HTTP: parsed extension Referer	HTTP extension.
HTTP: parsed line http://www.company.com/	Value of HTTP extension.
HTTP: received GET ''	HTTP request method.

Related Commands

Command	Description
debug ip http all	Displays authentication processes for all HTTP server processes on the system.
debug ip http authentication	Displays authentication processes for HTTP server and client access.
debug ip http ezsetup	Displays the configuration changes that occur during the EZ Setup process.
debug ip http token	Displays individual tokens parsed by the HTTP server.
debug ip http ssi	Displays SSI translations and SSI ECHO command execution.
debug ip http url	Displays the URLs accessed from the router.

debug ip http url

To show the URLs accessed from the router, use the **debug ip http url** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip http url

no debug ip http url

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(2)T	This command was introduced.
12.2(31)SB2	This command was integrated into Cisco IOS Release 12.2(31)SB2.

Usage Guidelines

Use the **debug ip http url**command to keep track of the URLs that are accessed and to determine from which hosts the URLs are accessed.

Examples

The following is sample output from the **debug ip http url**command. In this example, the HTTP server accessed the URLs and /exec. The output shows the URL being requested and the IP address of the host requesting the URL.

```
Router# debug ip http url
HTTP: processing URL '/' from host 172.31.2.141
HTTP: processing URL '/exec' from host 172.31.2.141
```

Related Commands

Command	Description
debug ip http all	Displays authentication processes for all HTTP server processes on the system.
debug ip http authentication	Displays authentication processes for HTTP server and client access.
debug ip http ezsetup	Displays the configuration changes that occur during the EZ Setup process.
debug ip http ssi	Displays SSI translations and SSI ECHO command execution.

Command	Description
debug ip http token	Displays individual tokens parsed by the HTTP server.
debug ip http transaction	Displays HTTP server transaction processing.

debug ip icmp

To display information on Internal Control Message Protocol (ICMP) transactions, use the **debug ip icmp** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip icmp

no debug ip icmp

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

This command helps you determine whether the router is sending or receiving ICMP messages. Use it, for example, when you are troubleshooting an end-to-end connection problem.

Note For more information about the fields in **debug ip icmp** command output, refer to RFC 792, *Internet Control Message Protocol* ; Appendix I of RFC 950, *Internet Standard Subnetting Procedure* ; and RFC 1256, *ICMP Router Discovery Messages* .

Examples

The following is sample output from the **debug ip icmp** command:

```
Router# debug ip icmp
ICMP: rcvd type 3, code 1, from 10.95.192.4
ICMP: src 10.56.0.202, dst 172.69.16.1, echo reply
ICMP: dst (10.120.1.0) port unreachable rcv from 10.120.1.15
ICMP: src 172.69.12.35, dst 172.69.20.7, echo reply
ICMP: dst (255.255.255.255) protocol unreachable rcv from 10.31.7.21
ICMP: dst (10.120.1.0) port unreachable rcv from 10.120.1.15
ICMP: dst (255.255.255.255) protocol unreachable rcv from 10.31.7.21
ICMP: dst (10.120.1.0) port unreachable rcv from 10.120.1.15
ICMP: src 10.56.0.202, dst 172.69.16.1, echo reply
ICMP: dst (10.120.1.0) port unreachable rcv from 10.120.1.15
ICMP: dst (255.255.255.255) protocol unreachable rcv from 10.31.7.21
ICMP: dst (10.120.1.0) port unreachable rcv from 10.120.1.15
```

The table below describes the significant fields shown in the display.

Table 27: debug ip icmp Field Descriptions

Field	Description
ICMP:	Indication that this message describes an ICMP packet.

Field	Description
rcvd type 3	The type field can be one of the following: • 0--Echo Reply • 3--Destination Unreachable • 4--Source Quench • 5--Redirect • 8--Echo • 9--Router Discovery Protocol Advertisement • 10--Router Discovery Protocol Solicitations • 11--Time Exceeded • 12--Parameter Problem • 13--Timestamp • 14--Timestamp Reply • 15--Information Request • 16--Information Reply • 17--Mask Request • 18--Mask Reply

Field	Description
code 1	This field is a code. The meaning of the code depends upon the type field value, as follows: • Echo and Echo Reply--The code field is always zero. • Destination Unreachable--The code field can have the following values: 0--Network unreachable 1--Host unreachable 2--Protocol unreachable 3--Port unreachable 4--Fragmentation needed and DF bit set 5--Source route failed • Source Quench--The code field is always 0. • Redirect--The code field can have the following values: 0--Redirect datagrams for the network 1--Redirect datagrams for the host 2--Redirect datagrams for the command mode of service and network 3--Redirect datagrams for the command mode of service and host • Router Discovery Protocol Advertisements and Solicitations--The code field is always zero.

Field	Description
	• Time Exceeded--The code field can have the following values: 0--Time to live exceeded in transit 1--Fragment reassembly time exceeded • Parameter Problem--The code field can have the following values: 0--General problem 1--Option is missing 2--Option missing, no room to add • Timestamp and Timestamp Reply--The code field is always zero. • Information Request and Information Reply--The code field is always zero. • Mask Request and Mask Reply--The code field is always zero.
from 10.95.192.4	Source address of the ICMP packet.

The table below describes the significant fields shown in the second line of the display.

Table 28: debug ip icmp Field Descriptions

Field	Description
ICMP:	Indicates that this message describes an ICMP packet.
src 10.56.10.202	Address of the sender of the echo.
dst 172.69.16.1	Address of the receiving router.
echo reply	Indicates that the router received an echo reply.

Other messages that the **debug ip icmp** command can generate follow.

When an IP router or host sends out an ICMP mask request, the following message is generated when the router sends a mask reply:

```
ICMP: sending mask reply (255.255.255.0) to 172.69.80.23 via Ethernet0
```

The following two lines are examples of the two forms of this message. The first form is generated when a mask reply comes in after the router sends out a mask request. The second form occurs when the router receives a mask reply with a nonmatching sequence and ID. Refer to Appendix I of RFC 950, Internet Standard Subnetting Procedures, for details.

```
ICMP: mask reply 255.255.255.0 from 172.69.80.31
ICMP: unexpected mask reply 255.255.255.0 from 172.69.80.32
```

The following output indicates that the router sent a redirect packet to the host at address 172.69.80.31, instructing that host to use the gateway at address 172.69.80.23 in order to reach the host at destination address 172.69.1.111:

```
ICMP: redirect sent to 172.69.80.31 for dest 172.69.1.111 use gw 172.69.80.23
```

The following message indicates that the router received a redirect packet from the host at address 172.69.80.23, instructing the router to use the gateway at address 172.69.80.28 in order to reach the host at destination address 172.69.81.34:

```
ICMP: redirect rcvd from 172.69.80.23 -- for 172.69.81.34 use gw 172.69.80.28
```

The following message is displayed when the router sends an ICMP packet to the source address (172.69.94.31 in this case), indicating that the destination address (172.69.13.33 in this case) is unreachable:

```
ICMP: dst (172.69.13.33) host unreachable sent to 172.69.94.31
```

The following message is displayed when the router receives an ICMP packet from an intermediate address (172.69.98.32 in this case), indicating that the destination address (172.69.13.33 in this case) is unreachable:

```
ICMP: dst (172.69.13.33) host unreachable rcv from 172.69.98.32
```

Depending on the code received (as the first table above describes), any of the unreachable messages can have any of the following “strings” instead of the “host” string in the message:

```
net
protocol
port
frag. needed and DF set
source route failed
prohibited
```

The following message is displayed when the TTL in the IP header reaches zero and a time exceed ICMP message is sent. The fields are self-explanatory.

```
ICMP: time exceeded (time to live) send to 10.95.1.4 (dest was 172.69.1.111)
```

The following message is generated when parameters in the IP header are corrupted in some way and the parameter problem ICMP message is sent. The fields are self-explanatory.

```
ICMP: parameter problem sent to 128.121.1.50 (dest was 172.69.1.111)
```

Based on the preceding information, the remaining output can be easily understood:

```
ICMP: parameter problem rcvd 172.69.80.32
ICMP: source quench rcvd 172.69.80.32
ICMP: source quench sent to 128.121.1.50 (dest was 172.69.1.111)
ICMP: sending time stamp reply to 172.69.80.45
ICMP: sending info reply to 172.69.80.12
ICMP: rdp advert rcvd type 9, code 0, from 172.69.80.23
ICMP: rdp solicit rcvd type 10, code 0, from 172.69.80.43
```

debug ip igmp

To display Internet Group Management Protocol (IGMP) packets received and sent, and IGMP-host related events, use the **debug ip igmp** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip igmp [**vrf** *vrf-name*] [*group-address*]

no debug ip igmp [**vrf** *vrf-name*] [*group-address*]

Syntax Description

vrf	(Optional) Supports the multicast Virtual Private Network (VPN) routing and forwarding (VRF) instance.
vrf-name	(Optional) Name assigned to the VRF.
group-address	(Optional) Address of a particular group about which to display IGMP information.

Command Modes

Privileged EXEC

Command History

Release	Modification
10.2	This command was introduced.
12.1(3)T	Fields were added to the output of this command to support the Source Specific Multicast (SSM) feature.
12.0(23)S	The **vrf** keyword and *vrf-name* argument were added.
12.2(13)T	The **vrf** keyword and *vrf-name* argument were added.
12.2(14)S	This command was integrated into Cisco IOS Release 12.2(14)S.
12.3(2)T	Fields were added to the output of this command to support the SSM Mapping feature. The *group-address* attribute was added.
12.2(18)SXD3	This command was integrated into Cisco IOS Release 12.2(18)SXD3.
12.2(27)SBC	This command was integrated into Cisco IOS Release 12.2(27)SBC.

Usage Guidelines

This command helps discover whether the IGMP processes are functioning. In general, if IGMP is not working, the router process never discovers that another host is on the network that is configured to receive multicast packets. In dense mode, this situation will result in packets being delivered intermittently (a few every 3 minutes). In sparse mode, packets will never be delivered.

Use this command in conjunction with the **debug ip pim** and **debug ip mrouting** commands to observe additional multicast activity and to learn the status of the multicast routing process, or why packets are forwarded out of particular interfaces.

When SSM mapping is enabled, a debug message is displayed to indicate that the router is converting an IGMP version 2 report from the group (G) into an IGMP version 3 join. After SSM mapping has generated the appropriate IGMP version 3 report, any debug output that follows is seen as if the router had received the same IGMP version 3 report directly.

Examples

The following is sample output from the **debug ip igmp** command:

```
Router# debug ip igmp
IGMP: Received Host-Query from 172.16.37.33 (Ethernet1)
IGMP: Received Host-Report from 172.16.37.192 (Ethernet1) for 224.0.255.1
IGMP: Received Host-Report from 172.16.37.57 (Ethernet1) for 224.2.127.255
IGMP: Received Host-Report from 172.16.37.33 (Ethernet1) for 225.2.2.2
```

The messages displayed by the **debug ip igmp** command show query and report activity received from other routers and multicast group addresses.

The following is sample output from the **debug ip igmp** command when SSM is enabled. Because IGMP version 3 lite (IGMPv3lite) requires the host to send IGMP version 2 (IGMPv2) packets, IGMPv2 host reports also will be displayed in response to the router IGMPv2 queries. If SSM is disabled, the word “ignored” will be displayed in the **debug ip igmp** command output.

```
IGMP:Received v3-lite Report from 10.0.119.142 (Ethernet3/3), group count 1
IGMP:Received v3 Group Record from 10.0.119.142 (Ethernet3/3) for 232.10.10.10
IGMP:Update source 224.1.1.1
IGMP:Send v2 Query on Ethernet3/3 to 224.0.0.1
IGMP:Received v2 Report from 10.0.119.142 (Ethernet3/3) for 232.10.10.10
IGMP:Update source 224.1.1.1
```

The following is sample output from the **debug ip igmp**command when SSM static mapping is enabled. The following output indicates that the router is converting an IGMP version 2 join for group (G) into an IGMP version 3 join:

```
IGMP(0): Convert IGMPv2 report (*,232.1.2.3) to IGMPv3 with 2 source(s) using STATIC.
```

The following is sample output from the **debug ip igmp** command when SSM DNS-based mapping is enabled. The following output indicates that a DNS lookup has succeeded:

```
IGMP(0): Convert IGMPv2 report (*,232.1.2.3) to IGMPv3 with 2 source(s) using DNS.
```

The following is sample output from the **debug ip igmp** command when SSM DNS-based mapping is enabled and a DNS lookup has failed:

```
IGMP(0): DNS source lookup failed for (*, 232.1.2.3), IGMPv2 report failed
```

Related Commands

Command	Description
debug ip mrm	Displays MRM control packet activity.
debug ip mrouting	Displays changes to the mroute table.

Command	Description
debug ip pim	Displays PIM packets received and sent and PIM-related events.

debug ip igmp snooping

To display debugging messages about Internet Group Management Protocol (IGMP) snooping services, use the **debug ip igmp snooping** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip igmp snooping {**group**| **management**| **router**| **timer**}

no debug ip igmp snooping {**group**| **management**| **router**| **timer**}

Syntax Description

group	Displays debugging messages related to multicast groups.
management	Displays debugging messages related to IGMP management services.
router	Displays debugging messages related to the local router.
timer	Displays debugging messages related to the IGMP timer.

Command Default

Debugging is disabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.1(6)EA2	This command was introduced.
12.2(15)ZJ	This command was implemented on the following platforms: Cisco 2600 series, Cisco 3600 series, and Cisco 3700 series routers.
12.3(4)T	This command was integrated into Cisco IOS Release 12.3(4)T on the following platforms: Cisco 2600 series, Cisco 3600 series, and Cisco 3700 series routers.

Examples

The following example shows debugging messages for the IGMP snooping services being displayed:

```
Router# debug ip igmp snooping
IGMP snooping enabled
```

Related Commands

Command	Description
show ip igmp snooping	Displays the IGMP snooping configuration.

debug ip igrp events

To display summary information on Interior Gateway Routing Protocol (IGRP) routing messages that indicate the source and destination of each update, and the number of routes in each update, use the **debug ip igrp events** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip igrp events [*ip-address*]

no debug ip igrp events [*ip-address*]

Syntax Description

ip-address	(Optional) The IP address of an IGRP neighbor.

Command Modes

Privileged EXEC

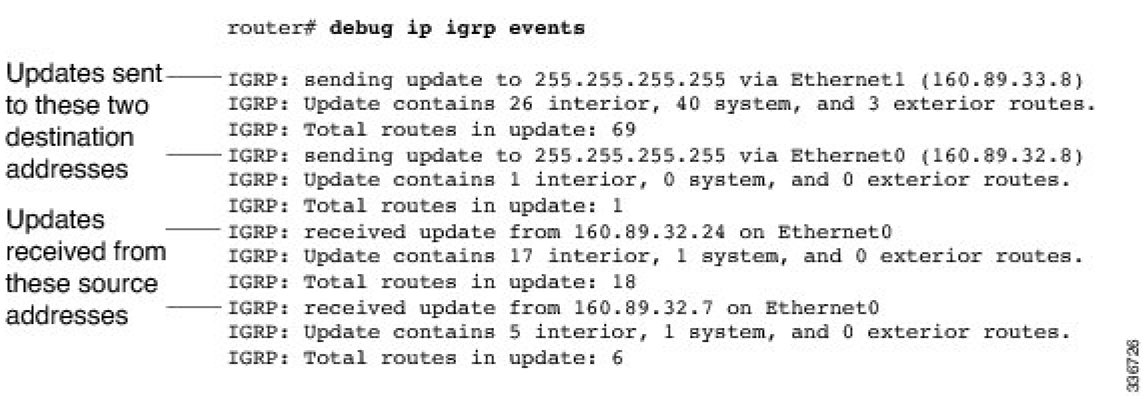
Usage Guidelines

If the IP address of an IGRP neighbor is specified, the resulting **debug ip igrp events** output includes messages describing updates from that neighbor and updates that the router broadcasts toward that neighbor. Messages are not generated for each route.

This command is particularly useful when there are many networks in your routing table. In this case, using **debug ip igrp transactions** could flood the console and make the router unusable. Use **debug ip igrp events** instead to display summary routing information.

Examples

The following is sample output from the **debug ip igrp events** command:



This shows that the router has sent two updates to the broadcast address 255.255.255.255. The router also received two updates. Three lines of output describe each of these updates.

The first line indicates whether the router sent or received the update packet, the source or destination address, and the interface through which the update was sent or received. If the update was sent, the IP address assigned to this interface is shown (in parentheses).

```
IGRP: sending update to 255.255.255.255 via Ethernet1 (160.89.33.8)
```

The second line summarizes the number and types of routes described in the update:

```
IGRP: Update contains 26 interior, 40 system, and 3 exterior routes.
```

The third line indicates the total number of routes described in the update:

```
IGRP: Total routes in update: 69
```

debug ip igrp transactions

To display transaction information on Interior Gateway Routing Protocol (IGRP) routing transactions, use the **debug ip igrp transactions** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip igrp transactions [*ip-address*]

no debug ip igrp transactions [*ip-address*]

Syntax Description

ip-address	(Optional) The IP address of an IGRP neighbor.

Command Modes

Privileged EXEC

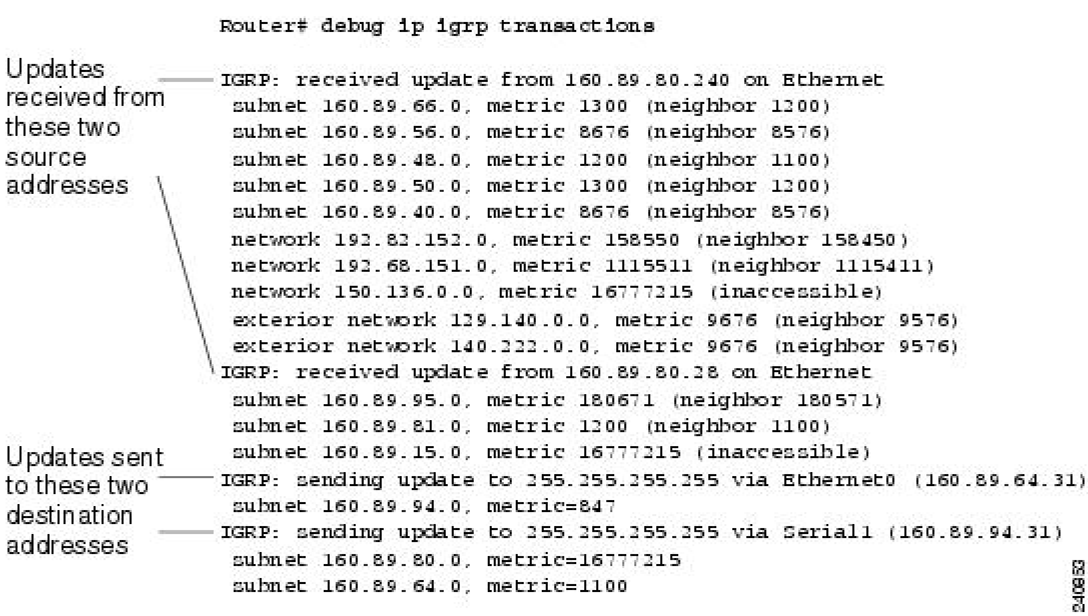
Usage Guidelines

If the IP address of an IGRP neighbor is specified, the resulting **debug ip igrp transactions** output includes messages describing updates from that neighbor and updates that the router broadcasts toward that neighbor.

When many networks are in your routing table, the **debug ip igrp transactions** command can flood the console and make the router unusable. In this case, use the **debug ip igrp events** command instead to display summary routing information.

Examples

The following is sample output from the **debug ip igrp transactions** command:



The output shows that the router being debugged has received updates from two other routers on the network. The router at source address 160.89.80.240 sent information about ten destinations in the update; the router at source address 160.89.80.28 sent information about three destinations in its update. The router being debugged also sent updates--in both cases to the broadcast address 255.255.255.255 as the destination address.

On the second line the first field refers to the type of destination information: “subnet” (interior), “network” (system), or “exterior” (exterior). The second field is the Internet address of the destination network. The third field is the metric stored in the routing table and the metric advertised by the neighbor sending the information. “Metric... inaccessible” usually means that the neighbor router has put the destination in a hold down state.

The entries show that the router is sending updates that are similar, except that the numbers in parentheses are the source addresses used in the IP header. A metric of 16777215 is inaccessible.

Other examples of output that the **debug ip igrp transactions** command can produce follow.

The following entry indicates that the routing table was updated and shows the new edition number (97 in this case) to be used in the next IGRP update:

```
IGRP: edition is now 97
```

Entries such as the following occur on startup or when some event occurs such as an interface making a transition or a user manually clearing the routing table:

```
IGRP: broadcasting request on Ethernet0
IGRP: broadcasting request on Ethernet1
```

The following type of entry can result when routing updates become corrupted between sending and receiving routers:

```
IGRP: bad checksum from 172.69.64.43
```

An entry such as the following should never appear. If it does, the receiving router has a bug in the software or a problem with the hardware. In either case, contact your technical support representative.

```
IGRP: system 45 from 172.69.64.234, should be system 109
```

debug ip inspect

Note Effective with Cisco IOS Release 12.4(20)T, the **debug ip inspect** command is replaced by the **debug policy-firewall** command. See the **debug policy-firewall** command for more information.

To display messages about Cisco IOS Firewall events, use the **debug ip inspect** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip inspect {**function-trace**| **object-creation**| **object-deletion**| **events**| **timers**| *protocol*| **detailed**| **update**}

Firewall MIB Statistics Syntax

debug ip inspect mib {**object-creation**| **object-deletion**| **events**| **retrieval**| **update**}

no debug ip inspect

Syntax Description

mib	(Optional) Displays messages about MIB functionality.
function-trace	Displays messages about software functions called by the Cisco IOS Firewall.
object-creation	Displays messages about software objects being created by the Cisco IOS Firewall. Object creation corresponds to the beginning of Cisco IOS Firewall-inspected sessions.
object-deletion	Displays messages about software objects being deleted by the Cisco IOS Firewall. Object deletion corresponds to the closing of Cisco IOS Firewall-inspected sessions.
events	Displays messages about Cisco IOS Firewall software events, including information about Cisco IOS Firewall packet processing or MIB special events.
timers	Displays messages about Cisco IOS Firewall timer events such as when the Cisco IOS Firewall idle timeout is reached.
protocol	Displays messages about Cisco IOS Firewall-inspected protocol events, including details about the packets of the protocol. The table below provides a list of *protocol* keywords.

detailed	Displays detailed information to be displayed for all the other enabled Cisco IOS Firewall debugging. Use this form of the command in conjunction with other Cisco IOS Firewall debug commands.
retrieval	Displays messages of statistics requested via Simple Network Management Protocol (SNMP) or command-line interface (CLI).
update	Displays messages about Cisco IOS Firewall software updates or updates to MIB counters.

Table 29: Protocol Keywords for the debug ip inspect Command

Application Protocol	Protocol Keyword
Transport-layer protocols	
ICMP	icmp
TCP	tcp
User Datagram Protocol (UDP)	udp
Application-layer protocols	
CU-SeeMe	cuseeme
FTP commands and responses	ftp-cmd
FTP tokens (enables tracing of the FTP tokens parsed)	ftp-tokens
H.323 (version 1 and version 2)	h323
HTTP	http
IMAP	imap
Microsoft NetShow	netshow
POP3	pop3
RealAudio	realaudio
Remote procedure call (RPC)	rpc
Real Time Streaming Protocol (RTSP)	rtsp
Session Initiation Protocol (SIP)	sip

Application Protocol	Protocol Keyword
Simple Mail Transfer Protocol (SMTP)	smtp
Skinny Client Control Protocol (SCCP)	skinny
Structured Query Language*Net (SQL*Net)	sqlnet
StreamWorks	streamworks
TFTP	tftp
UNIX r-commands (rlogin, rexec, rsh)	rcmd
VDOLive	vdolive

Command Modes

Privileged EXEC

Command History

Release	Modification
11.2 P	This command was introduced.
12.0(5)T	NetShow support was added.
12.0(7)T	H.323 V2 and RTSP protocol support were added.
12.2(11)YU	Support for the ICMP and SIP protocols was added.
12.2(15)T	This command was integrated into Cisco IOS Release 12.2(15)T.
12.3(1)	Support for the skinny protocol was added.
12.3(14)T	Support for the IMAP and POP3 protocols was added.
12.4(6)T	The MIB syntax was added.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.4(20)T	This command was replaced by the **debug policy-firewall**command.

Examples

The following is sample output from the **debug ip inspect function-trace** command:

```
Router# debug ip inspect function-trace

*Mar  2 01:16:16: CBAC FUNC: insp_inspection
*Mar  2 01:16:16: CBAC FUNC: insp_pre_process_sync
*Mar  2 01:16:16: CBAC FUNC: insp_find_tcp_host_entry addr 40.0.0.1 bucket 41
```

```
*Mar  2 01:16:16: CBAC FUNC: insp_find_pregen_session
*Mar  2 01:16:16: CBAC FUNC: insp_get_idbsb
*Mar  2 01:16:16: CBAC FUNC: insp_get_idbsb
*Mar  2 01:16:16: CBAC FUNC: insp_get_irc_of_idb
*Mar  2 01:16:16: CBAC FUNC: insp_get_idbsb
*Mar  2 01:16:16: CBAC FUNC: insp_create_sis
*Mar  2 01:16:16: CBAC FUNC: insp_inc_halfopen_sis
*Mar  2 01:16:16: CBAC FUNC: insp_link_session_to_hash_table
*Mar  2 01:16:16: CBAC FUNC: insp_inspect_pak
*Mar  2 01:16:16: CBAC FUNC: insp_l4_inspection
*Mar  2 01:16:16: CBAC FUNC: insp_process_tcp_seg
*Mar  2 01:16:16: CBAC FUNC: insp_listen_state
*Mar  2 01:16:16: CBAC FUNC: insp_ensure_return_traffic
*Mar  2 01:16:16: CBAC FUNC: insp_add_acl_item
*Mar  2 01:16:16: CBAC FUNC: insp_ensure_return_traffic
*Mar  2 01:16:16: CBAC FUNC: insp_add_acl_item
*Mar  2 01:16:16: CBAC FUNC: insp_process_syn_packet
*Mar  2 01:16:16: CBAC FUNC: insp_find_tcp_host_entry addr 40.0.0.1 bucket 41
*Mar  2 01:16:16: CBAC FUNC: insp_create_tcp_host_entry
*Mar  2 01:16:16: CBAC* FUNC: insp_fast_inspection
*Mar  2 01:16:16: CBAC* FUNC: insp_inspect_pak
*Mar  2 01:16:16: CBAC* FUNC: insp_l4_inspection
*Mar  2 01:16:16: CBAC* FUNC: insp_process_tcp_seg
*Mar  2 01:16:16: CBAC* FUNC: insp_synrcvd_state
*Mar  2 01:16:16: CBAC* FUNC: insp_fast_inspection
*Mar  2 01:16:16: CBAC* FUNC: insp_inspect_pak
*Mar  2 01:16:16: CBAC* FUNC: insp_l4_inspection
*Mar  2 01:16:16: CBAC* FUNC: insp_process_tcp_seg
*Mar  2 01:16:16: CBAC* FUNC: insp_synrcvd_state
*Mar  2 01:16:16: CBAC FUNC: insp_dec_halfopen_sis
*Mar  2 01:16:16: CBAC FUNC: insp_remove_sis_from_host_entry
*Mar  2 01:16:16: CBAC FUNC: insp_find_tcp_host_entry addr 40.0.0.1 bucket 41
```

This output shows the functions called by the Cisco IOS Firewall as a session is inspected. Entries with an asterisk (*) after the word "CBAC" are entries when the fast path is used; otherwise, the process path is used.

The following is sample output from the **debug ip inspect object-creation** and **debug ip inspect object-deletion** commands:

```
Router# debug ip inspect object-creation
Router# debug ip inspect object-deletion
*Mar  2 01:18:30: CBAC OBJ_CREATE: create pre-gen sis 25A3574
*Mar  2 01:18:30: CBAC OBJ_CREATE: create acl wrapper 25A36FC -- acl item 25A3634
*Mar  2 01:18:30: CBAC OBJ_CREATE: create sis 25C1CC4
*Mar  2 01:18:30: CBAC OBJ_DELETE: delete pre-gen sis 25A3574
*Mar  2 01:18:30: CBAC OBJ_CREATE: create host entry 25A3574 addr 10.0.0.1 bucket 31
*Mar  2 01:18:30: CBAC OBJ_DELETE: delete sis 25C1CC4
*Mar  2 01:18:30: CBAC OBJ_DELETE: delete create acl wrapper 25A36FC -- acl item 25A3634
*Mar  2 01:18:31: CBAC OBJ_DELETE: delete host entry 25A3574 addr 10.0.0.1
```

The following is sample output from the **debug ip inspect object-creation**, **debug ip inspect object-deletion**, and **debug ip inspect events** commands:

```
Router# debug ip inspect object-creation
Router# debug ip inspect object-deletion
Router# debug ip inspect events
*Mar  2 01:18:51: CBAC OBJ_CREATE: create pre-gen sis 25A3574
*Mar  2 01:18:51: CBAC OBJ_CREATE: create acl wrapper 25A36FC -- acl item 25A3634
*Mar  2 01:18:51: CBAC Src 10.1.0.1 Port [1:65535]
*Mar  2 01:18:51: CBAC Dst 10.0.0.1 Port [46406:46406]
*Mar  2 01:18:51: CBAC Pre-gen sis 25A3574 created: 10.1.0.1[1:65535] 30.0.0.1[46406:46406]
*Mar  2 01:18:51: CBAC OBJ_CREATE: create sis 25C1CC4
*Mar  2 01:18:51: CBAC sis 25C1CC4 initiator_addr (10.1.0.1:20) responder_addr
(30.0.0.1:46406) initiator_alt_addr (40.0.0.1:20) responder_alt_addr (10.0.0.1:46406)
*Mar  2 01:18:51: CBAC OBJ_DELETE: delete pre-gen sis 25A3574
*Mar  2 01:18:51: CBAC OBJ_CREATE: create host entry 25A3574 addr 10.0.0.1 bucket 31
*Mar  2 01:18:51: CBAC OBJ_DELETE: delete sis 25C1CC4
*Mar  2 01:18:51: CBAC OBJ_DELETE: delete create acl wrapper 25A36FC -- acl item 25A3634
*Mar  2 01:18:51: CBAC OBJ_DELETE: delete host entry 25A3574 addr 10.0.0.1
```

The following is sample output from the **debug ip inspect timers** command:

```
Router# debug ip inspect timers
*Mar  2 01:19:15: CBAC Timer Init Leaf: Pre-gen sis 25A3574
*Mar  2 01:19:15: CBAC Timer Start: Pre-gen sis 25A3574 Timer: 25A35D8 Time: 30000 milisecs
*Mar  2 01:19:15: CBAC Timer Init Leaf: sis 25C1CC4
*Mar  2 01:19:15: CBAC Timer Stop: Pre-gen sis 25A3574 Timer: 25A35D8
*Mar  2 01:19:15: CBAC Timer Start: sis 25C1CC4 Timer: 25C1D5C Time: 30000 milisecs
*Mar  2 01:19:15: CBAC Timer Start: sis 25C1CC4 Timer: 25C1D5C Time: 3600000 milisecs
*Mar  2 01:19:15: CBAC Timer Start: sis 25C1CC4 Timer: 25C1D5C Time: 5000 milisecs
*Mar  2 01:19:15: CBAC Timer Stop: sis 25C1CC4 Timer: 25C1D5C
```

The following is sample output from the **debug ip inspect tcp** command:

```
Router# debug ip inspect tcp
*Mar  2 01:20:43: CBAC* sis 25A3604 pak 2541C58 TCP P ack 4223720032 seq 4200176225(22)
(10.0.0.1:46409) => (10.1.0.1:21)
*Mar  2 01:20:43: CBAC* sis 25A3604 ftp L7 inspect result: PROCESS-SWITCH packet
*Mar  2 01:20:43: CBAC sis 25A3604 pak 2541C58 TCP P ack 4223720032 seq 4200176225(22)
(10.0.0.1:46409) => (10.1.0.1:21)
*Mar  2 01:20:43: CBAC sis 25A3604 ftp L7 inspect result: PASS packet
*Mar  2 01:20:43: CBAC* sis 25A3604 pak 2544374 TCP P ack 4200176247 seq 4223720032(30)
(10.0.0. 1:46409) <= (10.1.0.1:21)
*Mar  2 01:20:43: CBAC* sis 25A3604 ftp L7 inspect result: PASS packet
*Mar  2 01:20:43: CBAC* sis 25A3604 pak 25412F8 TCP P ack 4223720062 seq 4200176247(15)
(10.0.0. 1:46409) => (10.1.0.1:21)
*Mar  2 01:20:43: CBAC* sis 25A3604 ftp L7 inspect result: PASS packet
*Mar  2 01:20:43: CBAC sis 25C1CC4 pak 2544734 TCP S seq 4226992037(0) (10.1.0.1:20) =>
(10.0.0.1:46411)
*Mar  2 01:20:43: CBAC* sis 25C1CC4 pak 2541E38 TCP S ack 4226992038 seq 4203405054(0)
(10.1.0.1:20) <= (10.0.0.1:46411)
```

This sample shows TCP packets being processed and lists the corresponding acknowledge (ACK) packet numbers and sequence (SEQ) numbers. The number of data bytes in the TCP packet is shown in parentheses--for example, (22). For each packet shown, the addresses and port numbers are shown separated by a colon. For example, (10.1.0.1:21) indicates an IP address of 10.1.0.1 and a TCP port number of 21.

Entries with an asterisk (*) after the word "CBAC" are entries when the fast path is used; otherwise, the process path is used.

The following is sample output from the **debug ip inspect tcp** and **debug ip inspect detailed** commands:

```
Router# debug ip inspect tcp
Router# debug ip inspect detailed
*Mar  2 01:20:58: CBAC* Pak 2541E38 Find session for (30.0.0.1:46409) (40.0.0.1:21) tcp
*Mar  2 01:20:58:  P ack 4223720160 seq 4200176262(22)
*Mar  2 01:20:58: CBAC* Pak 2541E38 Addr:port pairs to match: (30.0.0.1:46409) (40.0.0.1:21)
*Mar  2 01:20:58: CBAC* sis 25A3604 SIS_OPEN
*Mar  2 01:20:58: CBAC* Pak 2541E38 IP: s=30.0.0.1 (Ethernet0), d=40.0.0.1 (Ethernet1), len
 76,proto=6
*Mar  2 01:20:58: CBAC sis 25A3604 Saving State: SIS_OPEN/ESTAB iisn 4200176160 i_rcvnxt
4223720160 i_sndnxt 4200176262 i_rcvwnd 8760 risn 4223719771 r_rcvnxt 4200176262 r_sndnxt
4223720160 r_rcvwnd 8760
*Mar  2 01:20:58: CBAC* sis 25A3604 pak 2541E38 TCP P ack 4223720160 seq 4200176262(22)
(30.0.0.1:46409) => (40.0.0.1:21)
*Mar  2 01:20:58: CBAC* sis 25A3604 pak 2541E38 SIS_OPEN/ESTAB TCP seq 4200176262(22) Flags:
 ACK 4223720160 PSH
*Mar  2 01:20:58: CBAC* sis 25A3604 pak 2541E38 --> SIS_OPEN/ESTAB iisn 4200176160 i_rcvnxt
 4223720160 i_sndnxt 4200176284 i_rcvwnd 8760 risn 4223719771 r_rcvnxt 4200176262 r_sndnxt
 4223720160 r_rcvwnd 8760
*Mar  2 01:20:58: CBAC* sis 25A3604 L4 inspect result: PASS packet 2541E38 (30.0.0.1:46409)
 (40.0.0.1:21) bytes 22 ftp
*Mar  2 01:20:58: CBAC sis 25A3604 Restoring State: SIS_OPEN/ESTAB iisn 4200176160 i_rcvnxt
 4223
720160 i_sndnxt 4200176262 i_rcvwnd 8760 risn 4223719771 r_rcvnxt 4200176262 r_sndnxt
4223720160 r_rcvwnd 8760
*Mar  2 01:20:58: CBAC* sis 25A3604 ftp L7 inspect result: PROCESS-SWITCH packet
*Mar  2 01:20:58: CBAC* sis 25A3604 ftp L7 inspect result: PROCESS-SWITCH packet
*Mar  2 01:20:58: CBAC* Bump up: inspection requires the packet in the process path(30.0.0.1)
 (40.0.0.1)
```

```
*Mar  2 01:20:58: CBAC Pak 2541E38 Find session for (30.0.0.1:46409) (40.0.0.1:21) tcp
*Mar  2 01:20:58:  P ack 4223720160 seq 4200176262(22)
*Mar  2 01:20:58: CBAC Pak 2541E38 Addr:port pairs to match: (30.0.0.1:46409) (40.0.0.1:21)
*Mar  2 01:20:58: CBAC sis 25A3604 SIS_OPEN
*Mar  2 01:20:58: CBAC Pak 2541E38 IP: s=30.0.0.1 (Ethernet0), d=40.0.0.1 (Ethernet1), len
 76, proto=6
```

The following is sample output from the **debug ip inspect icmp** and **debug ip inspect detailed** commands:

```
Router# debug ip inspect icmp
Router# debug ip inspect detailed
1w6d:CBAC sis 81073F0C SIS_CLOSED
1w6d:CBAC Pak 80D2E9EC IP:s=192.168.133.3 (Ethernet1), d=0.0.0.0 (Ethernet0), len 98, proto=1
1w6d:CBAC ICMP:sis 81073F0C pak 80D2E9EC SIS_CLOSED ICMP packet (192.168.133.3:0) =>
(0.0.0.0:0) datalen 56
1w6d:CBAC ICMP:start session from 192.168.133.3
1w6d:CBAC sis 81073F0C --> SIS_OPENING (192.168.133.3:0) (0.0.0.0:0)
1w6d:CBAC sis 81073F0C L4 inspect result:PASS packet 80D2E9EC (192.168.133.3:0) (0.0.0.0:0)
 bytes 56 icmp
1w6d:CBAC sis 81073F0C SIS_OPENING
1w6d:CBAC Pak 80E72BFC IP:s=0.0.0.0 (Ethernet0), d=192.168.133.3 (Ethernet1), len 98, proto=1
1w6d:CBAC ICMP:sis 81073F0C pak 80E72BFC SIS_OPENING ICMP packet (192.168.133.3:0) <=
(0.0.0.0:0) datalen 56
1w6d:CBAC sis 81073F0C --> SIS_OPEN (192.168.133.3:0) (0.0.0.0:0)
1w6d:CBAC sis 81073F0C L4 inspect result:PASS packet 80E72BFC (0.0.0.0:0) (192.168.133.3:0)
 bytes 56 icmp
1w6d:CBAC* sis 81073F0C SIS_OPEN
1w6d:CBAC* Pak 80D2F2C8 IP:s=192.168.133.3 (Ethernet1), d=0.0.0.0 (Ethernet0), len 98,
proto=1
1w6d:CBAC* ICMP:sis 81073F0C pak 80D2F2C8 SIS_OPEN ICMP packet (192.168.133.3:0) =>
(0.0.0.0:0) datalen 56
1w6d:CBAC* sis 81073F0C --> SIS_OPEN (192.168.133.3:0) (0.0.0.0:0)
1w6d:CBAC* sis 81073F0C L4 inspect result:PASS packet 80D2F2C8 (192.168.133.3:0) (0.0.0.0:0)
 bytes 56 icmp
1w6d:CBAC* sis 81073F0C SIS_OPEN
1w6d:CBAC* Pak 80E737CC IP:s=0.0.0.0 (Ethernet0), d=192.168.133.3 (Ethernet1), len 98,
proto=1
1w6d:CBAC* ICMP:sis 81073F0C pak 80E737CC SIS_OPEN ICMP packet (192.168.133.3:0) <=
(0.0.0.0:0) datalen 56
1w6d:CBAC* sis 81073F0C --> SIS_OPEN (192.168.133.3:0) (0.0.0.0:0)
1w6d:CBAC* sis 81073F0C L4 inspect result:PASS packet 80E737CC (0.0.0.0:0) (192.168.133.3:0)
 bytes 56 icmp
1w6d:CBAC* sis 81073F0C SIS_OPEN
1w6d:CBAC* Pak 80F554F0 IP:s=192.168.133.3 (Ethernet1), d=0.0.0.0 (Ethernet0), len 98,
proto=1
1w6d:CBAC* ICMP:sis 81073F0C pak 80F554F0 SIS_OPEN ICMP packet (192.168.133.3:0) =>
(0.0.0.0:0) datalen 56
1w6d:CBAC* sis 81073F0C --> SIS_OPEN (192.168.133.3:0) (0.0.0.0:0)
1w6d:CBAC* sis 81073F0C L4 inspect result:PASS packet 80F554F0 (192.168.133.3:0) (0.0.0.0:0)
 bytes 56 icmp
1w6d:CBAC* sis 81073F0C SIS_OPEN
1w6d:CBAC* Pak 80E73AC0 IP:s=0.0.0.0 (Ethernet0), d=192.168.133.3 (Ethernet1), len 98,
proto=1
1w6d:CBAC* ICMP:sis 81073F0C pak 80E73AC0 SIS_OPEN ICMP packet (192.168.133.3:0) <=
(0.0.0.0:0) datalen 56
1w6d:CBAC* sis 81073F0C --> SIS_OPEN (192.168.133.3:0) (0.0.0.0:0)
1w6d:CBAC* sis 81073F0C L4 inspect result:PASS packet 80E73AC0 (0.0.0.0:0) (192.168.133.3:0)
 bytes 56 icmp
```

debug ip inspect ha

To display messages about Cisco IOS stateful failover high availability (HA) events, use the **debug ip inspect ha**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip inspect ha [**manager**| **packet**| **update**]

no debug ip inspect ha [**manager**| **packet**| **update**]

Syntax Description

manager	(Optional) Displays detailed messages for interaction of firewall HA manager with the box-to-box high availability infrastructure.
packet	(Optional) Used to debug the processing of the first packet postfailover on the new active device.
update	(Optional) Used to debug the periodic update messages between the active and standby. The Firewall HA sends periodical messages to update the standby of the firewall sessions state on the active.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.4(6)T	This command was introduced.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.

Examples

The following is sample output from the **debug ip inspect ha**command. This example shows an add session message and a delete session message received by the the active and standby devices:

```
Router# debug ip inspect ha
Active debugs -
*Apr 13 17:15:20.795:  FW-HA:Send add session msg for session 2C6B820
*Apr 13 17:15:36.919:  FW-HA:Send delete session msg for session 2C6B820
Standby debugs -
*Apr 13 17:19:00.471:  FW-HA:Received add session message
(10.0.0.10:56733:0)=>(11.0.0.10:23:0)
*Apr 13 17:19:12.051:  FW-HA:Received delete session message
(10.0.0.10:56733:0)=>(11.0.0.10:23:0)
```

The following is sample output from the **debug ip inspect ha manager**command. Using the **manager** keyword provides a more detailed debug analysis:

```
Router# debug ip inspect ha manager
*Apr 13 17:23:28.995: HA Message 0:flags=0x01 len=727 FW_HA_MSG_INSERT_SESSION (1)
*Apr 13 17:23:28.995:   ID: grp1
*Apr 13 17:23:28.995:   attr FW_HA_ATT_INITIATOR_ADDR (1) len 4
*Apr 13 17:23:28.995:    0A 00 00 0A
*Apr 13 17:23:28.995:   attr FW_HA_ATT_RESPONDER_ADDR (2) len 4
*Apr 13 17:23:28.995:    0B 00 00 0A
*Apr 13 17:23:28.995:   attr FW_HA_ATT_INITIATOR_PORT (3) len 2
*Apr 13 17:23:28.995:    BF 1C
*Apr 13 17:23:28.995:   attr FW_HA_ATT_RESPONDER_PORT (4) len 2
*Apr 13 17:23:28.995:    00 17
*Apr 13 17:23:28.995:   attr FW_HA_ATT_L4_PROTOCOL (5) len 4
*Apr 13 17:23:28.995:    00 00 00 01
*Apr 13 17:23:28.995:   attr FW_HA_ATT_SRC_TABLEID (6) len 1
*Apr 13 17:23:28.995:    00
*Apr 13 17:23:28.995:   attr FW_HA_ATT_DST_TABLEID (7) len 1
*Apr 13 17:23:28.995:    00
*Apr 13 17:23:28.995:   attr FW_HA_ATT_R_RCVNXT (20) len 4
*Apr 13 17:23:28.995:    79 EA E2 9A
*Apr 13 17:23:28.995:   attr FW_HA_ATT_R_SNDNXT (21) len 4
*Apr 13 17:23:28.995:    6C 7D E4 04
*Apr 13 17:23:28.995:   attr FW_HA_ATT_R_RCVWND (22) len 4
*Apr 13 17:23:28.995:    00 00 10 20
*Apr 13 17:23:28.995:   attr FW_HA_ATT_R_LAST_SEQ_TO_SND (23) len 4
*Apr 13 17:23:28.995:    00 00 00 00
*Apr 13 17:23:28.995:   attr FW_HA_ATT_I_RCVNXT (24) len 4
*Apr 13 17:23:28.995:    6C 7D E4 04
*Apr 13 17:23:28.995:   attr FW_HA_ATT_I_SNDNXT (25) len 4
*Apr 13 17:23:28.995:    79 EA E2 9A
*Apr 13 17:23:28.995:   attr FW_HA_ATT_I_RCVWND (26) len 4
*Apr 13 17:23:28.995:    00 00 10 20
*Apr 13 17:23:28.995:   attr FW_HA_ATT_I_LAST_SEQ_TO_SND (27) len 4
*Apr 13 17:23:28.995:    00 00 00 00
*Apr 13 17:23:28.995:   attr FW_HA_ATT_TCP_STATE (28) len 4
*Apr 13 17:23:28.995:    00 00 00 04
*Apr 13 17:23:28.995:   attr FW_HA_ATT_INITIATOR_ALT_ADDR (8) len 4
*Apr 13 17:23:28.995:    0A 00 00 0A
*Apr 13 17:23:28.995:   attr FW_HA_ATT_RESPONDER_ALT_ADDR (9) len 4
*Apr 13 17:23:28.995:    0B 00 00 0A
*Apr 13 17:23:28.995:   attr FW_HA_ATT_INITIATOR_ALT_PORT (10) len 2
*Apr 13 17:23:28.995:    BF 1C
*Apr 13 17:23:28.995:   attr FW_HA_ATT_RESPONDER_ALT_PORT (11) len 2
*Apr 13 17:23:28.995:    00 00
*Apr 13 17:23:28.995:   attr FW_HA_ATT_L7_PROTOCOL (12) len 4
*Apr 13 17:23:28.995:    00 00 00 05
*Apr 13 17:23:28.995:   attr FW_HA_ATT_INSP_DIR (13) len 4
*Apr 13 17:23:28.995:    00 00 00 01
*Apr 13 17:23:28.995:   attr FW_HA_ATT_I_ISN (29) len 4
*Apr 13 17:23:28.995:    79 EA E2 99
*Apr 13 17:23:28.995:   attr FW_HA_ATT_R_ISN (30) len 4
*Apr 13 17:23:28.995:    6C 7D E4 03
*Apr 13 17:23:28.995:   attr FW_HA_ATT_APPL_INSP_FLAGS (15) len 2
*Apr 13 17:23:28.995:    00 10
*Apr 13 17:23:28.995:   attr FW_HA_ATT_TERM_FLAGS (16) len 1
*Apr 13 17:23:28.995:    00
*Apr 13 17:23:28.995:   attr FW_HA_ATT_IS_LOCAL_TRAFFIC (17) len 1
*Apr 13 17:23:28.995:    00
*Apr 13 17:23:28.995:   attr FW_HA_ATT_DATA_DIR (18) len 4
*Apr 13 17:23:28.995:    00 00 00 00
*Apr 13 17:23:28.995:   attr FW_HA_ATT_SESSION_LIMITING_DONE (19) len 1
*Apr 13 17:23:28.995:    00
*Apr 13 17:23:28.995:   attr FW_HA_ATT_INSPECT_RULE (14) len 256
*Apr 13 17:23:28.995:    74 65 73 74 00 00 00 00
```

debug ip inspect L2-transparent

To enable debugging messages for transparent firewall events, use the **debug ip inspect L2-transparent** command in privileged EXEC mode. To disable debugging messages, use the **no** form of this command.

debug ip inspect L2-transparent {**packet**| **dhcp-passthrough**}

no debug ip inspect L2-transparent {**packet**| **dhcp-passthrough**}

Syntax Description

packet	Displays messages for all debug packets that are inspected by the transparent firewall. **Note** Only IP packets (TCP, User Datagram Protocol [UDP], and Internet Control Management Protocol [ICMP]) are subjected to inspection by the transparent firewall.
dhcp-passthrough	Displays debug messages only for DHCP pass-through traffic that the transparent firewall forwards across the bridge. To allow a transparent firewall to forward DHCP pass-through traffic, use the **ip inspect L2-transparent dhcp-passthrough** command.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(7)T	This command was introduced.

Usage Guidelines

The **debug ip inspect L2-transparent**command can be used to help verify and troubleshoot transparent firewall-related configurations, such as a Telnet connection from the client to the server with inspection configured.

Examples

The following example shows how the transparent firewall debug command works in a basic transparent firewall configuration. (Note that each debug message is preceded by an asterisk (*).)

```
! Enable debug commands.
Router# debug ip inspect L2-transparent packet
INSPECT L2 firewall debugging is on
Router# debug ip inspect object-creation
INSPECT Object Creations debugging is on
Router# debug ip inspect object-deletion
INSPECT Object Deletions debugging is on
```

```
! Start the transparent firewall configuration process
Router# config terminal
Enter configuration commands, one per line.  End with CNTL/Z.
! Configure bridging
Router(config)# bridge 1 protocol ieee
Router(config)# bridge irb
Router(config)# bridge 1 route ip
Router(config)# interface bvi1
*Mar  1 00:06:42.511:%LINK-3-UPDOWN:Interface BVI1, changed state to down.
Router(config-if)# ip address 209.165.200.225 255.255.255.254
! Configure inspection
Router(config)# ip inspect name test tcp
! Following debugs show the memory allocated for CBAC rules.
*Mar  1 00:07:21.127:CBAC OBJ_CREATE:create irc 817F04F0 (test)
*Mar  1 00:07:21.127:CBAC OBJ_CREATE:create irt 818AED20 Protocol:tcp Inactivity time:0
test
Router(config)# ip inspect name test icmp
Router(config)#
*Mar  1 00:07:39.211:CBAC OBJ_CREATE:create irt 818AEDCC Protocol:icmp Inactivity time:0
! Configure Bridging on ethernet0 interface
Router(config)# interface ethernet0
Router(config-if)# bridge-group 1
*Mar  1 00:07:49.071:%LINK-3-UPDOWN:Interface BVI1, changed state to up
*Mar  1 00:07:50.071:%LINEPROTO-5-UPDOWN:Line protocol on Interface BVI1, changed state to
 up
! Configure inspection on ethernet0 interface
Router(config-if)# ip inspect test in
Router(config-if)#
*Mar  1 00:07:57.543:CBAC OBJ_CREATE:create idbsb 8189CBFC (Ethernet0)
! Incremented the number of bridging interfaces configured for inspection */
*Mar  1 00:07:57.543:L2FW:Incrementing L2FW i/f count
Router(config-if)# interface ethernet1
! Configure bridging and ACL on interface ethernet1
Router(config-if)# bridge-group 1
Router(config-if)# ip access-group 101 in
*Mar  1 00:08:26.711:%LINEPROTO-5-UPDOWN:Line protocol on Interface Ethernet1, changed state
 to up
Router(config-if)# end
```

Related Commands

Command	Description
ip inspect L2-transparent dhcp-passthrough	Allows a transparent firewall to forward DHCP pass-through traffic.

debug ip ips

To enable debugging messages for Cisco IOS Intrusion Prevention System (IPS), use the **debug ip ips** command in privileged EXEC mode. To disable debugging messages, use the **no** form of this command.

debug ip ips [*engine*] **[detailed] [service-msrpc] [service-sm]**

no debug ip ips [*engine*] **[detailed]**

Syntax Description

engine	(Optional) Displays debugging messages only for a specific signature engine.
detailed	(Optional) Displays detailed debugging messages for the specified signature engine or for all IPS actions.
service-msrpc	(Optional) Displays debugging messages for Microsoft RPC (Remote Procedure Call) (MSRPC) actions.
service-sm	(Optional) Displays debugging messages for Microsoft SMB(Server Message Block) actions.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(8)T	This command was introduced.
12.4(15)T	The **service-msrpc** and the **service-sm** keywords were added to support Microsoft communication protocols MSRPC and SMB.

Examples

The following example shows how to enable debugging messages for the Cisco IOS IPS:

```
Router# debug ip ips
```

debug ip mbgp dampening

To log route flap dampening activity related to multiprotocol Border Gateway Protocol (BGP), use the **debug ip mbgp dampening**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip mbgp dampening [*access-list-number*]

no debug ip mbgp dampening [*access-list-number*]

Syntax Description

access-list-number	(Optional) The number of an access list in the range from 1 to 99. If an access list number is specified, debugging occurs only for the routes permitted by the access list.

Command Default

Logging for route flap dampening activity is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
11.1(20)CC	This command was introduced.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Examples

The following is sample output from the **debug ip mbgp dampening** command:

```
Router# debug ip mbgp dampening
BGP: charge penalty for 173.19.0.0/16 path 49 with halflife-time 15 reuse/suppress 750/2000
BGP: flapped 1 times since 00:00:00. New penalty is 1000
BGP: charge penalty for 173.19.0.0/16 path 19 49 with halflife-time 15 reuse/suppress
750/2000
BGP: flapped 1 times since 00:00:00. New penalty is 1000
```

debug ip mbgp updates

To log multiprotocol Border Gateway Protocol (BGP)-related information passed in BGP update messages, use the **debug ip mbgp updates**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip mbgp updates

no debug ip mbgp updates

Syntax Description

This command has no arguments or keywords.

Command Default

Logging for multiprotocol BGP-related information in BGP update messages is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
11.1(20)CC	This command was introduced.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Examples

The following is sample output from the **debug ip mbgp updates** command:

```
Router# debug ip mbgp updates
BGP: NEXT_HOP part 1 net 200.10.200.0/24, neigh 171.69.233.49, next 171.69.233.34
BGP: 171.69.233.49 send UPDATE 200.10.200.0/24, next 171.69.233.34, metric 0, path 33 34
19 49 109 65000 297 3561 6503
BGP: NEXT_HOP part 1 net 200.10.202.0/24, neigh 171.69.233.49, next 171.69.233.34
BGP: 171.69.233.49 send UPDATE 200.10.202.0/24, next 171.69.233.34, metric 0, path 33 34
19 49 109 65000 297 1239 1800 3597
BGP: NEXT_HOP part 1 net 200.10.228.0/22, neigh 171.69.233.49, next 171.69.233.34
BGP: 171.69.233.49 rcv UPDATE about 222.2.2.0/24, next hop 171.69.233.49, path 49 109 metric
 0
BGP: 171.69.233.49 rcv UPDATE about 131.103.0.0/16, next hop 171.69.233.49, path 49 109
metric 0
BGP: 171.69.233.49 rcv UPDATE about 206.205.242.0/24, next hop 171.69.233.49, path 49 109
metric 0
BGP: 171.69.233.49 rcv UPDATE about 1.0.0.0/8, next hop 171.69.233.49, path 49 19 metric 0
BGP: 171.69.233.49 rcv UPDATE about 198.1.2.0/24, next hop 171.69.233.49, path 49 19 metric
 0
BGP: 171.69.233.49 rcv UPDATE about 171.69.0.0/16, next hop 171.69.233.49, path 49 metric
0
BGP: 171.69.233.49 rcv UPDATE about 172.19.0.0/16, next hop 171.69.233.49, path 49 metric
0
BGP: nettable_walker 172.19.0.0/255.255.0.0 calling revise_route
BGP: revise route installing 172.19.0.0/255.255.0.0 -> 171.69.233.49
BGP: 171.69.233.19 computing updates, neighbor version 267099, table version 267100, starting
 at 0.0.0.0
BGP: NEXT_HOP part 1 net 172.19.0.0/16, neigh 171.69.233.19, next 171.69.233.49
BGP: 171.69.233.19 send UPDATE 172.19.0.0/16, next 171.69.233.49, metric 0, path 33 49
BGP: 1 updates (average = 46, maximum = 46)
```

```
BGP: 171.69.233.19 updates replicated for neighbors : 171.69.233.34, 171.69.233.49,
171.69.233.56
BGP: 171.69.233.19 1 updates enqueued (average=46, maximum=46)
BGP: 171.69.233.19 update run completed, ran for 0ms, neighbor version 267099, start version
 267100, throttled to 267100, check point net 0.0.0.0
```

debug ip mcache

Note Effective with Cisco IOS Release 15.0(1)M and Cisco IOS Release 12.2(33)SRE, the **debug ip mcache**command is not available in Cisco IOS software.

To display IP multicast fast-switching events, use the **debug ip mcache**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip mcache [**vrf** *vrf-name*] [*hostname*| *group-address*]

no debug ip mcache [**vrf** *vrf-name*] [*hostname*| *group-address*]

Syntax Description

vrf	(Optional) Supports the Multicast Virtual Private Network (VPN) routing and forwarding (VRF) instance.
vrf-name	(Optional) Name assigned to the VRF.
hostname	(Optional) The host name.
group-address	(Optional) The group address.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
11.0	This command was introduced.
12.0(23)S	The **vrf** keyword and *vrf-name* argument were added.
12.2(13)T	This command was integrated into Cisco IOS Release 12.2(13)T.
12.2(14)S	This command was integrated into Cisco IOS Release 12.2(14)S.
12.2(27)SBC	This command was integrated into Cisco IOS Release 12.2(27)SBC.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
15.0(1)M	This command was removed.
12.2(33)SRE	This command was removed.

Usage Guidelines

Use this command when multicast fast switching appears not to be functioning.

Examples

The following is sample output from the **debug ip mcache** command when an IP multicast route is cleared:

```
Router# debug ip mcache
IP multicast fast-switching debugging is on

Router# clear ip mroute *
MRC: Build MAC header for (172.31.60.185/32, 224.2.231.173), Ethernet0
MRC: Fast-switch flag for (172.31.60.185/32, 224.2.231.173), off -> on, caller
ip_mroute_replicate-1
MRC: Build MAC header for (172.31.191.10/32, 224.2.127.255), Ethernet0
MRC: Build MAC header for (172.31.60.152/32, 224.2.231.173), Ethernet0
```

The table below describes the significant fields shown in the display.

Table 30: debug ip mcache Field Descriptions

Field	Description
MRC	Multicast route cache.
Fast-switch flag	Route is fast switched.
(172.31.60.185/32)	Host route with 32 bits of mask.
off -> on	State has changed.
caller ...	The code function that activated the state change.

Related Commands

Command	Description
debug ip dvmrp	Displays information on DVMRP packets received and sent.
debug ip igmp	Displays IGMP packets received and sent, and IGMP-host related events.
debug ip igrp transactions	Displays transaction information on IGRP routing transactions.
debug ip mrm	Displays MRM control packet activity.
debug ip sd	Displays all SD announcements received.

debug ip mds ipc

To debug multicast distributed switching (MDS) interprocessor communication, that is, synchronization between the Multicast Forwarding Information Base (MFIB) on the line card and the multicast routing table in the Route Processor (RP), use the **debug ip mds ipc**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip mds ipc {**event**| **packet**}

no debug ip mds ipc {**event**| **packet**}

Syntax Description

event	Displays MDS events when there is a problem.
packet	Displays MDS packets.

Command Modes

Privileged EXEC

Usage Guidelines

Use this command on the line card or RP.

Examples

The following is sample output from the **debug ip mds ipc packet** command:

```
Router# debug ip mds ipc packet
MDFS ipc packet debugging is on
Router#
MDFS: LC sending statistics message to RP with code 0 of size 36
MDFS: LC sending statistics message to RP with code 1 of size 680
MDFS: LC sending statistics message to RP with code 2 of size 200
MDFS: LC sending statistics message to RP with code 3 of size 152
MDFS: LC sending window message to RP with code 36261 of size 8
MDFS: LC received IPC packet of size 60 sequence 36212
```

The following is sample output from the **debug ip mds ipc event** command:

```
Router# debug ip mds ipc event
MDFS: LC received invalid sequence 21 while expecting 20
```

debug ip mds mevent

To debug Multicast Forwarding Information Base (MFIB) route creation, route updates, and so on, use the **debug ip mds mevent**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip mds mevent

no debug ip mds mevent

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

Use this command on the line card.

Examples

The following is sample output from the **debug ip mds mevent**command:

```
Router# debug ip mds mevent
MDFS mroute event debugging is on
Router#clear ip mdfs for *
Router#
MDFS: Create (*, 239.255.255.255)
MDFS: Create (192.168.1.1/32, 239.255.255.255), RPF POS2/0/0
MDFS: Add OIF for mroute (192.168.1.1/239.255.255.255) on Fddi0/0/0
MDFS: Create (*, 224.2.127.254)
MDFS: Create (192.168.1.1/32, 224.2.127.254), RPF POS2/0/0
MDFS: Add OIF for mroute (192.168.1.1/224.2.127.254) on Fddi0/0/0
MDFS: Create (128.9.160.67/32, 224.2.127.254), RPF POS2/0/0
```

debug ip mds mpacket

To debug multicast distributed switching (MDS) events such as packet drops, interface drops, and switching failures, use the **debug ip mds mpacket**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip mds mpacket

no debug ip mds mpacket

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

Use this command on the line card.

debug ip mds process

To debug line card process level events, use the **debug ip mds process**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip mds process

no debug ip mds process

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

Use this command on the line card or Route Processor (RP).

Examples

The following is sample output from the **debug ip mds process**command:

```
Router# debug ip mds process
MDFS process debugging is on
Mar 19 16:15:47.448: MDFS: RP queueing mdb message for (210.115.194.5, 224.2.127.254) to
all linecards
Mar 19 16:15:47.448: MDFS: RP queueing midb message for (210.115.194.5, 224.2.127.254) to
all linecards
Mar 19 16:15:47.628: MDFS: RP servicing low queue for LC in slot 0
Mar 19 16:15:47.628: MDFS: RP servicing low queue for LC in slot 2
Mar 19 16:15:48.229: MDFS: RP queueing mdb message for (171.68.224.10, 224.2.127.254) to
all linecards
Mar 19 16:15:48.229: MDFS: RP queueing mdb message for (171.68.224.10, 224.2.127.254) to
all linecards
Mar 19 16:15:48.229: MDFS: RP queueing mdb message for (171.69.67.106, 224.2.127.254) to
all linecards
Mar 19 16:15:48.229: MDFS: RP queueing mdb message for (171.69.67.106, 224.2.127.254) to
all linecards
Mar 19 16:15:48.229: MDFS: RP queueing mdb message for (206.14.154.181, 224.2.127.254) to
all linecards
Mar 19 16:15:48.229: MDFS: RP queueing mdb message for (206.14.154.181, 224.2.127.254) to
all linecards
Mar 19 16:15:48.233: MDFS: RP queueing mdb message for (210.115.194.5, 224.2.127.254) to
all linecards
```

debug ip mfib adjacency

To enable debugging output for IPv4 Multicast Forwarding Information Base (MFIB) adjacency management activity, use the **debug ip mfib adjacency**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip mfib adjacency

no debug ip mfib adjacency

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS XE Release 2.1	This command was introduced.
15.0(1)M	This command was integrated into Cisco IOS Release 15.0(1)M.
12.2(33)SRE	This command was integrated into Cisco IOS Release 12.2(33)SRE.

Examples

The following example shows how to enable debugging output for IPv4 MFIB adjacency management activity:

```
Router# debug ip mfib adjacency
```

debug ip mfib db

To enable debugging output for IPv4 Multicast Forwarding Information Base (MFIB) route database management activity, use the **debug ip mfib db** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip mfib [**vrf** {*vrf-name*| *****}] **db** [*source-address* [*group-address*]| *group-address* [*source-address*]]

no debug ip mfib [**vrf** {*vrf-name*| *****}] **db** [*source-address* [*group-address*]| *group-address* [*source-address*]]

Syntax Description

vrf {*vrf-name* | *****}	(Optional) Enables debugging output for IPv4 MFIB route database management activity associated with Multicast Virtual Private Network (MVPN) routing and forwarding (MVRF) instances. After specifying the optional **vrf** keyword, you must specify either: • *vrf-name* --Name of an MVRF. Enables debugging output for IPv4 MFIB route database management activity associated with the MVRF specified for the *vrf-name* argument. • ***** --Enables debugging output for route database management activity associated with all tables (all MVRF tables and the global table).
source-address	(Optional) Multicast source address.
group-address	(Optional) Multicast group address.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS XE Release 2.1	This command was introduced.
15.0(1)M	This command was integrated into Cisco IOS Release 15.0(1)M.
12.2(33)SRE	This command was integrated into Cisco IOS Release 12.2(33)SRE.

Examples

The following example shows how to enable debugging output for IPv4 MFIB route database management activity:

```
Router# debug ip mfib db
```

debug ip mfib fs

To enable debugging output for IPv4 Multicast Forwarding Information Base (MFIB) fast switching activity, use the **debug ip mfib fs**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip mfib [**vrf** {*vrf-name*| *****}] **fs** [*source-address* [*group-address*]| *group-address* [*source-address*]]

no debug ip mfib [**vrf** {*vrf-name*| *****}] **fs** [*source-address* [*group-address*]| *group-address* [*source-address*]]

Syntax Description

vrf {*vrf-name* | *****}	(Optional) Enables debugging output for IPv4 MFIB fast switching activity associated with Multicast Virtual Private Network (MVPN) routing and forwarding (MVRF) instances. After specifying the optional **vrf** keyword, you must specify either: • *vrf-name* --Name of an MVRF. Enables debugging output for IPv4 MFIB fast switching activity associated with the MVRF specified for the *vrf-name* argument. • ***** --Enables debugging output for IPv4 MFIB fast switching activity associated with all tables (all MVRF tables and the global table).
source-address	(Optional) Multicast source address.
group-address	(Optional) Multicast group address.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS XE Release 2.1	This command was introduced.
15.0(1)M	This command was integrated into Cisco IOS Release 15.0(1)M.
12.2(33)SRE	This command was integrated into Cisco IOS Release 12.2(33)SRE.

Examples

The following example shows how to enable debugging output for IPv4 MFIB fast switching activity:

```
Router# debug ip mfib fs
```

debug ip mfib init

To enable debugging output for events related to IPv4 Multicast Forwarding Information Base (MFIB) system initialization, use the **debug ip mfib init**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip mfib init

no debug ip mfib init

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS XE Release 2.1	This command was introduced.
15.0(1)M	This command was integrated into Cisco IOS Release 15.0(1)M.
12.2(33)SRE	This command was integrated into Cisco IOS Release 12.2(33)SRE.

Examples

The following example shows how to enable debugging output for events related to IPv4 MFIB system initialization:

```
Router# debug ip mfib init
```

debug ip mfib interface

To enable debugging output for IPv4 Multicast Forwarding Information Base (MFIB) interfaces, use the **debug ip mfib interface**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip mfib interface

no debug ip mfib interface

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS XE Release 2.1	This command was introduced.
15.0(1)M	This command was integrated into Cisco IOS Release 15.0(1)M.
12.2(33)SRE	This command was integrated into Cisco IOS Release 12.2(33)SRE.

Examples

The following example shows how to enable debugging output for IPv4 MFIB interfaces:

```
Router# debug ip mfib interface
```

debug ip mfib mrib

To enable debugging output for IPv4 Multicast Forwarding Information Base (MFIB) communication with the IPv4 Multicast Routing Information Base (MRIB), use the **debug ip mfib mrib**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip mfib [**vrf** {*vrf-name*| *****}] **mrib** [*source-address* [*group-address*]| *group-address* [*source-address*]] **[detail]**

no debug ip mfib [**vrf** {*vrf-name*| *****}] **mrib** [*source-address* [*group-address*]| *group-address* [*source-address*]] **[detail]**

Syntax Description

vrf {*vrf-name*| *****}	(Optional) Enables debugging output for IPv4 MFIB communication with the IPv4 MRIB associated with Multicast Virtual Private Network (MVPN) routing and forwarding (MVRF) instances. After specifying the optional **vrf** keyword, you must specify either: • *vrf-name* --Name of an MVRF. Enables debugging output for IPv4 MFIB communication with the IPv4 MRIB associated with the MVRF specified for the *vrf-name* argument. • ***** --Enables debugging output for IPv4 MFIB communication with the IPv4 MRIB associated with all tables (all MVRF tables and the global table).
source-address	(Optional) Multicast source address.
group-address	(Optional) Multicast group address.
detail	(Optional) Displays detailed debugging output for IPv4 MFIB communication with the IPv4 MRIB.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS XE Release 2.1	This command was introduced.
15.0(1)M	This command was integrated into Cisco IOS Release 15.0(1)M.

Release	Modification
12.2(33)SRE	This command was modified. The **detail** keyword was added.
15.1(1)T	This command was modified. The **detail** keyword was added.

Examples

The following example shows how to enable debugging output for IPv4 MFIB communication with the IPv4 MRIB:

```
Router# debug ip mfib mrib
```

debug ip mfib nat

To enable debugging output for IPv4 Multicast Forwarding Information Base (MFIB) Network Address Translation (NAT) events associated with all tables, use the **debug ip mfib nat**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip mfib nat [*source-address* [*group-address*]| *group-address* [*source-address*]]

no debug ip mfib nat [*source-address* [*group-address*]| *group-address* [*source-address*]]

Syntax Description

source-address	(Optional) Multicast source address.
group-address	(Optional) Multicast group address.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
15.0(1)M	This command was introduced.
12.2(33)SRE	This command was integrated into Cisco IOS Release 12.2(33)SRE.

Examples

The following example shows how to enable debugging output for IPv4 MFIB NAT events associated with all tables:

```
Router# debug ip mfib nat
```

debug ip mfib pak

To enable debugging output for IPv4 Multicast Forwarding Information Base (MFIB) packet forwarding activity, use the **debug ip mfib pak**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip mfib [**vrf** {*vrf-name*| *****}] **pak** [*source-address* [*group-address*]| *group-address* [*source-address*]]

no debug ip mfib [**vrf** {*vrf-name*| *****}] **pak** [*source-address* [*group-address*]| *group-address* [*source-address*]]

Syntax Description

vrf {*vrf-name* | *****	(Optional) Enables debugging output for IPv4 MFIB packet forwarding activity associated with Multicast Virtual Private Network (MVPN) routing and forwarding (MVRF) instances. After specifying the optional **vrf** keyword, you must specify either: • ***** --Enables debugging output for IPv4 MFIB packet forwarding activity associated with all tables (all MVRF tables and the global table). • *vrf-name* --Name of an MVRF. Enables debugging output for IPv4 MFIB packet forwarding activity associated with the MVRF specified for the *vrf-name* argument.
source-address	(Optional) Multicast source address.
group-address	(Optional) Multicast group address.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS XE Release 2.1	This command was introduced.
15.0(1)M	This command was integrated into Cisco IOS Release 15.0(1)M.
12.2(33)SRE	This command was integrated into Cisco IOS Release 12.2(33)SRE.

Examples

The following example shows how to enable debugging output for IPv4 MFIB packet forwarding activity:

```
Router# debug ip mfib pak
```

debug ip mfib platform

To enable debugging output related to the hardware platform use of IPv4 Multicast Forwarding Information Base (MFIB) application program interfaces (APIs), use the **debug ip mfib platform**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip mfib [**vrf** {*vrf-name*| *****}] **platform** {**api**| **callbacks**| **errors**| **notify**| **trnx**}

no debug ip mfib [**vrf** {*vrf-name*| *****}] **platform** {**api**| **callbacks**| **errors**| **notify**| **trnx**}

Syntax Description

vrf {*vrf-name* | *****}	(Optional) Enables debugging output related to the hardware platform use of IPv4 MFIB APIs associated with Multicast Virtual Private Network (MVPN) routing and forwarding (MVRF) instances. After specifying the optional **vrf** keyword, you must specify either: • *vrf-name* --Name of an MVRF. Enables debugging output related to the hardware platform use of IPv4 MFIB APIs associated with the MVRF specified for the *vrf-name* argument. • ***** --Enables debugging output related to the hardware platform use of IPv4 MFIB APIs associated with all tables (all MVRF tables and the global table).
api	Enables debugging output related to the hardware platform use of IPv4 MFIB API calls.
callbacks	Enables debugging output related to the hardware platform use of IPv4 MFIB API callbacks.
errors	Enables debugging output related to the hardware platform use of IPv4 MFIB API errors.
notify	Enables debugging output related to the hardware platform use of IPv4 MFIB notifications.
trnx	Enables debugging output related to the hardware platform use of IPv4 MFIB database transactions.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS XE Release 2.1	This command was introduced.
15.0(1)M	This command was integrated into Cisco IOS Release 15.0(1)M.
12.2(33)SRE	This command was integrated into Cisco IOS Release 12.2(33)SRE.

Examples

The following example shows how to enable debugging output related to the hardware platform use of IPv4 MFIB API errors:

```
Router# debug ip mfib platform errors
```

debug ip mfib ppr

To enable debugging output for IPv4 Multicast Forwarding Information Base (MFIB) packet preservation events, use the **debug ip mfib ppr**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip mfib [**vrf** {*vrf-name*| *****}] **ppr** [**errors**| **limit**| **preserve**| **release**| **trnx**] [*source-address* [*group-address*]| *group-address* [*source-address*]]

no debug ip mfib [**vrf** {*vrf-name*| *****}] **ppr** [**errors**| **limit**| **preserve**| **release**| **trnx**] [*source-address* [*group-address*]| *group-address* [*source-address*]]

Syntax Description

vrf {*vrf-name* | *****}	(Optional) Enables debugging output for IPv4 MFIB packet preservation events associated with Multicast Virtual Private Network (MVPN) routing and forwarding (MVRF) instances. After specifying the optional **vrf** keyword, you must specify either: • *vrf-name* --Name of an MVRF. Enables debugging output for IPv4 MFIB packet preservation events associated with the MVRF specified for the *vrf-name* argument. • ***** --Enables debugging output for IPv4 MFIB packet preservation events associated with all tables (all MVRF tables and the global table).
errors	(Optional) Enables debugging output for IPv4 MFIB packet preservation errors.
limit	(Optional) Enables debugging output for IPv4 MFIB packet preservation limits.
preserve	(Optional) Enables debugging output for IPv4 MFIB packet preservation events.
release	(Optional) Enables debugging output for IPv4 MFIB packet preservation release events.
trnx	(Optional) Enables debugging output for IPv4 MFIB packet preservation database transaction events.
source-address	(Optional) Multicast source address.
group-address	(Optional) Multicast group address.

Command Modes Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS XE Release 2.1	This command was introduced.
15.0(1)M	This command was integrated into Cisco IOS Release 15.0(1)M.
12.2(33)SRE	This command was integrated into Cisco IOS Release 12.2(33)SRE.

Examples The following example shows how to enable debugging output for IPv4 MFIB packet preservation errors:

```
Router# debug ip mfib ppr errors
```

debug ip mfib ps

To enable debugging output for IPv4 Multicast Forwarding Information Base (MFIB) process switching activity, use the **debug ip mfib ps**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip mfib [**vrf** {*vrf-name*| *****}] **ps** [*source-address* [*group-address*]| *group-address* [*source-address*]]

no debug ip mfib [**vrf** {*vrf-name*| *****}] **ps** [*source-address* [*group-address*]| *group-address* [*source-address*]]

Syntax Description

vrf {*vrf-name* | *****}	(Optional) Enables debugging output for IPv4 MFIB process switching activity associated with Multicast Virtual Private Network (MVPN) routing and forwarding (MVRF) instances. After specifying the optional **vrf** keyword, you must specify either: • *vrf-name* --Name of an MVRF. Enables debugging output for IPv4 MFIB process switching activity associated with the MVRF specified for the *vrf-name* argument. • ***** --Enables debugging output for IPv4 MFIB process switching activity associated with all tables (all MVRF tables and the global table).
source-address	(Optional) Multicast source address.
group-address	(Optional) Multicast group address.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS XE Release 2.1	This command was introduced.
15.0(1)M	This command was integrated into Cisco IOS Release 15.0(1)M.
12.2(33)SRE	This command was integrated into Cisco IOS Release 12.2(33)SRE.

Examples

The following example shows how to enable debugging output for IPv4 MFIB process switching activity:

```
Router# debug ip mfib ps
```

debug ip mfib signal

To enable debugging output for IPv4 Multicast Forwarding Information Base (MFIB) signal activity, use the **debug ip mfib signal**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip mfib [**vrf** {*vrf-name*| *****}] **signal** [*source-address* [*group-address*]| *group-address* [*source-address*]]

no debug ip mfib [**vrf** {*vrf-name*| *****}] **signal** [*source-address* [*group-address*]| *group-address* [*source-address*]]

Syntax Description

vrf {*vrf-name* | *****}	(Optional) Enables debugging output for IPv4 MFIB signal activity associated with Multicast Virtual Private Network (MVPN) routing and forwarding (MVRF) instances. After specifying the optional **vrf** keyword, you must specify either: • *vrf-name* --Name of an MVRF. Enables debugging output for IPv4 MFIB signal activity associated with the MVRF specified for the *vrf-name* argument. • ***** --Enables debugging output for IPv4 MFIB fast signal activity associated with all tables (all MVRF tables and the global table).
source-address	(Optional) Multicast source address.
group-address	(Optional) Multicast group address.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS XE Release 2.1	This command was introduced.
15.0(1)M	This command was integrated into Cisco IOS Release 15.0(1)M.
12.2(33)SRE	This command was integrated into Cisco IOS Release 12.2(33)SRE.

Examples

The following example shows how to enable debugging output for IPv4 MFIB signal activity for the default IPv4 table:

```
Router# debug ip mfib signal
```

The following example shows how to enable debugging output for IPv4 MFIB signal activity for the group 224.0.1.40, the source 10.1.1.1, and for the VRF Mgmt-intf:

```
Router# debug ip mfib vrf Mgmt-intf signal 10.1.1.1 224.0.1.40
```

debug ip mfib table

To enable debugging output for IPv4 Multicast Forwarding Information Base (MFIB) table activity, use the **debug ip mfib table**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip mfib [**vrf** {*vrf-name*| *****}] **table** {**db**| **mrib**}

no debug ip mfib [**vrf** {*vrf-name*| *****}] **table** {**db**| **mrib**}

Syntax Description

vrf {*vrf-name* | *****}	(Optional) Enables debugging output for IPv4 MFIB signal activity associated with Multicast Virtual Private Network (MVPN) routing and forwarding (MVRF) instances. After specifying the optional **vrf** keyword, you must specify either: • *vrf-name* --Name of an MVRF. Enables debugging output for IPv4 MFIB signal activity associated with the MVRF specified for the *vrf-name* argument. • ***** --Enables debugging output for IPv4 MFIB fast signal activity associated with all tables (all MVRF tables and the global table).
db	Enables debugging output for IPv4 MFIB database table events and operations.
mrib	Enables debugging output for IPv4 MFIB Multicast Routing Information Base (MRIB) API table events and operations.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS XE Release 2.1	This command was introduced.
15.0(1)M	This command was integrated into Cisco IOS Release 15.0(1)M.
12.2(33)SRE	This command was integrated into Cisco IOS Release 12.2(33)SRE.

Examples

The following example shows how to enable debugging output for IPv4 MFIB database table events and operations:

```
Router# debug ip mfib table db
```

The following example shows how to enable debugging output for IPv4 MFIB MRIB API table events and operations:

```
Router# debug ip mfib table mrib
```

debug ip mhbeat

To monitor the action of the heartbeat trap, use the **debug ip mhbeat**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip mhbeat

no debug ip mhbeat

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.1(2)XH	This command was introduced.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Examples

The following is sample output from the **debug ip mhbeat** command.

```
Router# debug ip mhbeat
IP multicast heartbeat debugging is on
Router debug snmp packets

SNMP packet debugging is on
!
Router(config)# ip multicast heartbeat intervals-of 10
 Dec 23 13:34:21.132: MHBEAT: ip multicast-heartbeat group 224.0.1.53 port 0
        source 0.0.0.0 0.0.0.0 at-least 3 in 5 intervals-of 10 secondsd
Router#
 Dec 23 13:34:23: %SYS-5-CONFIG_I: Configured from console by console
 Dec 23 13:34:31.136: MHBEAT: timer ticked, t=1,i=1,c=0
 Dec 23 13:34:41.136: MHBEAT: timer ticked, t=2,i=2,c=0
 Dec 23 13:34:51.136: MHBEAT: timer ticked, t=3,i=3,c=0
 Dec 23 13:35:01.136: MHBEAT: timer ticked, t=4,i=4,c=0
 Dec 23 13:35:11.136: MHBEAT: timer ticked, t=5,i=0,c=0
 Dec 23 13:35:21.135: Send SNMP Trap for missing heartbeat
 Dec 23 13:35:21.135: SNMP: Queuing packet to 171.69.55.12
 Dec 23 13:35:21.135: SNMP: V1 Trap, ent ciscoExperiment.2.3.1, addr 4.4.4.4, gentrap 6,
spectrap 1
  ciscoIpMRouteHeartBeat.1.0 = 224.0.1.53
  ciscoIpMRouteHeartBeat.2.0 = 0.0.0.0
  ciscoIpMRouteHeartBeat.3.0 = 10
  ciscoIpMRouteHeartBeat.4.0 = 5
  ciscoIpMRouteHeartBeat.5.0 = 0
  ciscoIpMRouteHeartBeat.6.0 = 3
```

Related Commands

Command	Description
ip multicast heartbeat	Monitors the health of multicast delivery, and alerts when the delivery fails to meet certain parameters.

debug ip mobile

To display IP mobility activities, use the **debug ip mobile**command in privileged EXEC mode.

debug ip mobile [**advertise**| **host** [*access-list-number*]| **local-area**| **redundancy**| **udp-tunneling**]

Syntax Description

advertise	(Optional) Advertisement information.
host	(Optional) The mobile node host.
access-list-number	(Optional) The number of an IP access list.
local-area	(Optional) The local area.
redundancy	(Optional) Redundancy activities.
udp-tunneling	(Optional) User Datagram Protocol (UDP) tunneling activities.

Command Default

No default behavior or values.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.0(1)T	This command was introduced.
12.0(2)T	The **standby** keyword was added.
12.2(8)T	The **standby** keyword was replaced by the **redundancy** keyword.
12.2(13)T	This command was enhanced to display information about foreign agent reverse tunnels and the mobile networks attached to the mobile router.
12.3(8)T	The **udp-tunneling** keyword was added and the command was enhanced to display information about NAT traversal using UDP tunneling.
12.3(7)XJ	This command was enhanced to include the Resource Management capability.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.

Usage Guidelines

Use the **debug ip mobile redundancy** command to troubleshoot redundancy problems.

No per-user debugging output is shown for mobile nodes using the network access identifier (NAI) for the **debug ip mobile host**command. Debugging of specific mobile nodes using an IP address is possible through the access list.

Examples

The following is sample output from the **debug ip mobile**command when foreign agent reverse tunneling is enabled:

```
MobileIP:MN 14.0.0.30 deleted from ReverseTunnelTable of Ethernet2/1(Entries 0)
```

The following is sample output from the **debug ip mobile advertise**command:

```
Router# debug ip mobile advertise
MobileIP: Agent advertisement sent out Ethernet1/2: type=16, len=10, seq=1, lifetime=36000,

flags=0x1400(rbhFmGv-rsv-),
Care-of address: 68.0.0.31
Prefix Length ext: len=1 (8 )
FA Challenge value:769C808D
```

The table below describes the significant fields shown in the display.

Table 31: debug ip mobile advertise Field Descriptions

Field	Description
type	Type of advertisement.
len	Length of extension (in bytes).
seq	Sequence number of this advertisement.
lifetime	Lifetime (in seconds).
flags	Capital letters represent bits that are set; lowercase letters represent unset bits.
Care-of address	IP address.
Prefix Length ext	Number of prefix lengths advertised. This is the bits in the mask of the interface sending this advertisement. Used for roaming detection.
FA Challenge value	Foreign Agent challenge value (randomly generated by the foreign agent.)

The following is sample output from the **debug ip mobile host**command:

```
Router# debug ip mobile host
MobileIP: HA received registration for MN 20.0.0.6 on interface Ethernet1 using COA
```

```
68.0.0.31 HA 66.0.0.5 lifetime 30000 options sbdmgvT
MobileIP: Authenticated FA 68.0.0.31 using SPI 110 (MN 20.0.0.6)
MobileIP: Authenticated MN 20.0.0.6 using SPI 300
MobileIP: HA accepts registration from MN 20.0.0.6
MobileIP: Mobility binding for MN 20.0.0.6 updated
MobileIP: Roam timer started for MN 20.0.0.6, lifetime 30000
MobileIP: MH auth ext added (SPI 300) in reply to MN 20.0.0.6
MobileIP: HF auth ext added (SPI 220) in reply to MN 20.0.0.6
MobileIP: HA sent reply to MN 20.0.0.6
```

The following is sample output from the **debug ip mobile redundancy**command. In this example, the active home agent receives a registration request from mobile node 20.0.0.2 and sends a binding update to peer home agent 1.0.0.2:

```
MobileIP:MN 20.0.0.2 - sent BindUpd to HA 1.0.0.2 HAA 20.0.0.1
MobileIP:HA standby maint started - cnt 1
MobileIP:MN 20.0.0.2 - sent BindUpd id 3780410816 cnt 0 elapsed 0
adjust -0 to HA 1.0.0.2 in grp 1.0.0.10 HAA 20.0.0.1
```

In this example, the standby home agent receives a binding update for mobile node 20.0.0.2 sent by the active home agent:

```
MobileIP:MN 20.0.0.2 - HA rcv BindUpd from 1.0.0.3 HAA 20.0.0.1
```

The following is sample output from the **debug ip mobile udp-tunneling** command and displays the registration, authentication, and establishment of UDP tunneling of a mobile node (MN) with a foreign agent (FA):

```
Dec 31 12:34:25.707: UDP: rcvd src=10.10.10.10(434),dst=10.30.30.1(434), length=54
Dec 31 12:34:25.707: MobileIP: ParseRegExt type MHAE(32) addr 2000FEEC end 2000FF02
Dec 31 12:34:25.707: MobileIP: ParseRegExt skipping 10 to next
Dec 31 12:34:25.707: MobileIP: FA rcv registration for MN 10.10.10.10 on Ethernet2/2 using
  COA 10.30.30.1 HA 10.10.10.100 lifetime 65535 options sbdmg-T-identification
C1BC0D4FB01AC0D8
Dec 31 12:34:25.707: MobileIP: Ethernet2/2 glean 10.10.10.10 accepted
Dec 31 12:34:25.707: MobileIP: Registration request byte count = 74
Dec 31 12:34:25.707: MobileIP: FA queued MN 10.10.10.10 in register table
Dec 31 12:34:25.707: MobileIP: Visitor registration timer started for MN 10.10.10.10,
lifetime 120
Dec 31 12:34:25.707: MobileIP:  Adding UDP Tunnel req extension
Dec 31 12:34:25.707: MobileIP: Authentication algorithm MD5 and 16 byte key
Dec 31 12:34:25.707: MobileIP: MN 10.10.10.10 FHAE added to HA 10.10.10.100 using SPI 1000
Dec 31 12:34:25.707: MobileIP: FA forwarded registration for MN 10.10.10.10 to HA
10.10.10.100
Dec 31 12:34:25.715: UDP: rcvd src=10.10.10.100(434), dst=10.30.30.1(434), length=94
Dec 31 12:34:25.715: MobileIP: ParseRegExt type NVSE(134) addr 20010B28 end 20010B6A
Dec 31 12:34:25.715: MobileIP: ParseRegExt type MN-config NVSE(14) subtype 1 (MN prefix
length) prefix length (24)
Dec 31 12:34:25.715: MobileIP: ParseRegExt skipping 12 to next
Dec 31 12:34:25.715: MobileIP: ParseRegExt type MHAE(32) addr 20010B36 end 20010B6A
Dec 31 12:34:25.715: MobileIP: ParseRegExt skipping 10 to next
Dec 31 12:34:25.715: MobileIP: ParseRegExt type UDPTUNREPE(44) addr 20010B4C end 20010B6A
Dec 31 12:34:25.715: Parsing UDP Tunnel Reply Extension - length 6
Dec 31 12:34:25.715: MobileIP: ParseRegExt skipping 6 to next
Dec 31 12:34:25.715: MobileIP: ParseRegExt type FHAE(34) addr 20010B54 end 20010B6A
Dec 31 12:34:25.715: MobileIP: ParseRegExt skipping 20 to next
Dec 31 12:34:25.715: MobileIP: FA rcv accept (0) reply for MN 10.10.10.10 on Ethernet2/3
using HA 10.10.10.100 lifetime 65535
Dec 31 12:34:25.719: MobileIP: Authenticating HA 10.10.10.100 using SPI 1000
Dec 31 12:34:25.719: MobileIP: Authentication algorithm MD5 and 16 byte key
Dec 31 12:34:25.719: MobileIP: Authenticated HA 10.10.10.100 using SPI 1000 and 16 byte
key
Dec 31 12:34:25.719: MobileIP: HA accepts UDP Tunneling
Dec 31 12:34:25.719: MobileIP: Update visitor table for MN 10.10.10.10
Dec 31 12:34:25.719: MobileIP: Enabling UDP Tunneling
Dec 31 12:34:25.719: MobileIP: Tunnel0 (MIPUDP/IP) created with src 10.30.30.1 dst
10.10.10.100
Dec 31 12:34:25.719: MobileIP: Setting up UDP Keep-Alive Timer for tunnel 10.30.30.1:0 -
10.10.10.100:0 with keep-alive 30
Dec 31 12:34:25.719: MobileIP: Starting the tunnel keep-alive timer
```

```
Dec 31 12:34:25.719: MobileIP: ARP entry for MN 10.10.10.10 using 10.10.10.10 inserted on
 Ethernet2/2
Dec 31 12:34:25.719: MobileIP: FA route add 10.10.10.10 successful. Code = 0
Dec 31 12:34:25.719: MobileIP: MN 10.10.10.10 added to ReverseTunnelTable of Ethernet2/2
(Entries 1)
Dec 31 12:34:25.719: MobileIP: FA dequeued MN 10.10.10.10 from register table
Dec 31 12:34:25.719: MobileIP: MN 10.10.10.10 using 10.10.10.10 visiting on Ethernet2/2 Dec
 31 12:34:25.719: MobileIP: Reply in for MN 10.10.10.10 using 10.10.10.10, accepted
Dec 31 12:34:25.719: MobileIP: registration reply byte count = 84
Dec 31 12:34:25.719: MobileIP: FA forwarding reply to MN 10.10.10.10 (10.10.10.10 mac
0060.70ca.f021)
Dec 31 12:34:26.095: MobileIP: agent advertisement byte count = 48
Dec 31 12:34:26.095: MobileIP: Agent advertisement sent out Ethernet2/2: type=16, len=10,
 seq=55, lifetime=65535, flags=0x1580(rbhFmG-TU),
Dec 31 12:34:26.095: Care-of address: 10.30.30.1
Dec 31 12:34:26.719: MobileIP: swif coming up Tunnel0
!
Dec 31 12:34:35.719: UDP: sent src=10.30.30.1(434), dst=10.10.10.100(434)
Dec 31 12:34:35.719: UDP: rcvd src=10.10.10.100(434), dst=10.30.30.1(434), length=32d0
```

The following is sample output from the **debug ip mobile udp-tunneling** command and displays the registration, authentication, and establishment of UDP tunneling of a MN with a home agent (HA):

```
Dec 31 12:34:26.167: MobileIP: ParseRegExt skipping 20 to next
Dec 31 12:34:26.167: MobileIP: ParseRegExt type UDPTUNREQE(144) addr 2001E762 end 2001E780
Dec 31 12:34:26.167: MobileIP: Parsing UDP Tunnel Request Extension - length 6
Dec 31 12:34:26.167: MobileIP: ParseRegExt skipping 6 to next
Dec 31 12:34:26.167: MobileIP: ParseRegExt type FHAE(34) addr 2001E76A end 2001E780
Dec 31 12:34:26.167: MobileIP: ParseRegExt skipping 20 to next
Dec 31 12:34:26.167: MobileIP: HA 167 rcv registration for MN 10.10.10.10 on Ethernet2/1
using HomeAddr 10.10.10.10 COA 10.30.30.1 HA 10.10.10.100 lifetime 65535 options
sbdmg-T-identification C1BC0D4FB01AC0D8
Dec 31 12:34:26.167: MobileIP: NAT detected SRC:10.10.10.50 COA: 10.30.30.1
Dec 31 12:34:26.167: MobileIP: UDP Tunnel Request accepted 10.10.10.50:434
Dec 31 12:34:26.167: MobileIP: Authenticating FA 10.30.30.1 using SPI 1000
Dec 31 12:34:26.167: MobileIP: Authentication algorithm MD5 and 16 byte key
Dec 31 12:34:26.167: MobileIP: Authentication algorithm MD5 and truncated key
Dec 31 12:34:26.167: MobileIP: Authentication algorithm MD5 and 16 byte key
Dec 31 12:34:26.167: MobileIP: Authenticated FA 10.30.30.1 using SPI 1000 and 16 byte key
Dec 31 12:34:26.167: MobileIP: Authenticating MN 10.10.10.10 using SPI 1000
Dec 31 12:34:26.167: MobileIP: Authentication algorithm MD5 and 16 byte key
Dec 31 12:34:26.167: MobileIP: Authentication algorithm MD5 and truncated key
Dec 31 12:34:26.167: MobileIP: Authentication algorithm MD5 and 16 byte key
Dec 31 12:34:26.167: MobileIP: Authenticated MN 10.10.10.10 using SPI 1000 and 16 byte key
Dec 31 12:34:26.167: MobileIP: Mobility binding for MN 10.10.10.10 created
Dec 31 12:34:26.167: MobileIP: NAT detected for MN 10.10.10.10. Terminating tunnel on
10.10.10.50
Dec 31 12:34:26.167: MobileIP: Tunnel0 (MIPUDP/IP) created with src 10.10.10.100 dst
10.10.10.50
Dec 31 12:34:26.167: MobileIP: Setting up UDP Keep-Alive Timer for tunnel 10.10.10.100:0 -
  10.10.10.50:0 with keep-alive 30
Dec 31 12:34:26.167: MobileIP: Starting the tunnel keep-alive timer
Dec 31 12:34:26.167: MobileIP: MN 10.10.10.10 Insert route for 10.10.10.10/255.255.255.255
  via gateway 10.10.10.50 on Tunnel0
Dec 31 12:34:26.167: MobileIP: MN 10.10.10.10 is now roaming
Dec 31 12:34:26.171: MobileIP: Gratuitous ARPs sent for MN 10.10.10.10 MAC 0002.fca5.bc39
Dec 31 12:34:26.171: MobileIP: Mask for address is 24
Dec 31 12:34:26.171: MobileIP: HA accepts registration from MN 10.10.10.10
Dec 31 12:34:26.171: MobileIP: Dynamic and Static Network Extension Length 0 - 0
Dec 31 12:34:26.171: MobileIP: Composed mobile network extension length:0
Dec 31 12:34:26.171: MobileIP: Added prefix length vse in reply
Dec 31 12:34:26.171: MobileIP: Authentication algorithm MD5 and 16 byte key
Dec 31 12:34:26.171: MobileIP: MN 10.10.10.10 MHAE added to MN 10.10.10.10 using SPI 1000
Dec 31 12:34:26.171: MobileIP: Authentication algorithm MD5 and 16 byte key
Dec 31 12:34:26.171: MobileIP: MN 10.10.10.10 FHAE added to FA 10.10.10.50 using SPI 1000
Dec 31 12:34:26.171: MobileIP: MN 10.10.10.10 - HA sent reply to 10.10.10.50
Dec 31 12:34:26.171: MobileIP: Authentication algorithm MD5 and 16 byte key
Dec 31 12:34:26.171: MobileIP: MN 10.10.10.10 HHAE added to HA 10.10.10.3 using SPI 1000
Dec 31 12:34:26.175: MobileIP: ParseRegExt type CVSE(38) addr 2000128C end 200012AE
Dec 31 12:34:26.175: MobileIP: ParseRegExt type HA red. version CVSE(6)
Dec 31 12:34:26.175: MobileIP: ParseRegExt skipping 8 to next
Dec 31 12:34:26.175: MobileIP: ParseRegExt type HHAE(35) addr 20001298 end 200012AE
```

```
Dec 31 12:34:26.175: MobileIP: ParseRegExt skipping 20 to next
Dec 31 12:34:26.175: MobileIP: Authenticating HA 10.10.10.3 using SPI 1000
Dec 31 12:34:26.175: MobileIP: Authentication algorithm MD5 and 16 byte key
Dec 31 12:34:26.175: MobileIP: Authentication algorithm MD5 and truncated key
Dec 31 12:34:26.175: MobileIP: Authentication algorithm MD5 and 16 byte key
Dec 31 12:34:26.175: MobileIP: Authenticated HA 10.10.10.3 using SPI 1000 and 16 byte key
Dec 31 12:34:27.167: MobileIP: swif coming up Tunnel0d0
```

debug ip mobile advertise

The debug ip mobile advertise command was consolidated with the debug ip mobile command. See the description of the debug ip mobile command in the “Debug Commands” chapter for more information.

To display advertisement information, use the **debug ip mobile advertise** EXEC command .

debug ip mobile advertise

no debug ip mobile advertise

Syntax Description

This command has no arguments or keywords.

Command Default

No default values.

Command Modes

EXEC mode

Command History

Release	Modification
12.0(1)T	This command was introduced.

Examples

The following is sample output from the **debug ip mobile advertise**command. The table below describes significant fields shown in the display.

```
Router# debug ip mobile advertise
MobileIP: Agent advertisement sent out Ethernet1/2: type=16, len=10, seq=1,
lifetime=36000,
flags=0x1400(rbhFmGv-rsv-),
Care-of address: 14.0.0.31
Prefix Length ext: len=1 (8 )
```

Table 32: Debug IP Mobile Advertise Field Descriptions

Field	Description
type	Type of advertisement.
len	Length of extension in bytes.
seq	Sequence number of this advertisement.
lifetime	Lifetime in seconds.
flags	Capital letters represent bits that are set, lower case letters represent unset bits.

Field	Description
Care-of address	IP address.
Prefix Length ext	Number of prefix lengths advertised. This is the bits in the mask of the interface sending this advertisement. Used for roaming detection.

debug ip mobile dyn-pbr

To display debugging messages for the mobile IP (MIP) dynamic policy based routing (PBR) mobile router, use the **debug ip mobile dyn-pbr** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip mobile dyn-pbr

no debug ip mobile dyn-pbr

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.4(24)T	This command was introduced.

Examples

The following sample output from the **debug ip mobile dyn-pbr**command:

```
Router# debug ip mobile dyn-pbr
*Jan 12 19:50:16.271: %LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel2, changed state
 to up *Jan 12 19:50:16.271:        Looking for path WIFI in rmap MPATH_2 10 *Jan 12
19:50:16.271:        Found link_type WIFI, ACL template is VIDEO *Jan 12 19:50:16.271:
    Set int for link_type WIFI to Tunnel2 *Jan 12 19:50:16.271:        MIP-PBR: ACL handle
 VIDEO-to-192.0.2.0/24 created *Jan 12 19:50:16.271:        MIP-PBR: Retrieving ACL for
VIDEO-to-192.0.2.0/24
*Jan 12 19:50:16.271:        template->tos_value = 16 *Jan 12 19:50:16.271:        Creating
 new rmap entry_hdl 104835472 *Jan 12 19:50:16.271:        new dyn rmap info added to
map_entry->dyn_rmaps
*Jan 12 19:50:16.271:          map_entry->dyn_rmaps =
*Jan 12 19:50:16.271:               104835472, VIDEO-to-192.0.2.0/24
*Jan 12 19:50:16.271:        MIP-PBR: added route-map entry for
VIDEO-to-192.0.2.0/24 via Tunnel2
*Jan 12 19:50:16.271:       MIP-PBR: Dyn route-map entry added OK on HA *Jan 12 19:50:16.271:
        MIP-PBR: ACL handle VIDEO-to-192.0.2.32/20 created *Jan 12 19:50:16.271:
MIP-PBR: Retrieving ACL for
VIDEO-to-192.0.2.32/20
*Jan 12 19:50:16.271:        template->tos_value = 16 *Jan 12 19:50:16.271:        Creating
 new rmap entry_hdl 84396264 *Jan 12 19:50:16.271:        new dyn rmap info added to
map_entry->dyn_rmaps
*Jan 12 19:50:16.271:          map_entry->dyn_rmaps =
*Jan 12 19:50:16.271:               104835472, VIDEO-to-192.0.2.0/24
*Jan 12 19:50:16.271:               84396264, VIDEO-to-192.0.2.32/20
*Jan 12 19:50:16.271:        MIP-PBR: added route-map entry for
VIDEO-to-192.0.2.32/20 via Tunnel2
*Jan 12 19:50:16.271:        MIP-PBR: Dyn route-map entry added for home address 192.0.2.32
 on HA *Jan 12 19:50:16.271:        Looking for path WIFI in rmap MPATH_2 20 *Jan 12
19:50:16.271:        Looking for path WIFI in rmap MPATH_2 30 *Jan 12 19:50:16.271:
 MIP-PBR: MIP-01/12/09-19:46:39.495-1-MP-HA assoc with Ethernet2/0 *Jan 12 19:50:16.271:
      *Jan 12 19:50:16.271:        *Jan 12 19:50:16.271:        Looking for path WIFI in
rmap MPATH_1 10 *Jan 12 19:50:16.271:        Found link_type WIFI, ACL template is VIDEO
*Jan 12 19:50:16.271:        Set int for link_type WIFI to Tunnel2 *Jan 12 19:50:16.271:
      MIP-PBR: Using existing dyn acl hdl
VIDEO-to-192.0.2.0/24
```

```
*Jan 12 19:50:16.271:        MIP-PBR:   After api bind, ACL
VIDEO-to-192.0.2.0/24, user_count 3
*Jan 12 19:50:16.271:        MIP-PBR:   current map_entry->dyn_rmaps = 0
*Jan 12 19:50:16.271:        MIP-PBR:   found rmap_info =
VIDEO-to-192.0.2.0/24
*Jan 12 19:50:16.271:        MIP-PBR: Using existing dyn rmap entry
104835472
*Jan 12 19:50:16.271:        MIP-PBR: added route-map entry for
VIDEO-to-192.0.2.0/24 via Tunnel2
*Jan 12 19:50:16.271:       MIP-PBR: Dyn route-map entry added OK on HA *Jan 12 19:50:16.271:
        MIP-PBR: Using existing dyn acl hdl
VIDEO-to-192.0.2.32/20
*Jan 12 19:50:16.271:        MIP-PBR:   After api bind, ACL
VIDEO-to-192.0.2.32/20, user_count 3
*Jan 12 19:50:16.271:        MIP-PBR:   current map_entry->dyn_rmaps =
63A5320
*Jan 12 19:50:16.271:        MIP-PBR:   found rmap_info =
VIDEO-to-192.0.2.32/20
*Jan 12 19:50:16.271:        MIP-PBR: Using existing dyn rmap entry
84396264
*Jan 12 19:50:16.271:        MIP-PBR: added route-map entry for
VIDEO-to-192.0.2.32/20 via Tunnel2
*Jan 12 19:50:16.271:        MIP-PBR: Dyn route-map entry added for home address 192.0.2.32
 on HA *Jan 12 19:50:16.271:        Looking for path WIFI in rmap MPATH_1 20 *Jan 12
19:50:16.271:        Looking for path WIFI in rmap MPATH_1 30 *Jan 12 19:50:16.271:
 MIP-PBR: MIP-01/12/09-19:46:39.495-1-MP-HA assoc with Ethernet2/0 *Jan 12 19:50:16.271:
      *Jan 12 19:50:16.271:       *Jan 12 19:50:16.271: %LINEPROTO-5-UPDOWN: Line protocol
 on Interface Tunnel3, changed state to up *Jan 12 19:50:16.271: %LINEPROTO-5-UPDOWN: Line
 protocol on Interface Tunnel4, changed state to up *Jan 12 19:50:16.271:        Looking
for path UMTS in rmap MPATH_2 10 *Jan 12 19:50:16.271:        Looking for path UMTS in rmap
 MPATH_2 20 *Jan 12 19:50:16.271:        Found link_type UMTS, ACL template is VOICE *Jan
12 19:50:16.271:        Set int for link_type UMTS to Tunnel4 *Jan 12 19:50:16.271:
 MIP-PBR: ACL handle VOICE-to-192.0.2.0/24 created *Jan 12 19:50:16.271:        MIP-PBR:
Using existing dyn acl hdl
VOICE-to-192.0.2.0/24
*Jan 12 19:50:16.271:        MIP-PBR:   After api bind, ACL
VOICE-to-192.0.2.0/24, user_count 3
*Jan 12 19:50:16.271:        MIP-PBR:   current map_entry->dyn_rmaps = 0
*Jan 12 19:50:16.271:        MIP-PBR:   found rmap_info =
VOICE-to-192.0.2.0/24
*Jan 12 19:50:16.271:        MIP-PBR: Using existing dyn rmap entry 84365440 *Jan 12
19:50:16.271:        MIP-PBR: added route-map entry for
VOICE-to-192.0.2.0/24 via Tunnel4
*Jan 12 19:50:16.271:       MIP-PBR: Dyn route-map entry added OK on HA *Jan 12 19:50:16.271:
        MIP-PBR: Using existing dyn acl hdl
VOICE-to-192.0.2.32/20
*Jan 12 19:50:16.271:        MIP-PBR:   After api bind, ACL
VOICE-to-192.0.2.32/20, user_count 3
*Jan 12 19:50:16.271:        MIP-PBR:   current map_entry->dyn_rmaps =
63A4390
*Jan 12 19:50:16.271:        MIP-PBR:   found rmap_info =
VOICE-to-192.0.2.32/20
*Jan 12 19:50:16.271:        MIP-PBR: Using existing dyn rmap entry
99337152
*Jan 12 19:50:16.271:        MIP-PBR: added route-map entry for
VOICE-to-192.0.2.32/20 via Tunnel4
*Jan 12 19:50:16.271:        MIP-PBR: Dyn route-map entry added for home address 192.0.2.32
 on HA *Jan 12 19:50:16.271:        Looking for path UMTS in rmap MPATH_1 30 *Jan 12
19:50:16.271:       MIP-PBR: MIP-01/12/09-19:46:39.495-1-MP-HA assoc with Ethernet2/0 *Jan
 12 19:50:16.271:        *Jan 12 19:50:16.271:
*Jan 12 19:50:16.291:            DELETING dyn_rmaps for reg_ptr 6436320:
*Jan 12 19:50:16.291:            Looking at reg_info: Tunnel2 MPATH_1 10
*Jan 12 19:50:16.291:            Looking at reg_info: Tunnel2 MPATH_2 10
*Jan 12 19:50:16.291:            Looking at reg_info: Tunnel2 MPATH_1 10
*Jan 12 19:50:16.291:            Looking at reg_info: Tunnel2 MPATH_2 10
*Jan 12 19:50:16.291:            Looking at reg_info: Tunnel4 MPATH_1 20
*Jan 12 19:50:16.291:            Looking at reg_info: Tunnel4 MPATH_2 20
*Jan 12 19:50:16.291:            Looking at reg_info: Tunnel4 MPATH_1 20
*Jan 12 19:50:16.291:            Looking at reg_info: Tunnel4 MPATH_2 20
*Jan 12 19:50:16.291:            Looking at reg_info: Tunnel2 MPATH_1 10
*Jan 12 19:50:16.291:            Looking at reg_info: Tunnel2 MPATH_2 10
*Jan 12 19:50:16.291:            Looking at reg_info: Tunnel2 MPATH_1 10
*Jan 12 19:50:16.291:            Looking at reg_info: Tunnel2 MPATH_2 10
```

debug ip mobile host

The **debug ip mobile host** command was consolidated with the **debug ip mobile**command. See the description of the **debug ip mobile** command in the "Debug Commands" chapter for more information.

Use the **debug ip mobile host** EXEC command to display IP mobility events.

debug ip mobile host [access-list-number]| [**nai** {**NAI username**| **username@realm**}]

no debug ip mobile host [access-list-number]| [**nai** {**NAI username**| **username@realm**}]

Syntax Description

access-list-number	(Optional) The mobile node host.
nai { *NAI username* | *username@realm* }	(Optional) Mobile host identified by NAI.

Command Default

No default values.

Command History

Release	Modification
12.0(1)T	This command was introduced.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.

Examples

The following is sample output from the **debug ip mobile host** command:

```
Router# debug ip mobile host
MobileIP: HA received registration for MN 10.0.0.6 on interface Ethernet1 using COA
14.0.0.31 HA 15.0.0.5 lifetime 30000 options sbdmgvT
MobileIP: Authenticated FA 15.0.0.31 using SPI 110 (MN 20.0.0.6)
MobileIP: Authenticated MN 11.0.0.6 using SPI 300

MobileIP: HA accepts registration from MN 11.0.0.6
MobileIP: Mobility binding for MN 11.0.0.6 updated
MobileIP: Roam timer started for MN 11.0.0.6, lifetime 30000
MobileIP: MH auth ext added (SPI 300) in reply to MN 11.0.0.6
MobileIP: HF auth ext added (SPI 220) in reply to MN 11.0.0.6

MobileIP: HA sent reply to MN 11.0.0.6
```

debug ip mobile mib

To display deb ugging messages for mobile networks, use the **debug ip mobile mib**command in privileged EXEC mode. To disable, use the **no** form of this command.

debug ip mobile mib

no debug ip mobile mib

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(4)T	This command was introduced.

Usage Guidelines

This command is useful for customers deploying mobile networks functionality that need to monitor and debug mobile router information via the Simple Network Management Protocol (SNMP).

Set operations (performed from a Network Management System) are supported for mobile network services. While setting the values for MIBs, a set operation may fail. The **debug ip mobile mib** command allows error messages explaining the failure to be displayed on the console of the home agent .

Examples

The following mobile networks deployment MIB debug messages are displayed only on certain conditions or when a certain condition fails.

```
Router# debug ip mobile mib
! Mobile router is not enabled
MIPMIB: Mobile Router is not enabled
! Care-of-interface can be set as transmit-only only if its a Serial interface
MIPMIB: Serial interfaces can only be set as transmit-only
! The Care of address can be configured only if foreign agent is running
MIPMIB: FA cannot be started
! Check if home agent is active
MIPMIB: HA is not enabled
! For mobile router configuration, host configuration must have been done already
MIPMIB: MN <address> is not configured
! Mobile Network does not match the existing mobile network
MIPMIB: Conflict with existing mobile networks <name>
! Mobile router present
MIPMIB: MR <address> is not configured

! Static mobile networks can be configured only for single member mobilenetgroups
MIPMIB: MR is part of group <name>, network cannot be configured
! If a binding exists for this mobile router, then delete the route for this unconfigured
! mobile network
```

```
MIBMIB: Delete static mobile net for MR
! Check if its a dynamically registered mobile network
nMIPMIB: Mobile network <address mask> is dynamically registered, cannot be removed
! Check if the mobile network has already been configured for another group
nMIPMIB: Mobile network already configured for MR
! Check if the network has been dynamically registered
nMIPMIB: Deleted dynamic mobnet <address mask> for MR <name>
! Check if the redundancy group exists
MIPMIB: Redundancy group <name> does not exist
! CCOA configuration, use primary interface address as the CCoA
MIPMIB: No IP address on this interface
! CCOA configuration, CCoA address shouldn't be the same as the Home Address
```

nMIPMIB: Collocated CoA is the same as the Home Address, registrations will fail

debug ip mobile redundancy

The debug ip mobile redundancy command was consolidated with the debug ip mobile command. See the description of the debug ip mobile command in the “Debug Commands” chapter for more information.

Use the **debug ip mobile redundancy** EXEC command to display IP mobility events.

debug ip mobile redundancy

no debug ip mobile redundancy

Syntax Description

This command has no keywords or arguments.

Command Default

No default values.

Command History

Release	Modification
12.0(1)T	This command was introduced.

Examples

The following is sample output from the debug ip mobile redundancy command:

```
Router# debug ip mobile redundancy
00:19:21: MobileIP: Adding MN service flags to bindupdate
00:19:21: MobileIP:  Adding MN service flags 0 init registration flags 1
00:19:21: MobileIP: Adding a hared version cvse - bindupdate
00:19:21: MobileIP: HARelayBindUpdate version number 2MobileIP: MN 14.0.0.20 - sent BindUpd
 to HA 11.0.0.3 HAA 11.0.0.4
00:19:21: MobileIP: HA standby maint started - cnt 1
00:19:21: MobileIP: MN 14.0.0.20 - HA rcv BindUpdAck accept from 11.0.0.3 HAA 11.0.0.4
00:19:22: MobileIP: HA standby maint started - cnt 1
```

debug ip mobile router

To display debugging messages for the mobile router, use the **debug ip mobile router**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip mobile router [detail]

no debug ip mobile router [detail]

Syntax Description

detail	(Optional) Displays detailed mobile router debug messages.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(4)T	This command was introduced.
12.2(13)T	This command was enhanced to display information about the addition and deletion of mobile networks.
15.4(3)T	This command was enhanced to display information about Multi-VRF for Network Mobility.

Usage Guidelines

The mobile router operations can be debugged. The following conditions trigger debugging messages:

- Agent discovery
- Registration
- Mobile router state change
- Routes and tunnels created or deleted
- Roaming information

Debugging messages are prefixed with “MobRtr” and detail messages are prefixed with “MobRtrX”.

Examples

The following is sample output from the **debug ip mobile router** command:

```
Device# debug ip mobile router
```

```
MobileRouter: New FA 27.0.0.12 coa 27.0.0.12 int Ethernet0/1 MAC 0050.50c1.c855
2w2d: MobileRouter: Register reason: isolated
2w2d: MobileRouter: Snd reg request agent 27.0.0.12 coa 27.0.0.12 home 9.0.0.1 ha 29.0.0.4
 lifetime 36000 int Ethernet0/1 flag sbdmgvt cnt 0 id B496B69C.55E77974
2w2d: MobileRouter: Status Isolated -> Pending
```

The following is sample output from the **debug ip mobile router detail** command:

```
Device# debug ip mobile router detail
1d09h: MobRtr: New agent 20.0.0.2 coa 30.0.0.2 int Ethernet3/1 MAC 00b0.8e35.a055
1d09h: MobRtr: Register reason: left home
1d09h: MobRtrX: Extsize 18 add 1 delete 0
1d09h: MobRtrX: Add network 20.0.0.0/8
MobileIP: MH auth ext added (SPI 100) to HA 100.0.0.3
1d09h: MobRtr: Register to fa 20.0.0.2 coa 30.0.0.2 home 100.0.0.1 ha 100.0.0.3 life 120
int Ethernet3/1 flag sbdmgvt cnt 0 id BE804340.447F50A4
1d09h: MobRtr: Status Isolated -> Pending
1d09h: MobRtr: MN rcv accept (0) reply on Ethernet3/1 from 20.0.0.2 lifetime 120
MobileIP: MN 100.0.0.3 - authenticating HA 100.0.0.3 using SPI 100
MobileIP: MN 100.0.0.3 - authenticated HA 100.0.0.3 using SPI 100
1d09h: MobRtr: Status Pending -> Registered
1d09h: MobRtr: Add default gateway 20.0.0.2 (Ethernet3/1)
1d09h: MobRtr: Add default route via 20.0.0.2 (Ethernet3/1)
```

The following is sample output from the **debug ip mobile router detail** command when Multi-VRF for Network Mobility feature is configured:

```
Device# debug ip mobile router detail
1d09h: MobRtr: New agent 10.0.0.2 coa 10.1.0.2 int Ethernet3/1 MAC 00b0.8e35.a055
1d09h: MobRtr: Register reason: left home
1d09h: MobRtrX: Extsize 18 add 1 delete 0
1d09h: MobRtrX: Add network 10.0.0.0/8
MobileIP: MH auth ext added (SPI 100) to HA 10.0.1.3
1d09h: MobRtr: Register to fa 10.1.0.20 coa 30.0.0.2 home 10.0.10.11 ha 10.1.1.3 life 120
int Ethernet3/1 flag sbdmgvt cnt 0 id BE804340.447F50A4
1d09h: MobRtr: Status Isolated -> Pending
1d09h: MobRtr: MN rcv accept (0) reply on Ethernet3/1 from 10.3.1.2 lifetime 120
MobileIP: MN 10.0.0.3 - authenticating HA 10.0.0.3 using SPI 100
MobileIP: MN 10.0.0.3 - authenticated HA 10.0.0.3 using SPI 100
1d09h: MobRtr: Status Pending -> Registered
1d09h: MobRtr: Add default gateway 10.20.1.2 (Ethernet3/1)
1d09h: MobRtr: Add default route via 10.2.1.2 (Ethernet3/1)
```

Related Commands

Command	Description
debug ip mobile	Displays Mobile IP information.

debug ip mpacket

Note Effective with Cisco IOS Release 15.0(1)M and Cisco IOS Release 12.2(33)SRE, the **debug ip mpacket**command is replaced by the **debug ip mfib ps**command and the **debug ip mcache**command with the **fastswitch** keyword is replaced by the **debug ip mfib pak** command. See the **debug ip mfib ps**and **debug ip mfib pak** commands for more information.

To display IP multicast packets received and sent, use the **debug ip mpacket** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip mpacket [**vrf** *vrf-name*] [**detail| fastswitch**] [*access-list*] [*group*]

no debug ip mpacket [**vrf** *vrf-name*] [**detail| fastswitch**] [*access-list*] [*group*]

Syntax Description

vrf	(Optional) Supports the Multicast Virtual Private Network (VPN) routing and forwarding (VRF) instance.
vrf-name	(Optional) Name assigned to the VRF.
detail	(Optional) Displays IP header information and MAC address information.
fastswitch	(Optional) Displays IP packet information in the fast path.
access-list	(Optional) The access list number.
group	(Optional) The group name or address.

Command Default

The **debug ip mpacket** command displays all IP multicast packets switched at the process level.

Command Modes

Privileged EXEC

Command History

Release	Modification
10.2	This command was introduced.
12.1(2)T	This command was modified. The **fastswitch** keyword was added.
12.0(23)S	This command was modified. The **vrf** keyword and *vrf-name* argument were added.

Release	Modification
12.2(13)T	This command was integrated into Cisco IOS Release 12.2(13)T.
12.2(14)S	This command was integrated into Cisco IOS Release 12.2(14)S.
12.2(27)SBC	This command was integrated into Cisco IOS Release 12.2(27)SBC.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
15.0(1)M	This command was replaced.
12.2(33)SRE	This command was replaced.

Usage Guidelines

This command displays information for multicast IP packets that are forwarded from this router. Use the *access-list* or *group* argument to limit the display to multicast packets from sources described by the access list or a specific multicast group.

Use this command with the **debug ip packet** command to display additional packet information.

Note The **debug ip mpacket** command generates many messages. Use this command with care so that performance on the network is not affected by the debug message traffic.

Examples

The following is sample output from the **debug ip mpacket** command:

```
Router# debug ip mpacket 224.2.0.1

IP: s=10.188.34.54 (Ethernet1), d=224.2.0.1 (Tunnel0), len 88, mforward
IP: s=10.188.34.54 (Ethernet1), d=224.2.0.1 (Tunnel0), len 88, mforward
IP: s=10.188.34.54 (Ethernet1), d=224.2.0.1 (Tunnel0), len 88, mforward
IP: s=10.162.3.27 (Ethernet1), d=224.2.0.1 (Tunnel0), len 68, mforward
```

The table below describes the significant fields shown in the display.

Table 33: debug ip mpacket Field Descriptions

Field	Description
IP	IP packet.
s=10.188.34.54	Source address of the packet.
(Ethernet1)	Name of the interface that received the packet.
d=224.2.0.1	Multicast group address that is the destination for this packet.
(Tunnel0)	Outgoing interface for the packet.

Field	Description
len 88	Number of bytes in the packet. This value will vary depending on the application and the media.
mforward	Packet has been forwarded.

Related Commands

Command	Description
debug ip dvmrp	Displays information on DVMRP packets received and sent.
debug ip igmp	Displays IGMP packets received and sent, and IGMP host-related events.
debug ip mrm	Displays MRM control packet activity.
debug ip packet	Displays general IP debugging information and IPSO security transactions.
debug ip sd	Displays all SD announcements received.

debug ip mrib

To enable debugging output for IPv4 Multicast Routing Information Base (MRIB) activity, use the **debug ip mrib**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip mrib [**vrf** *vrf-name*] {**client| io| issu| proxy| route| table| trans**}

no debug ip mrib [**vrf** *vrf-name*] {**client| io| issu| proxy| route| table| trans**}

Syntax Description

vrf *vrf-name*	(Optional) Enables debugging output for IPv4 MRIB activity associated with the Multicast Virtual Private Network (MVPN) routing and forwarding (MVRF) instance specified for the *vrf-name* argument.
client	Enables debugging output for IPv4 MRIB client management activity.
io	Enables debugging output for IPv4 MRIB input/output (I/O) events.
issu	Enables debugging output for IPv4 MRIB events associated with In-Service Software Upgrades (ISSUs).
proxy	Enables debugging output related to IPv4 MRIB proxy activity between the Route Processor (RP) and line cards.
route	Enables debugging output for IPv4 MRIB activity pertaining to routing entries.
table	Enables debugging output for IPv4 MRIB table management activity.
trans	Enables debugging output for activity related to IPv4 Protocol Independent Multicast (PIM) to MRIB translation.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS XE Release 2.1	This command was introduced.
15.0(1)M	This command was integrated into Cisco IOS Release 15.0(1)M.

Release	Modification
12.2(33)SRE	This command was integrated into Cisco IOS Release 12.2(33)SRE.

Examples

The following example shows how to enable debugging output for IPv4 MRIB client management activity:

```
Router# debug ip mrib client
```

debug ip mrm

To display Multicast Routing Monitor (MRM) control packet activity, use the **debug ip mrm** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip mrm [**events**| **packets**]

no debug ip mrm [**events**| **packets**]

Syntax Description

events	(Optional) Displays MRM events.
packets	(Optional) Displays MRM test packets.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.0(5)S	This command was introduced.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Examples

The following is sample output from the **debug ip mrm**command on different devices:

Examples

```
*Feb 28 16:25:44.009: MRM: Send Beacon for group 239.1.1.1, holdtime 86100 seconds
*Feb 28 16:26:01.095: MRM: Receive Status Report from 10.1.4.2 on Ethernet0
*Feb 28 16:26:01.099: MRM: Send Status Report Ack to 10.1.4.2 for group 239.1.1.1
```

Examples

```
MRM: Receive Test-Sender Request/Local trigger from 1.1.1.1 on Ethernet0
MRM: Send TS request Ack to 1.1.1.1 for group 239.1.2.3
MRM: Send test packet src:2.2.2.2 dst:239.1.2.3 manager:1.1.1.1
```

Examples

```
MRM: Receive Test-Receiver Request/Monitor from 1.1.1.1 on Ethernet0
MRM: Send TR request Ack to 1.1.1.1 for group 239.1.2.3
MRM: Receive Beacon from 1.1.1.1 on Ethernet0
MRM: Send Status Report to 1.1.1.1 for group 239.1.2.3
MRM: Receive Status Report Ack from 1.1.1.1 on Ethernet0
```

debug ip mrouting

To display information about activity in the multicast route (mroute) table, use the **debug ip mrouting** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip mrouting [**vrf** *vrf-name*] [**rpf-events**| **timers**] [*group-address*]

no debug ip mrouting [**vrf** *vrf-name*] [**rpf-events**| **timers**] [*group-address*]

Command Syntax in Cisco IOS 12.2(33)SXH and Subsequent 12.2SX Releases

debug ip mrouting [**vrf** *vrf-name*] [**high-availability**| **rpf-events** [*group-address*]| **timers** *group-address*]

no debug ip mrouting [**vrf** *vrf-name*] [**high-availability**| **rpf-events** [*group-address*]| **timers** *group-address*]

Syntax Description

vrf *vrf-name*	(Optional) Displays debugging information related to mroute activity associated with the Multicast Virtual Private Network (MVPN) routing and forwarding (MVRF) instance specified for the *vrf-name* argument.
high-availability	(Optional) Displays high availability (HA) events associated with supervisor engine switchovers on Catalyst 6500 series switches, in Cisco IOS Release 12.2(33)SXH and subsequent 12.2SX releases.
rpf-events	(Optional) Displays Reverse Path Forwarding (RPF) events associated with mroutes in the mroute table.
timers	(Optional) Displays timer-related events associated with mroutes in the mroute table.
group-address	(Optional) IP address or Domain Name System (DNS) name of a multicast group. Entering a multicast group address restricts the output to only display mroute activity associated with the multicast group address specified for the optional *group-address* argument.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
10.2	This command was introduced.
12.0(22)S	The **rpf-events** keyword was added.

Release	Modification
12.2(13)T	The **timers** keyword, **vrf** keyword, and *vrf-name* argument were added.
12.2(14)S	The **timers** keyword, **vrf** keyword, and *vrf-name* argument were added.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH. The **high-availability** keyword was added in support of the PIM Triggered Joins feature.

Usage Guidelines

This command indicates when the router has made changes to the mroute table. Use the **debug ip pim** and **debug ip mrouting** commands consecutively to obtain additional multicast routing information. In addition, use the **debug ip igmp** command to learn why an mroute message is being displayed.

This command generates a substantial amount of output. Use the optional *group-address* argument to limit the output to a single multicast group.

In Cisco IOS 12.2(33)SXH and subsequent 12.2SX releases, the **high-availability** keyword was added in support of the PIM Triggered Joins feature to monitor HA events in the event of a supervisor engine switchover on a Catalyst 6500 series switch. The PIM Triggered Joins feature is an HA multicast enhancement that improves the reconvergence of mroutes after a supervisor engine switchover on a Catalyst 6500 series switch. After a service engine switchover, all instances of PIM running on the newly active supervisor engine will modify the value of the Generation ID (GenID) that is included in PIM hello messages sent to adjacent PIM neighbors. When an adjacent PIM neighbor receives a PIM hello message on an interface with a new GenID, the PIM neighbor will interpret the modified GenID as an indication that all mroutes states on that interface have been lost. A modified GenID, thus, is utilized as a mechanism to alert all adjacent PIM neighbors that PIM forwarding on that interface has been lost, which then triggers adjacent PIM neighbors to send PIM joins for all (*, G) and (S, G) mroute states that use that interface as an RPF interface.

Examples

The following is sample output from the **debug ip mrouting** command:

```
Router# debug ip mrouting 224.2.0.1
```

```
MRT: Delete (10.0.0.0/8, 224.2.0.1)
MRT: Delete (10.4.0.0/16, 224.2.0.1)
MRT: Delete (10.6.0.0/16, 224.2.0.1)
MRT: Delete (10.9.0.0/16, 224.2.0.1)
MRT: Delete (10.16.0.0/16, 224.2.0.1)
MRT: Create (*, 224.2.0.1), if_input NULL
MRT: Create (224.69.15.0/24, 225.2.2.4), if_input Ethernet0, RPF nbr 224.69.61.15
MRT: Create (224.69.39.0/24, 225.2.2.4), if_input Ethernet1, RPF nbr 0.0.0.0
MRT: Create (10.0.0.0/8, 224.2.0.1), if_input Ethernet1, RPF nbr 224.0.0.0
MRT: Create (10.4.0.0/16, 224.2.0.1), if_input Ethernet1, RPF nbr 224.0.0.0
MRT: Create (10.6.0.0/16, 224.2.0.1), if_input Ethernet1, RPF nbr 224.0.0.0
MRT: Create (10.9.0.0/16, 224.2.0.1), if_input Ethernet1, RPF nbr 224.0.0.0
MRT: Create (10.16.0.0/16, 224.2.0.1), if_input Ethernet1, RPF nbr 224.0.0.0
```

The following lines show that multicast IP routes were deleted from the routing table:

```
MRT: Delete (10.0.0.0/8, 224.2.0.1)
```

```
MRT: Delete (10.4.0.0/16, 224.2.0.1)
MRT: Delete (10.6.0.0/16, 224.2.0.1)
```

The (*, G) entries are generally created by receipt of an Internet Group Management Protocol (IGMP) host report from a group member on the directly connected LAN or by a Protocol Independent Multicast (PIM) join message (in sparse mode) that this router receives from a router that is sending joins toward the Route Processor (RP). This router will in turn send a join toward the RP that creates the shared tree (or RP tree).

```
MRT: Create (*, 224.2.0.1), if_input NULL
```

The following lines are an example of creating an (S, G) entry that shows that an IP multicast packet (mpacket) was received on Ethernet interface 0. The second line shows a route being created for a source that is on a directly connected LAN. The RPF means "Reverse Path Forwarding," whereby the router looks up the source address of the multicast packet in the unicast routing table and determines which interface will be used to send a packet to that source.

```
MRT: Create (224.69.15.0/24, 225.2.2.4), if_input Ethernet0, RPF nbr 224.69.61.15
MRT: Create (224.69.39.0/24, 225.2.2.4), if_input Ethernet1, RPF nbr 0.0.0.0
```

The following lines show that multicast IP routes were added to the routing table. Note the 224.0.0.0 as the RPF, which means the route was created by a source that is directly connected to this router.

```
MRT: Create (10.9.0.0/16, 224.2.0.1), if_input Ethernet1, RPF nbr 224.0.0.0
MRT: Create (10.16.0.0/16, 224.2.0.1), if_input Ethernet1, RPF nbr 224.0.0.0
```

If the source is not directly connected, the neighbor address shown in these lines will be the address of the router that forwarded the packet to this router.

The shortest path tree state maintained in routers consists of source (S), multicast address (G), outgoing interface (OIF), and incoming interface (IIF). The forwarding information is referred to as the multicast forwarding entry for (S, G).

An entry for a shared tree can match packets from any source for its associated group if the packets come through the proper incoming interface as determined by the RPF lookup. Such an entry is denoted as (*, G). A (*, G) entry keeps the same information a (S, G) entry keeps, except that it saves the rendezvous point address in place of the source address in sparse mode or as 24.0.0.0 in dense mode.

The table below describes the significant fields shown in the display.

Table 34: debug ip mrouting Field Descriptions

Field	Description
MRT	Multicast route table.
RPF	Reverse Path Forwarding.
nbr	Neighbor.

Related Commands

Command	Description
debug ip dvmrp	Displays information on DVMRP packets received and sent.

Command	Description
debug ip igmp	Displays IGMP packets received and sent, and IGMP host-related events.
debug ip packet	Displays general IP debugging information and IPSO security transactions.
debug ip pim	Displays all PIM announcements received.
debug ip sd	Displays all SD announcements received.

debug ip mrouting limits

To display debugging information about configured per interface mroute state limiters and bandwidth-based multicast Call Admission Control (CAC) policies, use the **debug ip mrouting limits**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip mrouting [**vrf** *vrf-name*] **limits** [*group-address*]

no debug ip mrouting [**vrf** *vrf-name*] **limits** [*group-address*]

Syntax Description

vrf *vrf-name*	(Optional) Logs per interface mroute state limiter and bandwidth-based multicast CAC policy events related to multicast groups associated with the Multicast Virtual Private Network (VPN) routing and forwarding (MVRF) instance specified for the *vrf-name* argument.
group-address	(Optional) Multicast group address or group name for which to log per interface mroute state limiter and bandwidth-based multicast CAC policy events.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.3(14)T	This command was introduced.
12.2(33)SRB	This command was integrated into Cisco IOS Release 12.2(33)SRB.
12.2(33)SXI	This command was integrated into Cisco IOS Release 12.2(33)SXI.

Usage Guidelines

This command may generate a substantial amount of output. Use the optional *group-address* argument to restrict the output to display only per interface mroute state limiter and bandwidth-based multicast CAC policy events related to a particular multicast group.

Examples

The following output is from the **debug ip mrouting limits** command. The output displays the following events:

- An mroute state being created and the corresponding per interface mroute state limiter counter being increased by the default cost of 1 on incoming Ethernet interface 1/0.
- An mroute olist member being removed from the olist and the corresponding per interface mroute limiter being decreased by the default cost of 1 on outgoing Ethernet interface 1/0.

- An mroute being denied by the per interface mroute state limiter because the maximum number of mroute states has been reached.
- An mroute state being created and the corresponding per interface mroute state limiter counter being increased by the cost of 2 on incoming Ethernet interface 1/0.
- An mroute olist member being removed from the olist and the corresponding per interface mroute limiter being decreased by a cost of 2 on outgoing Ethernet interface 1/0.

```
Router# debug ip mrouting limits

MRL(0): incr-ed acl ‘rpf-list’ to (13 < max 32), [n:0,p:0], (main) GigabitEthernet0/0,
(10.41.0.41, 225.30.200.60)
MRL(0): decr-ed acl ‘out-list’ to (10 < max 32), [n:0,p:0], (main) GigabitEthernet0/0, (*,
225.40.202.60)
MRL(0): Add mroute (10.43.0.43, 225.30.200.60) denied for GigabitEthernet0/2, acl std-list,
 (16 = max 16)
MRL(0): incr-ed limit-acl `rpf-list' to (12 < max 32), cost-acl 'cost-list' cost 2, [n:0,p:0],
 (main) GigabitEthernet0/0, (10.41.0.41, 225.30.200.60)
MRL(0): decr-ed limit-acl `out-list' to (8 < max 32), cost-acl 'cost-list'' cost 2, [n:0,p:0],
 (main) GigabitEthernet0/0, (*, 225.40.202.60)
```

Related Commands

Command	Description
clear ip multicast limit	Resets the exceeded counter for per interface mroute state limiters.
ip multicast limit	Configures per interface mroute state limiters.
ip multicast limit cost	Applies costs to per interface mroutes state limiters.
show ip multicast limit	Displays statistics about configured per interface mroute state limiters.

debug ip msdp

To debug Multicast Source Discovery Protocol (MSDP) activity, use the **debug ip msdp**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip msdp [**vrf** *vrf-name*] [*peer-address*| *name*] **[detail] [routes]**

no debug ip msdp [**vrf** *vrf-name*] [*peer-address*| *name*] **[detail] [routes]**

Syntax Description

vrf	(Optional) Supports the Multicast Virtual Private Network (VPN) routing and forwarding (VRF) instance.
vrf-name	(Optional) Name assigned to the VRF.
peer-address | *name*	(Optional) The peer for which debug events are logged.
detail	(Optional) Provides more detailed debugging information.
routes	(Optional) Displays the contents of Source-Active messages.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(7)T	This command was introduced.
12.0(23)S	The **vrf** keyword and *vrf-name* argument were added.
12.2(13)T	This command was integrated into Cisco IOS Release 12.2(13)T.
12.2(14)S	This command was integrated into Cisco IOS Release 12.2(14)S.
12.2(27)SBC	This command was integrated into Cisco IOS Release 12.2(27)SBC.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Examples

The following is sample output from the **debug ip msdp** command:

```
Router# debug ip msdp
```

```
MSDP debugging is on
Router#
MSDP: 224.150.44.254: Received 1388-byte message from peer
MSDP: 224.150.44.254: SA TLV, len: 1388, ec: 115, RP: 172.31.3.92
MSDP: 224.150.44.254: Peer RPF check passed for 172.31.3.92, used EMBGP peer
MSDP: 224.150.44.250: Forward 1388-byte SA to peer
MSDP: 224.150.44.254: Received 1028-byte message from peer
MSDP: 224.150.44.254: SA TLV, len: 1028, ec: 85, RP: 172.31.3.92
MSDP: 224.150.44.254: Peer RPF check passed for 172.31.3.92, used EMBGP peer
MSDP: 224.150.44.250: Forward 1028-byte SA to peer
MSDP: 224.150.44.254: Received 1388-byte message from peer
MSDP: 224.150.44.254: SA TLV, len: 1388, ec: 115, RP: 172.31.3.111
MSDP: 224.150.44.254: Peer RPF check passed for 172.31.3.111, used EMBGP peer
MSDP: 224.150.44.250: Forward 1388-byte SA to peer
MSDP: 224.150.44.250: Received 56-byte message from peer
MSDP: 224.150.44.250: SA TLV, len: 56, ec: 4, RP: 205.167.76.241
MSDP: 224.150.44.250: Peer RPF check passed for 205.167.76.241, used EMBGP peer
MSDP: 224.150.44.254: Forward 56-byte SA to peer
MSDP: 224.150.44.254: Received 116-byte message from peer
MSDP: 224.150.44.254: SA TLV, len: 116, ec: 9, RP: 172.31.3.111
MSDP: 224.150.44.254: Peer RPF check passed for 172.31.3.111, used EMBGP peer
MSDP: 224.150.44.250: Forward 116-byte SA to peer
MSDP: 224.150.44.254: Received 32-byte message from peer
MSDP: 224.150.44.254: SA TLV, len: 32, ec: 2, RP: 172.31.3.78
MSDP: 224.150.44.254: Peer RPF check passed for 172.31.3.78, used EMBGP peer
MSDP: 224.150.44.250: Forward 32-byte SA to peer
```

The table below describes the significant fields shown in the display.

Table 35: debug ip msdp Field Descriptions

Field	Description
MSDP	Protocol being debugged.
224.150.44.254:	IP address of the MSDP peer.
Received 1388-byte message from peer	MSDP event.

debug ip msdp resets

To debug Multicast Source Discovery Protocol (MSDP) peer reset reasons, use the **debug ip msdp resets** command in privileged EXEC mode.

debug ip msdp [**vrf** *vrf-name*] **resets**

Syntax Description

vrf	(Optional) Supports the Multicast Virtual Private Network (VPN) routing and forwarding (VRF) instance.
vrf-name	(Optional) Name assigned to the VRF.

Command Default

No default behavior or values.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(7)T	This command was introduced.
12.0(23)S	The **vrf** keyword and *vrf-name* argument were added.
12.2(13)T	This command was integrated into Cisco IOS Release 12.2(13)T.
12.2(14)S	This command was integrated into Cisco IOS Release 12.2(14)S.
12.2(27)SBC	This command was integrated into Cisco IOS Release 12.2(27)SBC.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

debug ip multicast hardware-switching

To display information about multicast hardware switching, use the **debug ip multicast hardware-switching**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip multicast hardware-switching {**control group-name**| **error A**.**B**.**C**.**D**| **event A**.**B**.**C**.**D**| **ha-error A**.**B**.**C**.**D**| **ha-event A**.**B**.**C**.**D**}

no debug ip multicast hardware-switching {**control group-name**| **error A**.**B**.**C**.**D**| **event A**.**B**.**C**.**D**| **ha-error A**.**B**.**C**.**D**| **ha-event A**.**B**.**C**.**D**}

Syntax Description

control	Displays all multicast hardware switching debugging information, including errors, events, and packets for the specified group.
group-name	Specifies the selected group.
A.B.C.D	Specifies the source or group I.D. address.
error	Displays error messages related to multicast hardware switching.
event	Displays the run-time sequence of events for multicast hardware switching.
ha-error	Displays the run-time sequence of ha-errors for multicast hardware switching.
ha-event	Displays the run-time sequence of ha-events for multicast hardware switching.

Command Default

Debugging is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(33)SRE	This command was introduced on Cisco 7600 series routers.

Usage Guidelines

Only one of the keywords is required.

Examples

The following example shows output from the **debug ip multicast hardware-switching**command using the **error** keyword:

```
Router# debug ip multicast hardware-switching error 232.0.1.4
PE1-7600#debug ip multicast hardware-switching error 232.0.1.4
CMFIB-RP IPv4  error debugging enabled for group 232.0.1.4
PE1-7600#
```

The following example shows output from the **debug ip multicast hardware-switching**command using the **event** keyword:

```
Router# debug ip multicast hardware-switching event 232.0.1.4
CMFIB-RP IPv4  event debugging enabled for group 232.0.1.4
Router#
```

The following example shows output from the **debug ip multicast hardware-switching** command using the **ha-event** keyword:

```
Router# debug ip multicast hardware-switching ha-event 232.0.1.4
CMFIB-RP IPv4  ha event debugging enabled for group 232.0.1.4
PE1-7600#
Router#
Router#
The following example shows output from the debug ip multicast hardware-switching
 command using the ha-error
 keyword:
Router# debug ip multicast hardware-switching ha-error 232.0.1.4
CMFIB-RP IPv4  ha error debugging enabled for group 232.0.1.4
Router#
```

Related Commands

Command	Description
ipv6 multicast hardware-switching connected	Downloads the interface and mask entry for IPv6 multicast packet.

debug ip multicast redundancy

To display information about IP multicast redundancy events, use the **debug ip multicast redundancy**command in privileged EXEC mode. To disable debugging output for IP multicast redundancy events, use the **no** form of this command.

debug ip multicast [**default-vrf**| **vrf** *vrf-name*] [**group** *group-address*] **redundancy [verbose]**

no debug ip multicast [**default-vrf**| **vrf** *vrf-name*] [**group** *group-address*] **redundancy [verbose]**

Syntax Description

default-vrf	(Optional) Restricts the logging of IP multicast events associated with Multicast Virtual Private Network routing and forwarding (MVRF) instances to events associated with the default MVRF.
vrf *vrf-name*	(Optional) Restricts the logging of IP multicast events associated with MVRFs to events associated with the MVRF specified for the *vrf-name* argument.
group *group-address*	(Optional) Restricts the output for multicast groups to events associated with the multicast group specified for the *group-address*argument.
verbose	(Optional) Logs events that may occur frequently during normal operation, but that may be useful for tracking in short intervals.

Command Default

IP multicast events related to all multicast groups and all MVRFs are displayed. Logging events enabled with the **verbose** keyword are not displayed.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(33)SXI	This command was introduced.
12.2(33)SRE	This command was integrated into Cisco IOS Release 12.2(33)SRE.
Cisco IOS XE Release 2.6	This command was integrated into Cisco IOS XE Release 2.6.
15.0(1)S	This command was integrated into Cisco IOS Release 15.0(1)S.

Usage Guidelines

Use this command to display IP multicast redundancy events.

This command logs events that are important in verifying nonstop forwarding (NSF) with stateful switchover (SSO) for IP multicast. The classes of events logged by the **debug ip multicast redundancy** command include stateful switchover events during a Route Processor (RP) switchover and dynamic synchronization events that occur during steady state operation.

Use the optional **verbose** keyword to log events that may occur frequently during normal operation, but that may be useful for tracking in short intervals.

Examples

The following sample output from the **debug ip multicast redundancy** command shows the initial logging messages that display when the system detects an RP switchover:

```
00:10:33: %REDUNDANCY-3-SWITCHOVER: RP switchover (PEER_DOWN_INTERRUPT)
00:10:33: %REDUNDANCY-5-PEER_MONITOR_EVENT: Standby received a switchover
(raw-event=PEER_DOWN_INTERRUPT(11))
*Aug  7 02:31:28.051: MCAST-HA: Received cf status CHKPT_STATUS_PEER_NOT_READY
*Aug  7 02:31:28.063: MCAST-HA: Received cf status CHKPT_STATUS_PEER_NOT_READY
*Aug  7 02:31:28.063: MCAST-HA-RF: Status event: status=RF_STATUS_PEER_COMM  Op=0
RFState=STANDBY HOT
*Aug  7 02:31:28.063: MCAST-HA-RF: Status event: status=RF_STATUS_OPER_REDUNDANCY_MODE_CHANGE
  Op=0  RFState=STANDBY HOT
*Aug  7 02:31:28.063: MCAST-HA-RF: Status event: status=RF_STATUS_REDUNDANCY_MODE_CHANGE
Op=0  RFState=STANDBY HOT
*Aug  7 02:31:28.063: MCAST-HA-RF: Status event: status=RF_STATUS_PEER_PRESENCE  Op=0
RFState=STANDBY HOT
*Aug  7 02:31:28.063: MCAST-HA-RF: Status event: status=RF_STATUS_MAINTENANCE_ENABLE  Op=0
  RFState=ACTIVE-FAST
*Aug  7 02:31:28.063: MCAST-HA-RF: Progression event: RF_Event=RF_PROG_ACTIVE_FAST
RFState=ACTIVE-FAST
*Aug  7 02:31:28.091: MCAST-HA-RF: Progression event: RF_Event=RF_PROG_ACTIVE_DRAIN
RFState=ACTIVE-DRAIN
*Aug  7 02:31:28.091: MCAST-HA-RF: Progression event: RF_Event=RF_PROG_ACTIVE_PRECONFIG
RFState=ACTIVE_PRECONFIG
*Aug  7 02:31:28.091: MCAST-HA-RF: Progression event: RF_Event=RF_PROG_ACTIVE_POSTCONFIG
RFState=ACTIVE_POSTCONFIG
*Aug  7 02:31:28.103: MCAST-HA: Received cf status CHKPT_STATUS_IPC_FLOW_ON
*Aug  7 02:31:28.103: MCAST-HA-RF: Progression event: RF_Event=RF_PROG_ACTIVE RFState=ACTIVE
```

The following is sample output from the **debug ip multicast redundancy** command. As interfaces come up on the new active RP, unicast convergence occurs in parallel with a multicast route refresh from Protocol Independent Multicast (PIM) neighbors. Unicast convergence is followed by Reverse Path Forwarding (RPF) adjustments to the refreshed mroute information.

```
*Aug  7 02:31:28.107: MCAST-HA: Triggering unicast convergence notification process handling
 for MVRF IPv4 default
*Aug  7 02:31:28.107: MCAST-HA: Triggering unicast convergence notification process handling
 for MVRF blue
*Aug  7 02:31:28.107: MCAST-HA: Triggering unicast convergence notification process handling
 for MVRF green
*Aug  7 02:31:28.107: MCAST-HA: Triggering unicast convergence notification process handling
 for MVRF red
*Aug  7 02:31:28.107: MCAST-HA: Triggering unicast convergence notification process handling
 for all MVRFs
*Aug  7 02:31:28.111: MCAST-HA: Beginning unicast convergence notification process handling.
*Aug  7 02:31:28.111: MCAST-HA: Unicast convergence completed for MVRF IPv4 default:
Triggering RPF updates
*Aug  7 02:31:28.111: MCAST-HA: Beginning unicast convergence notification process handling.
*Aug  7 02:31:28.111: MCAST-HA: Unicast convergence completed for MVRF blue:  Triggering
RPF updates
*Aug  7 02:31:28.111: MCAST-HA: Beginning unicast convergence notification process handling.
*Aug  7 02:31:28.111: MCAST-HA: Unicast convergence completed for MVRF green:  Triggering
RPF updates
```

```
*Aug  7 02:31:28.111: MCAST-HA: Beginning unicast convergence notification process handling.
*Aug  7 02:31:28.111: MCAST-HA: Unicast convergence completed for MVRF red:  Triggering RPF
 updates
*Aug  7 02:31:28.111: MCAST-HA: Unicast convergence notification has been received for the
 only unconverged VRF.
Stopping the unicast routing convergence failsafe timer.
*Aug  7 02:31:28.111: MCAST-HA: Beginning unicast convergence notification process handling.
*Aug  7 02:31:28.111: MCAST-HA: Unicast convergence notification received for the wildcard
 tableid (all VRFs).
Triggering RPF updates for all MVRFs and stopping the unicast IGP convergence failsafe
timer.
00:10:34: %PIM-5-DRCHG: DR change from neighbor 0.0.0.0 to 172.16.1.1 on interface Loopback0
00:10:34: %PIM-5-DRCHG: DR change from neighbor 0.0.0.0 to 172.31.10.1 on interface Loopback1
00:10:35: %PIM-5-DRCHG: VRF green: DR change from neighbor 0.0.0.0 to 172.16.1.1 on interface
 Tunnel1
00:10:35: %PIM-5-DRCHG: VRF red: DR change from neighbor 0.0.0.0 to 172.16.1.1 on interface
 Tunnel2
00:10:35: %LINK-3-UPDOWN: Interface Null0, changed state to up
00:10:35: %LINK-3-UPDOWN: Interface Loopback0, changed state to up
00:10:35: %LINK-3-UPDOWN: Interface Loopback1, changed state to up
00:10:35: %LINK-3-UPDOWN: Interface Tunnel0, changed state to up
00:10:35: %LINK-3-UPDOWN: Interface Tunnel1, changed state to up
00:10:35: %LINK-3-UPDOWN: Interface Tunnel2, changed state to up
00:10:35: %LINK-5-CHANGED: Interface Ethernet0/0, changed state to administratively down
00:10:35: %LINK-5-CHANGED: Interface Ethernet0/1, changed state to administratively down
00:10:35: %LINK-5-CHANGED: Interface Ethernet0/2, changed state to administratively down
00:10:35: %LINK-5-CHANGED: Interface Ethernet0/3, changed state to administratively down
00:10:35: %LINK-5-CHANGED: Interface Ethernet1/0, changed state to administratively down
00:10:35: %LINK-5-CHANGED: Interface Ethernet1/1, changed state to administratively down
00:10:35: %LINK-5-CHANGED: Interface Ethernet1/2, changed state to administratively down
00:10:35: %LINK-5-CHANGED: Interface Ethernet1/3, changed state to administratively down
00:10:36: %LINEPROTO-5-UPDOWN: Line protocol on Interface Null0, changed state to up
00:10:36: %LINEPROTO-5-UPDOWN: Line protocol on Interface Loopback0, changed state to up
00:10:36: %LINEPROTO-5-UPDOWN: Line protocol on Interface Loopback1, changed state to up
00:10:36: %LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel0, changed state to up
00:10:36: %LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel1, changed state to up
00:10:36: %LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel2, changed state to up
00:10:38: %PIM-5-DRCHG: VRF blue: DR change from neighbor 0.0.0.0 to 172.16.1.1 on interface
 Tunnel0
```

The following is sample output from the **debug ip multicast redundancy** command. After the processing of unicast and multicast route convergence, time is allowed for Internet Group Management Protocol (IGMP) reporting. Following IGMP reporting, the control plane then sends out requests for the Multicast Forwarding Information Base (MFIB) replay of data driven events (DDEs) to retrigger multicast route information that cannot be obtained from PIM neighbors or directly connected hosts. After this processing completes, the control plane waits for the NSF hold-off time period to terminate. The refreshed multicast control plane information is then downloaded to the forwarding plane; once the download is completed, the stale multicast forwarding plane information is subsequently flushed.

```
*Aug  7 02:31:43.651: MCAST-HA: IGMP response timer expired. Ready for DDE replay for MVRF
 red
*Aug  7 02:31:43.651: MCAST-HA: Sending DDE replay request for MVRF red.
*Aug  7 02:31:43.651: MCAST-HA: MFIB DDE replay completed for mvrf red
*Aug  7 02:31:43.651: MCAST-HA: No NSF Holdoff extension requested for mvrf red at completion
 of DDE replay.
*Aug  7 02:31:43.651: MCAST-HA: Terminating multicast NSF holdoff for MVRF red
*Aug  7 02:31:43.651: MCAST-HA: Still awaiting MFIB DDE replay for mvrf green
 DDE replay: NOT COMPLETED, MRIB update: NOT PENDING
*Aug  7 02:31:43.651: MCAST-HA: IGMP response timer expired. Ready for DDE replay for MVRF
 green
*Aug  7 02:31:43.651: MCAST-HA: Sending DDE replay request for MVRF green.
*Aug  7 02:31:43.651: MCAST-HA: MFIB DDE replay completed for mvrf green
*Aug  7 02:31:43.651: MCAST-HA: No NSF Holdoff extension requested for mvrf green at
completion of DDE replay.
*Aug  7 02:31:43.651: MCAST-HA: Terminating multicast NSF holdoff for MVRF green
*Aug  7 02:31:43.651: MCAST-HA: Still awaiting MFIB DDE replay for mvrf blue
 DDE replay: NOT COMPLETED, MRIB update: NOT PENDING
*Aug  7 02:31:43.651: MCAST-HA: IGMP response timer expired. Ready for DDE replay for MVRF
 blue
*Aug  7 02:31:43.651: MCAST-HA: Sending DDE replay request for MVRF blue.
```

```
*Aug  7 02:31:43.651: MCAST-HA: MFIB DDE replay completed for mvrf blue
*Aug  7 02:31:43.651: MCAST-HA: No NSF Holdoff extension requested for mvrf blue at completion
 of DDE replay.
*Aug  7 02:31:43.651: MCAST-HA: Terminating multicast NSF holdoff for MVRF blue
*Aug  7 02:31:43.651: MCAST-HA: Still awaiting MFIB DDE replay for mvrf IPv4 default
 DDE replay: NOT COMPLETED, MRIB update: NOT PENDING
*Aug  7 02:31:43.651: MCAST-HA: IGMP response timer expired. Ready for DDE replay for MVRF
 IPv4 default
*Aug  7 02:31:43.651: MCAST-HA: Sending DDE replay request for MVRF IPv4 default.
*Aug  7 02:31:43.651: MCAST-HA: MFIB DDE replay completed for mvrf IPv4 default
*Aug  7 02:31:43.651: MCAST-HA: No NSF Holdoff extension requested for mvrf IPv4 default
at completion of DDE replay.
*Aug  7 02:31:43.651: MCAST-HA: Terminating multicast NSF holdoff for MVRF IPv4 default
*Aug  7 02:31:43.651: MCAST-HA: MFIB DDE replay completed for all MVRFs.
*Aug  7 02:31:43.651: MCAST-HA: Stopping the MFIB DDE replay failsafe timer.
*Aug  7 02:32:13.651: MCAST-HA: Flush timer expired. Starting final RPF check for MVRF IPv4
 default
*Aug  7 02:32:13.651: MCAST-HA: Flush timer expired. Starting final RPF check for MVRF blue
*Aug  7 02:32:13.651: MCAST-HA: Flush timer expired. Starting final RPF check for MVRF green
*Aug  7 02:32:13.651: MCAST-HA: Flush timer expired. Starting final RPF check for MVRF red
*Aug  7 02:32:14.151: MCAST-HA: Flushing stale mcast state. RP failover processing complete
 for MVRF IPv4 default.
*Aug  7 02:32:14.151: MCAST-HA: Flushing stale mcast state. RP failover processing complete
 for MVRF blue.
*Aug  7 02:32:14.151: MCAST-HA: Flushing stale mcast state. RP failover processing complete
 for MVRF green.
*Aug  7 02:32:14.151: MCAST-HA: Flushing stale mcast state. RP failover processing complete
 for MVRF red.
*Aug  7 02:32:14.151: MCAST-HA: RP failover processing complete for all MVRFs.
```

The following is sample output from the **debug ip multicast redundancy** command. This output shows the events related to the reloading of the standby RP, in particular, ISSU negotiation between the active and standby RP and synchronization of dynamic multicast forwarding information from the active RP to the standby RP. Synchronization events are also logged in steady state for events that occur that affect dynamic group-to-RP mapping information or dynamic tunnel state.

```
00:11:50: %HA-6-MODE: Operating RP redundancy mode is SSO
*Aug  7 02:32:45.435: MCAST-HA-RF: Status event: status=RF_STATUS_OPER_REDUNDANCY_MODE_CHANGE
  Op=7  RFState=ACTIVE
*Aug  7 02:32:45.435: MCAST-HA-RF: Status event: status=RF_STATUS_REDUNDANCY_MODE_CHANGE
Op=7  RFState=ACTIVE
*Aug  7 02:32:45.435: MCAST-HA-RF: Status event: status=RF_STATUS_PEER_PRESENCE  Op=1
RFState=ACTIVE
*Aug  7 02:32:45.463: MCAST-HA-RF: Status event: status=RF_STATUS_PEER_COMM  Op=1
RFState=ACTIVE
*Aug  7 02:32:45.563: MCAST-HA-RF: Progression event: RF_Event=RF_PROG_ISSU_NEGOTIATION
RFState=ACTIVE
*Aug  7 02:32:46.039: MCAST-HA-RF: Progression event: RF_Event=RF_PROG_PLATFORM_SYNC
RFState=ACTIVE
*Aug  7 02:32:46.979: MCAST-HA: Received cf status CHKPT_STATUS_PEER_READY
*Aug  7 02:32:46.979: MCAST-ISSU Handling communication up transition for PIM HA transport
 type 0, RF comm = TRUE, renegotiation NOT PENDING
*Aug  7 02:32:46.979: MCAST-HA: Received cf status CHKPT_STATUS_IPC_FLOW_ON
*Aug  7 02:32:47.043: MCAST-HA-RF: Progression event:
RF_Event=RF_PROG_STANDBY_ISSU_NEGOTIATION_LATE RFState=ACTIVE
*Aug  7 02:32:50.943: MCAST-HA-RF: Progression event: RF_Event=RF_PROG_STANDBY_CONFIG
RFState=ACTIVE
*Aug  7 02:32:50.947: MCAST-ISSU Negotiation message sent from primary, rc = 0
*Aug  7 02:32:50.947: MCAST-HA-RF: Started PIM ISSU negotiation on the primary RP.
*Aug  7 02:32:50.947: MCAST-ISSU Negotiation message sent from primary, rc = 0
*Aug  7 02:32:50.947: MCAST-ISSU Negotiation message sent from primary, rc = 0
*Aug  7 02:32:50.951: MCAST-ISSU Negotiation message sent from primary, rc = 0
*Aug  7 02:32:50.951: MCAST-ISSU Negotiation message sent from primary, rc = 0
*Aug  7 02:32:50.951: MCAST-ISSU Negotiation message sent from primary, rc = 0
*Aug  7 02:32:50.951: MCAST-ISSU Negotiation message sent from primary, rc = 0
*Aug  7 02:32:50.955: MCAST-ISSU Negotiation message sent from primary, rc = 0
*Aug  7 02:32:50.955: MCAST-ISSU Negotiation message sent from primary, rc = 0
*Aug  7 02:32:50.955: MCAST-ISSU Negotiation message sent from primary, rc = 0
*Aug  7 02:32:50.955: MCAST-ISSU Negotiation message sent from primary, rc = 0
*Aug  7 02:32:50.959: MCAST-ISSU Negotiation message sent from primary, rc = 0
*Aug  7 02:32:50.959: MCAST-ISSU Negotiation message sent from primary, rc = 0
```

```
*Aug  7 02:32:50.959: MCAST-ISSU Negotiation message sent from primary, rc = 0
*Aug  7 02:32:50.959: MCAST-ISSU Negotiation message sent from primary, rc = 0
*Aug  7 02:32:50.959: MCAST-ISSU Negotiation message sent from primary, rc = 0
*Aug  7 02:32:50.963: MCAST-ISSU Negotiation message sent from primary, rc = 0
*Aug  7 02:32:50.963: MCAST-ISSU Negotiation message sent from primary, rc = 0
*Aug  7 02:32:50.963: MCAST-ISSU Negotiation message sent from primary, rc = 0
*Aug  7 02:32:50.963: MCAST-ISSU Negotiation message sent from primary, rc = 0
*Aug  7 02:32:50.967: MCAST-ISSU Negotiation message sent from primary, rc = 0
*Aug  7 02:32:50.967: MCAST-ISSU Negotiation message sent from primary, rc = 0
*Aug  7 02:32:50.967: MCAST-ISSU Negotiation message sent from primary, rc = 0
*Aug  7 02:32:50.967: MCAST-ISSU Negotiation message sent from primary, rc = 0
*Aug  7 02:32:50.967: MCAST-ISSU Negotiation message sent from primary, rc = 0
*Aug  7 02:32:50.971: MCAST-ISSU Negotiation message sent from primary, rc = 0
*Aug  7 02:32:50.971: MCAST-ISSU Negotiation message sent from primary, rc = 0
*Aug  7 02:32:50.971: MCAST-ISSU Negotiation completed for PIM Checkpoint Facility client,
 negotiation rc = 4, negotiation result = COMPATIBLE
*Aug  7 02:32:59.927: MCAST-HA-RF: Progression event: RF_Event=RF_PROG_STANDBY_FILESYS
RFState=ACTIVE
*Aug  7 02:32:59.963: MCAST-HA-RF: Progression event: RF_Event=RF_PROG_STANDBY_BULK
RFState=ACTIVE
*Aug  7 02:32:59.963: MCAST-HA-RF: Starting Bulk Sync.
*Aug  7 02:32:59.963: MCAST-HA: Successfully created the bulk sync process
*Aug  7 02:32:59.963: MCAST-HA: Starting Bulk sync
*Aug  7 02:32:59.963: MCAST HA Executing RP mapping bulk sync.
*Aug  7 02:32:59.963: MCAST HA Executing Bidir RP route bulk sync.
*Aug  7 02:32:59.963: MCAST HA Executing BSR cache bulk sync.
*Aug  7 02:32:59.963: MCAST-HA BSR cache sync request received for mvrf IPv4 default
*Aug  7 02:32:59.963: MCAST-HA: Creating Bootstrap cache sync request chunk size=112 max=585
 align=8
*Aug  7 02:32:59.963: MCAST-HA: Allocating Bootstrap cache sync request sync request
*Aug  7 02:32:59.963: MCAST-HA Formatting BSR cache sync message:
search for mvrf IPv4 default result is 0 mvrf at 0x4A21680
*Aug  7 02:32:59.971: MCAST-HA BSR cache sync request received for mvrf blue
*Aug  7 02:32:59.971: MCAST-HA: Allocating Bootstrap cache sync request sync request
*Aug  7 02:32:59.971: MCAST-HA Formatting BSR cache sync message:
search for mvrf blue result is 0 mvrf at 0x50EE660
*Aug  7 02:32:59.983: MCAST-HA BSR cache sync request received for mvrf green
*Aug  7 02:32:59.983: MCAST-HA: Allocating Bootstrap cache sync request sync request
*Aug  7 02:32:59.983: MCAST-HA Formatting BSR cache sync message:
search for mvrf green result is 0 mvrf at 0x5103300
*Aug  7 02:32:59.991: MCAST-HA BSR cache sync request received for mvrf red
*Aug  7 02:32:59.991: MCAST-HA: Allocating Bootstrap cache sync request sync request
*Aug  7 02:32:59.991: MCAST-HA Formatting BSR cache sync message:
search for mvrf red result is 0 mvrf at 0x5135FE0
*Aug  7 02:33:00.003: MCAST HA Executing AutoRP discovery IDB bulk sync.
*Aug  7 02:33:00.003: MCAST-HA AutoRP discovery IDB sync request received for
mvrf IPv4 default
*Aug  7 02:33:00.003: MCAST-HA: Creating Autorp discovery IDB sync request chunk size=112
max=585 align=8
*Aug  7 02:33:00.003: MCAST-HA: Allocating Autorp discovery IDB sync request sync request
*Aug  7 02:33:00.003: MCAST-HA Formatting AutoRP discovery IDB sync message:
search for mvrf IPv4 default result is 0 mvrf at 0x4A21680
*Aug  7 02:33:00.011: MCAST-HA AutoRP discovery IDB sync request received for
mvrf blue
*Aug  7 02:33:00.011: MCAST-HA: Allocating Autorp discovery IDB sync request sync request
*Aug  7 02:33:00.011: MCAST-HA Formatting AutoRP discovery IDB sync message:
search for mvrf blue result is 0 mvrf at 0x50EE660
*Aug  7 02:33:00.023: MCAST-HA AutoRP discovery IDB sync request received for
mvrf green
*Aug  7 02:33:00.023: MCAST-HA: Allocating Autorp discovery IDB sync request sync request
*Aug  7 02:33:00.023: MCAST-HA Formatting AutoRP discovery IDB sync message:
search for mvrf green result is 0 mvrf at 0x5103300
*Aug  7 02:33:00.031: MCAST-HA AutoRP discovery IDB sync request received for
mvrf red
*Aug  7 02:33:00.031: MCAST-HA: Allocating Autorp discovery IDB sync request sync request
*Aug  7 02:33:00.031: MCAST-HA Formatting AutoRP discovery IDB sync message:
search for mvrf red result is 0 mvrf at 0x5135FE0
*Aug  7 02:33:00.043: MCAST HA Executing dummy bulk sync function.
*Aug  7 02:33:00.043: MCAST HA Executing dummy bulk sync function.
*Aug  7 02:33:00.043: MCAST HA Executing dummy bulk sync function.
*Aug  7 02:33:00.043: MCAST HA Executing MDT tunnel bulk sync.
*Aug  7 02:33:00.043: MCAST-HA MDT tunnel sync request received for mvrf blue
*Aug  7 02:33:00.043: MCAST-HA: Creating MDT tunnel sync request chunk size=112 max=585
```

```
align=8
*Aug  7 02:33:00.043: MCAST-HA: Allocating MDT tunnel sync request sync request
*Aug  7 02:33:00.043: MCAST-HA Formatting MDT tunnel sync message:
search for mvrf blue result is 0 mvrf at 0x50EE660
*Aug  7 02:33:00.051: MCAST-HA MDT tunnel sync request received for mvrf green
*Aug  7 02:33:00.051: MCAST-HA: Allocating MDT tunnel sync request sync request
*Aug  7 02:33:00.051: MCAST-HA Formatting MDT tunnel sync message:
search for mvrf green result is 0 mvrf at 0x5103300
*Aug  7 02:33:00.063: MCAST-HA MDT tunnel sync request received for mvrf red
*Aug  7 02:33:00.063: MCAST-HA: Allocating MDT tunnel sync request sync request
*Aug  7 02:33:00.063: MCAST-HA Formatting MDT tunnel sync message:
search for mvrf red result is 0 mvrf at 0x5135FE0
*Aug  7 02:33:00.071: MCAST HA Executing Bidir RP DF bulk sync.
*Aug  7 02:33:00.071: MCAST HA Executing register tunnel bulk sync.
*Aug  7 02:33:00.071: MCAST-HA: Completed enqueuing of bulk sync messages.
*Aug  7 02:33:00.071: MCAST-HA: Bulk sync message queue has drained.
*Aug  7 02:33:00.071: MCAST-HA: Received acknowledgement from standby for all bulk sync
messages.
*Aug  7 02:33:00.071: MCAST-HA Creating bulk sync completion message for peer.
*Aug  7 02:33:00.071: MCAST-HA: Primary has notified standby of bulk sync completion. Waiting
 for final bulk sync ACK from stby.
*Aug  7 02:33:00.075: MCAST-HA: Received cf status CHKPT_STATUS_SEND_OK
*Aug  7 02:33:00.075: MCAST-HA: Sent message type is 2
*Aug  7 02:33:00.075: MCAST-HA Searching for sync request corresponding to the successfully
 received message.
*Aug  7 02:33:00.075: MCAST-HA Transmission from primary and reception by standby confirmed
 for sync type 2. Cleanup is complete.
*Aug  7 02:33:00.075: MCAST-HA: Received cf status CHKPT_STATUS_SEND_OK
*Aug  7 02:33:00.075: MCAST-HA: Sent message type is 2
*Aug  7 02:33:00.075: MCAST-HA Searching for sync request corresponding to the successfully
 received message.
*Aug  7 02:33:00.075: MCAST-HA Transmission from primary and reception by standby confirmed
 for sync type 2. Cleanup is complete.
*Aug  7 02:33:00.075: MCAST-HA: Received cf status CHKPT_STATUS_SEND_OK
*Aug  7 02:33:00.075: MCAST-HA: Sent message type is 2
*Aug  7 02:33:00.075: MCAST-HA Searching for sync request corresponding to the successfully
 received message.
*Aug  7 02:33:00.075: MCAST-HA Transmission from primary and reception by standby confirmed
 for sync type 2. Cleanup is complete.
*Aug  7 02:33:00.087: MCAST-HA: Received cf status CHKPT_STATUS_SEND_OK
*Aug  7 02:33:00.087: MCAST-HA: Sent message type is 2
*Aug  7 02:33:00.087: MCAST-HA Searching for sync request corresponding to the successfully
 received message.
*Aug  7 02:33:00.087: MCAST-HA Transmission from primary and reception by standby confirmed
 for sync type 2. Cleanup is complete.
*Aug  7 02:33:00.087: MCAST-HA: Received cf status CHKPT_STATUS_SEND_OK
*Aug  7 02:33:00.087: MCAST-HA: Sent message type is 3
*Aug  7 02:33:00.087: MCAST-HA Searching for sync request corresponding to the successfully
 received message.
*Aug  7 02:33:00.087: MCAST-HA Transmission from primary and reception by standby confirmed
 for sync type 3. Cleanup is complete.
*Aug  7 02:33:00.087: MCAST-HA: Received cf status CHKPT_STATUS_SEND_OK
*Aug  7 02:33:00.087: MCAST-HA: Sent message type is 3
*Aug  7 02:33:00.087: MCAST-HA Searching for sync request corresponding to the successfully
 received message.
*Aug  7 02:33:00.087: MCAST-HA Transmission from primary and reception by standby confirmed
 for sync type 3. Cleanup is complete.
*Aug  7 02:33:00.087: MCAST-HA: Received cf status CHKPT_STATUS_SEND_OK
*Aug  7 02:33:00.087: MCAST-HA: Sent message type is 3
*Aug  7 02:33:00.087: MCAST-HA Searching for sync request corresponding to the successfully
 received message.
*Aug  7 02:33:00.087: MCAST-HA Transmission from primary and reception by standby confirmed
 for sync type 3. Cleanup is complete.
*Aug  7 02:33:00.087: MCAST-HA: Received cf status CHKPT_STATUS_SEND_OK
*Aug  7 02:33:00.087: MCAST-HA: Sent message type is 3
*Aug  7 02:33:00.087: MCAST-HA Searching for sync request corresponding to the successfully
 received message.
*Aug  7 02:33:00.087: MCAST-HA Transmission from primary and reception by standby confirmed
 for sync type 3. Cleanup is complete.
*Aug  7 02:33:00.087: MCAST-HA: Received cf status CHKPT_STATUS_SEND_OK
*Aug  7 02:33:00.087: MCAST-HA: Sent message type is 8
*Aug  7 02:33:00.087: MCAST-HA Searching for sync request corresponding to the successfully
 received message.
```

```
*Aug  7 02:33:00.087: MCAST-HA Transmission from primary and reception by standby confirmed
 for sync type 8. Cleanup is complete.
*Aug  7 02:33:00.087: MCAST-HA: Received cf status CHKPT_STATUS_SEND_OK
*Aug  7 02:33:00.087: MCAST-HA: Sent message type is 8
*Aug  7 02:33:00.087: MCAST-HA Searching for sync request corresponding to the successfully
 received message.
*Aug  7 02:33:00.087: MCAST-HA Transmission from primary and reception by standby confirmed
 for sync type 8. Cleanup is complete.
*Aug  7 02:33:00.087: MCAST-HA: Received cf status CHKPT_STATUS_SEND_OK
*Aug  7 02:33:00.087: MCAST-HA: Sent message type is 8
*Aug  7 02:33:00.087: MCAST-HA Searching for sync request corresponding to the successfully
 received message.
*Aug  7 02:33:00.087: MCAST-HA Transmission from primary and reception by standby confirmed
 for sync type 8. Cleanup is complete.
*Aug  7 02:33:00.087: MCAST-HA: Received cf status CHKPT_STATUS_SEND_OK
*Aug  7 02:33:00.087: MCAST-HA: Sent message type is 11
*Aug  7 02:33:00.087: MCAST-HA Process: Primary RP received standby ACK for reception of
bulk sync completion message.
*Aug  7 02:33:00.087: MCAST-HA Notifying RF to continue progression.
*Aug  7 02:33:00.087: MCAST-HA: Wakeup received for bulk sync completion.
major = 4, minor = 2.
*Aug  7 02:33:00.091: MCAST-HA Process: Primary RP received bulk sync completion confirmation
 from standby.
*Aug  7 02:33:00.091: MCAST-HA RF notification previously sent.
*Aug  7 02:33:00.455: MCAST-HA-RF: Progression event: RF_Event=RF_PROG_STANDBY_HOT
RFState=ACTIVE
00:12:05: %HA_CONFIG_SYNC-6-BULK_CFGSYNC_SUCCEED: Bulk Sync succeeded
00:12:05: %HA-6-STANDBY_READY: Standby RP in slot 7 is operational in SSO  mode
00:12:05: %RF-5-RF_TERMINAL_STATE: Terminal state reached for (SSO)
```

debug ip multicast rpf tracked

To enable debugging output for IP multicast Return Path Forwarding (RPF) tracked events, use the **debug ip multicast rpf tracked**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip multicast rpf tracked

no debug ip multicast rpf tracked

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
15.0(1)M	This command was introduced.

Usage Guidelines

Use this command when IP multicast RPF appears not to be functioning.

Examples

The following example shows how to enable debugging output for IP multicast RPF tracked events:

```
Router# debug ip multicast rpf tracked
```

Related Commands

Command	Description
show ip multicast rpf tracked	Displays IP multicast RPF tracked information.

debug ip multicast topology

To enable debugging output for IP multicast stream topology creation events, deletion events, and IP multicast stream access control list (ACL) matching events, use the **debug ip multicast topology** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip multicast topology

no debug ip multicast topology

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS XE Release 3.2S	This command was introduced.

Usage Guidelines

Use this command when IP multicast stream topology creation, IP multicast stream topology deletion, or IP multicast stream ACL matching appears not to be functioning.

Examples

The following example shows how to enable debugging output for IP multicast stream topology creation events, IP multicast stream topology deletion events, and IP multicast stream ACL matching events:

```
Router# debug ip multicast topology
```

Related Commands

Command	Description
ip multicast rpf select topology	Associates a multicast topology with a multicast group with a specific mroute entry.
ip multicast topology	Configures topology selection for multicast streams.
show ip multicast topology	Displays IP multicast topology information.

debug ip nat

To display information about IP packets translated by the IP Network Address Translation (NAT) feature, use the **debug ip nat**command in privileged EXEC mode. To disable debugging output, use the **no**form of this command.

debug ip nat [*access-list*| **cce**| **detailed**| **h323**| **error**| **fragment**| **generic**| **ipsec**| **multipart**| **nvi**| **piggy-back**| **port**| **pptp**| **route**| **sbc**| **sip**| **skinny**| **tcp-alg**| **vrf**| **wlan-nat**]

no debug ip nat [*access-list*| **cce**| **detailed**| **h323**| **error**| **fragment**| **generic**| **ipsec**| **multipart**| **nvi**| **piggy-back**| **port**| **pptp**| **route**| **sbc**| **sip**| **skinny**| **tcp-alg**| **vrf**| **wlan-nat**]

Syntax Description

access-list	(Optional) Standard IP access list number. If the datagram is not permitted by the specified access list, the related debugging output is suppressed.
cce	(Optional) Displays debug information for all Common Classification Engine (CCE) events.
detailed	(Optional) Displays debugging information in a detailed format.
h323	(Optional) Displays H.225, H.245, and H.323 protocol information.
error	(Optional) Displays debug information for error conditions in NAT-Application Layer Gateway (ALG) segmentation with Layer 4 forwarding.
fragment	(Optional) Displays fragment events.
generic	(Optional) Displays generic ALG handler events.
ipsec	(Optional) Displays IPsec packet information.
multipart	(Optional) Displays multipart processing information.
nvi	(Optional) Displays NAT Virtual Interface (NVI) events.
piggy-back	(Optional) Displays piggyback support events.
port	(Optional) Displays port information.
pptp	(Optional) Displays Point-to-Point Tunneling Protocol (PPTP) information.
route	(Optional) Displays route information.

sbc	(Optional) Displays NAT Session Initiation Protocol (SIP) Session Border Controller (SBC) events.
sip	(Optional) Displays SIP information.
skinny	(Optional) Displays skinny protocol debug information.
tcp-alg	(Optional) Displays debug information for NAT-ALG segmentation with Layer 4 forwarding.
vrf	(Optional) Displays VPN routing and forwarding (VRF) traffic-related information.
wlan-nat	(Optional) Displays Wireless LAN (WLAN) information.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
11.2	This command was introduced.
12.1(5)T	This command was modified. The **h323** keyword was added.
12.2(8)T	This command was modified. The **sip**keyword was added.
12.2(13)T	This command was modified. The **ipsec** and **vrf** keywords were added.
12.3(2)XE	This command was modified. The **wlan-nat** keyword was added.
12.3(7)T	This command was modified. The **wlan-nat** keyword was implemented in Cisco IOS Release 12.3(7)T.
12.3(11)T	This command was modified. The output of the **h323** keyword was expanded to include H.245 tunneling.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
15.0(1)M	This command was modified. The **multipart** keyword was added.
15.1(3)T	This command was modified. The **cce** keyword was removed and the **tcp-alg** keyword was added.

Usage Guidelines

The NAT feature reduces the need for unique, registered IP addresses. It can also save private network administrators from needing to renumber the hosts and routers that do not conform to global IP addressing.

Use the **debug ip nat**command to verify the operation of the NAT feature by displaying information about each packet that the router translates. The **debug ip nat detailed** command generates a description of each packet considered for translation. This command also displays information about certain errors or exception conditions, such as the failure to allocate a global address. To display messages related to the processing of H.225 signaling and H.245 messages, use the **debug ip nat h323** command. To display messages related to the processing of SIP messages, use the **debug ip nat sip** command. To display messages related to the processing of VRF messages, use the **debug ip nat vrf** command. To display messages related to the processing of SIP multipart messages, use the **debug ip nat sip** command.

Caution Because the **debug ip nat** command generates a substantial amount of output, use it only when traffic on the IP network is low, so that the other activity on the system is not adversely affected.

Examples

The following is sample output from the **debug ip nat**command. In this example, the first two lines show the Domain Name System (DNS) request and reply debugging output. The remaining lines show debugging output from a Telnet connection from a host on the inside of the network to a host on the outside of the network. All Telnet packets, except for the first packet, were translated in the fast path, as indicated by the asterisk (*).

```
Router# debug ip nat
NAT: s=192.0.2.1->203.0.112.1, d=203.0.112.254 [6825]
NAT: s=203.0.112.254, d=203.0.112.1->192.0.2.1 [21852]
NAT: s=192.0.2.1->203.0.112.1, d=203.0.112.200 [6826]
NAT*: s=203.0.112.200, d=203.0.112.1->192.0.2.1 [23311]
NAT*: s=192.0.2.1->203.0.112.1, d=203.0.112.200 [6827]
NAT*: s=192.0.2.1->203.0.112.1, d=203.0.112.200 [6828]
NAT*: s=203.0.112.200, d=203.0.112.1->192.0.2.1 [23313]
NAT*: s=203.0.112.200, d=203.0.112.1->192.0.2.1 [23325]
s
```

The table below describes the significant fields shown in the display.

Table 36: debug ip nat Field Descriptions

Field	Description
NAT	Indicates that the packet is being translated by NAT. An asterisk (*) indicates that the translation is occurring in the fast path. The first packet in a conversation always goes through the slow path (that is, it is process switched). The remaining packets go through the fast path if a cache entry exists.
s=192.0.2.1->203.0.112.1	Source address of the packet and how it is being translated.
d=203.0.112.254	Destination address of the packet.

Field	Description
[6825]	IP identification number of the packet. Might be useful in the debugging process to correlate with other packet traces from protocol analyzers.

The following is sample output from the **debug ip nat detailed** command. In this example, the first two lines show the debugging output produced by a DNS request and reply. The remaining lines show the debugging output from a Telnet connection from a host on the inside of the network to a host on the outside of the network. In this example, the inside host 192.168.1.95 was assigned the global address 172.31.233.193. The output fields are self-explanatory.

```
Router# debug ip nat detailed
NAT: i: udp (192.168.1.95, 1493) -> (172.31.2.132, 53) [22399]
NAT: o: udp (172.31.2.132, 53) -> (172.31.233.193, 1493) [63671]
NAT*: i: tcp (192.168.1.95, 1135) -> (172.31.2.75, 23) [22400]
NAT*: o: tcp (172.31.2.75, 23) -> (172.31.233.193, 1135) [22002]
NAT*: i: tcp (192.168.1.95, 1135) -> (172.31.2.75, 23) [22401]
NAT*: i: tcp (192.168.1.95, 1135) -> (172.31.2.75, 23) [22402]
NAT*: o: tcp (172.31.2.75, 23) -> (172.31.233.193, 1135) [22060]
NAT*: o: tcp (172.31.2.75, 23) -> (172.31.233.193, 1135) [22071]
```

The following is sample output from the **debug ip nat h323** command. In this example, an H.323 call is established between two hosts, one host on the inside and the other host on the outside of the network. The debugging output displays the H.323 message names that NAT recognizes and the embedded IP addresses contained in those messages.

```
Router# debug ip nat h323
NAT:H225:[0] processing a Setup message
NAT:H225:[0] found Setup sourceCallSignalling
NAT:H225:[0] fix transportAddress addr=192.168.122.50 port=11140
NAT:H225:[0] found Setup fastStart
NAT:H225:[0] Setup fastStart PDU length:18
NAT:H245:[0] processing OpenLogicalChannel message, forward channel
number 1
NAT:H245:[0] found OLC forward mediaControlChannel
NAT:H245:[0] fix TransportAddress addr=192.168.122.50 port=16517
NAT:H225:[0] Setup fastStart PDU length:29
NAT:H245:[0] Processing OpenLogicalChannel message, forward channel
number 1
NAT:H245:[0] found OLC reverse mediaChannel
NAT:H245:[0] fix Transportaddress addr=192.168.122.50 port=16516
NAT:H245:[0] found OLC reverse mediaControlChannel
NAT:H245:[0] fix TransportAddress addr=192.168.122.50 port=16517
NAT:H225:[1] processing an Alerting message
NAT:H225:[1] found Alerting fastStart
NAT:H225:[1] Alerting fastStart PDU length:25
NAT:H245:[1] processing OpenLogicalChannel message, forward channel
number 1
NAT:H323:[0] received pak, payload_len=46
NAT:H323:[0] processed up to new_payload_len 4
NAT:H323:[0] expecting data len=42--payload_len left 42
NAT:H323:[0] try to process tpkt with len 42, payload_len left 42
NAT:H225:processing a Facility message
NAT:H225:pdu_len :31 msg_IE:28
NAT:H323:choice-value:9
NAT:H225:[0] found h245Tunneling
NAT:H225:[0] found h245Control
NAT:H225:[0] h245control PDU length:20
NAT:H245:[0] processing OpenLogicalChannel message, forward channel
number 2
NAT:H245:[0] found OLC forward mediaControlChannel
NAT:H245:[0] fix TransportAddress addr=192.168.122.50 port=51001
NAT:H245:[0] TransportAddress addr changed 192.168.122.50->172.31.122.129
```

```
NAT:H245:[0] message changed, encoding back
NAT:H245:exit process tpkt with new_len 20
NAT:H225:message changed, encoding back
NAT:H323:[0] processed up to new_payload_len 46
NAT:H323:[0] new pak payload len is 46
```

The table below describes the significant fields shown in the display.

Table 37: debug ip nat h323 Field Descriptions

Field	Description
NAT	Indicates that the packet is being translated by NAT.
H.225, H.245, and H.323	Protocol of the packet.
[0]	Indicates that the packet is moving from a host outside the network to one host inside the network.
[1]	Indicates that the packet is moving from a host inside the network to one host outside the network.

The following is sample output from the **debug ip nat ipsec** command. The output fields are self-explanatory.

```
Router# debug ip nat ipsec
5d21h:NAT:new IKE going In->Out, source addr 192.168.122.35, destination addr 192.168.22.20,
 initiator cookie
0x9C42065D
5d21h:NAT:IPSec:created In->Out ESP translation IL=192.168.122.35 SPI=0xAAE32A0A,
IG=192.168.22.40, OL=192.168.22.20,
OG=192.168.22.20
5d21h:NAT:IPSec:created Out->In ESP translation OG=192.168.22.20 SPI=0xA64B5BB6,
OL=192.168.22.20, IG=192.168.22.40,
IL=192.168.122.35
5d21h:NAT:new IKE going In->Out, source addr 192.168.122.20, destination addr 192.168.22.20,
 initiator cookie
0xC91738FF
5d21h:NAT:IPSec:created In->Out ESP translation IL=192.168.122.20 SPI=0x3E2E1B92,
IG=192.168.22.40, OL=192.168.22.20,
OG=192.168.22.20
5d21h:NAT:IPSec:Inside host (IL=192.168.122.20) trying to open an ESP connection to Outside
 host (OG=192.168.22.20),
wait for Out->In reply
5d21h:NAT:IPSec:created Out->In ESP translation OG=192.168.22.20 SPI=0x1B201366,
OL=192.168.22.20, IG=192.168.22.40,
IL=192.168.122.20
```

The following is sample output from the **debug ip nat sip**command. In this example, one IP phone registers with a Cisco SIP proxy and then calls another IP phone. The debugging output displays the SIP messages that NAT recognizes and the embedded IP addresses contained in those messages.

```
Router# debug ip nat sip
NAT:SIP:[0] processing REGISTER message
NAT:SIP:[0] translated embedded address
192.168.122.3->10.1.1.1
NAT:SIP:[0] translated embedded address
192.168.122.3->10.1.1.1
NAT:SIP:[0] message body found
NAT:SIP:[0] found address/port in SDP body:192.168.122.20
20332
NAT:SIP:[1] processing SIP/2.0 100 Trying reply message
NAT:SIP:[1] translated embedded address
10.1.1.1->192.168.122.3
```

```
NAT:SIP:[1] processing SIP/2.0 200 OK reply message
NAT:SIP:[1] translated embedded address
10.1.1.1->192.168.122.3
NAT:SIP:[1] translated embedded address
10.1.1.1->192.168.122.3
NAT:SIP:[1] processing INVITE message
NAT:SIP:[1] translated embedded address
10.1.1.1->192.168.122.3
NAT:SIP:[1] message body found
NAT:SIP:[1] found address/port in SDP body:192.168.22.20
```

The table below describes the significant fields shown in the display.

Table 38: debug ip nat sip Field Descriptions

Field	Description
NAT	Indicates that the packet is being translated by NAT.
SIP	Protocol of the packet.
[0]	Indicates that the packet is moving from a host outside the network to one host inside the network.
[1]	Indicates that the packet is moving from a host inside the network to one host outside the network.

The following is sample output from the **debug ip nat tcp-alg** command:

```
Router# debug ip nat tcp-alg
*Oct  6 04:56:13.411:  NAT-L4F:setting ALG_NEEDED flag in subblock
*Oct  6 04:56:13.411:  NAT-L4F : Still in the spoofing mode, tcpflags = 0x4
*Oct  6 04:56:13.411:  NAT-L4F : Close notify from L4F
*Oct  6 04:56:13.427:  NAT-L4F:setting ALG_NEEDED flag in subblock
*Oct  6 04:56:23.807:  NAT-L4F:setting ALG_NEEDED flag in subblock
*Oct  6 04:56:23.807:  NAT-L4F: Policy check successful
*Oct  6 04:56:23.807:  NAT-L4F: received fd1: 1073741825 and
                    tcp flags = 0x2, payload_len = 0
*Oct  6 04:56:23.811:  NAT-L4F:setting ALG_NEEDED flag in subblock
*Oct  6 04:56:23.811:  NAT-L4F: received fd2: 1073741826 and
                    tcp flags = 0x12,payload_len = 0
*Oct  6 04:56:23.811:  NAT-L4F:setting ALG_NEEDED flag in subblock
*Oct  6 04:56:23.811:  NAT-L4F: Received final ACK from fd1 : 1073741825 and
                   tcp flags = 0x10
*Oct  6 04:56:23.811:  NAT-L4F:Transistioning to proxy: rc 0 error 0
*Oct  6 04:56:23.811: NAT-ALG: H.225/H.245 ASN encode/decode library initialized
*Oct  6 04:56:23.811:  NAT-L4F: Successfully proxied this flow
*Oct  6 04:56:23.811:  NAT-L4F:setting ALG_NEEDED flag in subblock
*Oct  6 04:56:23.811: NAT-ALG: lookup=0 l7_bytes_recd=12 appl_type=5
*Oct  6 04:56:23.811: NAT-ALG: Skinny l7_msg_size = 12
*Oct  6 04:56:23.811: NAT-ALG: after state machine:
*Oct  6 04:56:23.811: NAT-ALG: remaining_hdr_sz=0
*Oct  6 04:56:23.811: NAT-ALG: remaining_payl_sz=0
*Oct  6 04:56:23.811: NAT-ALG: tcp_alg_state=0
*Oct  6 04:56:23.811: NAT-ALG: complete_msg_len=12
*Oct  6 04:56:23.811:  l4f_send returns 12 bytes
*Oct  6 04:56:23.811:  Complete buffer written to proxy
*Oct  6 04:56:23.811:  NAT-L4F:NO DATA to read
*Oct  6 04:56:23.815:  NAT-L4F:setting ALG_NEEDED flag in subblock
*Oct  6 04:56:24.027:  NAT-L4F:setting ALG_NEEDED flag in subblock
*Oct  6 04:56:24.027: NAT-ALG: lookup=0 l7_bytes_recd=56 appl_type=5
*Oct  6 04:56:24.027: NAT-ALG: Skinny l7_msg_size = 56
*Oct  6 04:56:24.027: NAT-ALG: after state machine:
*Oct  6 04:56:24.027: NAT-ALG: remaining_hdr_sz=0
```

```
*Oct  6 04:56:24.027: NAT-ALG: remaining_payl_sz=0
*Oct  6 04:56:24.027: NAT-ALG: tcp_alg_state=0
*Oct  6 04:56:24.027: NAT-ALG: complete_msg_len=56
*Oct  6 04:56:24.027:  l4f_send returns 56 bytes
*Oct  6 04:56:24.027:  Complete buffer written to proxy
*Oct  6 04:56:24.027:  NAT-L4F:NO DATA to read
*Oct  6 04:56:24.035:  NAT-L4F:setting ALG_NEEDED flag in subblock
*Oct  6 04:56:24.239:  NAT-L4F:setting ALG_NEEDED flag in subblock
*Oct  6 04:56:24.239: NAT-ALG: lookup=0 l7_bytes_recd=16 appl_type=5
*Oct  6 04:56:24.239: NAT-ALG: Skinny l7_msg_size = 16
*Oct  6 04:56:24.239: NAT-ALG: after state machine:
*Oct  6 04:56:24.239: NAT-ALG: remaining_hdr_sz=0
*Oct  6 04:56:24.239: NAT-ALG: remaining_payl_sz=0
*Oct  6 04:56:24.239: NAT-ALG: tcp_alg_state=0
*Oct  6 04:56:24.239: NAT-ALG: complete_msg_len=16
*Oct  6 04:56:24.239:  l4f_send returns 16 bytes
*Oct  6 04:56:24.239:  Complete buffer written to proxy
*Oct  6 04:56:24.239:  NAT-L4F:NO DATA to read
*Oct  6 04:56:24.239:  NAT-L4F:setting ALG_NEEDED flag in subblock
*Oct  6 04:56:24.239: NAT-ALG: lookup=1 l7_bytes_recd=116 appl_type=5
*Oct  6 04:56:24.239: NAT-ALG: Skinny l7_msg_size = 116
*Oct  6 04:56:24.239: NAT-ALG: after state machine:
*Oct  6 04:56:24.239: NAT-ALG: remaining_hdr_sz=0
*Oct  6 04:56:24.239: NAT-ALG: remaining_payl_sz=0
*Oct  6 04:56:24.239: NAT-ALG: tcp_alg_state=0
*Oct  6 04:56:24.239: NAT-ALG: complete_msg_len=116
*Oct  6 04:56:24.239:  l4f_send returns 116 bytes
*Oct  6 04:56:24.239:  Complete buffer written to proxy
*Oct  6 04:56:24.239:  NAT-L4F:NO DATA to read
*Oct  6 04:56:24.239:  NAT-L4F:setting ALG_NEEDED flag in subblock
*Oct  6 04:56:24.239: NAT-ALG: lookup=0 l7_bytes_recd=32 appl_type=5
*Oct  6 04:56:24.239: NAT-ALG: Skinny l7_msg_size = 32
*Oct  6 04:56:24.239: NAT-ALG: after state machine:
*Oct  6 04:56:24.239: NAT-ALG: remaining_hdr_sz=0
*Oct  6 04:56:24.239: NAT-ALG: remaining_payl_sz=0
*Oct  6 04:56:24.239: NAT-ALG: tcp_alg_state=0
*Oct  6 04:56:24.239: NAT-ALG: complete_msg_len=32
*Oct  6 04:56:24.239:  l4f_send returns 32 bytes
*Oct  6 04:56:24.239:  Complete buffer written to proxy
*Oct  6 04:56:24.239:  NAT-L4F:NO DATA to read
*Oct  6 04:56:24.243:  NAT-L4F:setting ALG_NEEDED flag in subblock
*Oct  6 04:56:24.243:  NAT-L4F:read RST, aborting
*Oct  6 04:56:24.243:  NAT-L4F:Buffer list is empty
*Oct  6 04:56:24.243:  NAT-L4F : Close notify from L4F
```

The table below describes the significant fields shown in the display.

Table 39: debug ip nat tcp-alg Field Descriptions

Field	Description
NAT-L4F	Indicates that the packet is being processed by the NAT-ALG interface with Layer 4 forwarding.
NAT-ALG	Indicates that the packet is being processed by NAT-ALG.

The following is sample output from the **debug ip nat vrf** command:

```
Router# debug ip nat vrf
6d00h:NAT:address not stolen for 192.168.121.113, proto 1 port 7224
6d00h:NAT:creating portlist proto 1 globaladdr 10.1.1.10
6d00h:NAT:Allocated Port for 192.168.121.113 -> 10.1.1.10:wanted 7224 got 7224
6d00h:NAT:i:icmp (192.168.121.113, 7224) -> (172.28.88.2, 7224) [2460]
6d00h:NAT:s=192.168.121.113->10.1.1.10, d=172.28.88.2 [2460] vrf=> shop
6d00h:NAT*:o:icmp (172.28.88.2, 7224) -> (10.1.1.10, 7224) [2460]      vrf=> shop
```

```
6d00h:NAT*:s=172.28.88.2, d=10.1.1.10->192.168.121.113 [2460] vrf=> shop
6d00h:NAT:Allocated Port for 192.168.121.113 -> 10.1.1.10:wanted 7225 got 7225
6d00h:NAT:i:icmp (192.168.121.113, 7225) -> (172.28.88.2, 7225) [2461]
6d00h:NAT:s=192.168.121.113->10.1.1.10, d=172.28.88.2 [2461] vrf=> shop
6d00h:NAT*:o:icmp (172.28.88.2, 7225) -> (10.1.1.10, 7225) [2461]      vrf=> shop
6d00h:NAT*:s=172.28.88.2, d=10.1.1.10->192.168.121.113 [2461] vrf=> shop
6d00h:NAT:Allocated Port for 192.168.121.113 -> 10.1.1.10:wanted 7226 got 7226
6d00h:NAT:i:icmp (192.168.121.113, 7226) -> (172.28.88.2, 7226) [2462]
6d00h:NAT:s=192.168.121.113->10.1.1.10, d=172.28.88.2 [2462] vrf=> shop
```

The table below describes the significant fields shown in the display.

Table 40: debug ip nat vrf Field Descriptions

Field	Description
NAT	Indicates that the packet is being translated by NAT.
s=192.168.121.113->10.1.1.10	Source address of the packet and how it is being translated.
d=172.28.88.2	Destination address of the packet.
[2460]	IP identification number of the packet.
vrf=>	Indicates that NAT is applied to a particular VPN.

The following is sample output from the **debug ip nat wlan-nat** command:

```
Router# debug ip nat wlan-nat
WLAN-NAT: Creating secure ARP entry (10.1.1.1,0010.7bc2.9ff6)
WLAN-NAT: Triggered Acct Start for (209.165.201.1,0010.7bc2.9ff6)
WLAN-NAT: Extracting addr:209.165.201.1,input_idb:Ethernet1/2 from pak
WLAN-NAT: Saving address:209.165.201.1,input_idb:Ethernet1/2 in pak
```

After the WLAN-entry times out, the following debugs will be seen:

```
WLAN-NAT: Removing secure arp entry (10.1.1.1,0010.7bc2.9ff6)
WLAN-NAT: triggered Acct Stop for (209.165.201.1,0010.7bc2.9ff6)
```

The table below describes the significant fields shown in the display.

Table 41: debug ip nat wlan-nat Field Descriptions

Field	Description
WLAN	Indicates that a wireless LAN is being translated.
NAT	Indicates that the packet is being translated using NAT.

Related Commands

Command	Description
clear ip nat translation	Clears dynamic NAT translations from the translation table.

Command	Description
ip nat	Designates that traffic originating from or destined for an interface is subject to NAT.
ip nat inside destination	Enables NAT of the inside destination address.
ip nat inside source	Enables NAT of the inside source address.
ip nat outside source	Enables NAT of the outside source address.
ip nat pool	Defines a pool of IP addresses for NAT.
ip nat service	Enables a port other than the default port.
show ip nat statistics	Displays NAT statistics.
show ip nat translations	Displays active NAT translations.

debug ip nat redundancy

To enable debugging output for the IP Network Address Translation (NAT) redundancy, use the **debug ip nat redundancy** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip nat redundancy {[**rf** | **db**] [**errors**]| **messages**| [**detailed** | **errors**]| **cf** | **packets**}

no debug ip nat redundancy {[**rf** | **db**] [**errors**]| **messages**| [**detailed** | **errors**]| **cf** | **packets**}

Syntax Description

rf	Specifies debugging for Redundancy Framework (RF).
db	Specifies debugging for the database.
errors	Specifies debugging for errors cases.
messages	Specifies debugging for messages.
detailed	Specifies detailed debugging for messages.
cf	Specifies debugging for the checkpointing facility.
packets	Specifies debugging for packet information.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
15.3(2)T	This command was introduced.

Usage Guidelines

Use the **debug ip nat redundancy** command to enable debugging output for NAT redundancy.

Examples

The following example shows how to enable debugging output for CF.

```
Device# debug ip nat redundancy cf

IP NAT HA Checkpointing Facility debugging is on

Device# show debugging

*Nov  6 18:41:42.669: NAT-HA-CF: ipnat_ha_cf_msg_callback cf_hndl=33554611 ent_hndl=0
cf_msg=0xE4007230
*Nov  6 18:41:42.669: NAT-HA-CF: Recevied msg: payload=0xE4007270 len=152
```

Related Commands

Command	Description
show ip nat redundancy	Displays NAT redundancy information.
show ip nat translations redundancy	Displays active NAT translations.

debug ip nbar trace

To enable detailed debugging of packets per flow on a data plane, use the **debug ip nbar trace** command in privileged EXEC mode. To disable debugging, use the **no** form of this command.

debug ip nbar trace{**detail** *acl-name* [*packets*] [*packets-per-flow*]| **summary** [*acl-name*] [*number-of-flows*]}

no debug ip nbar trace

Syntax Description

detail	Enables detailed debugging of packets per flow.
acl-name	Specifies the name of the access control list (ACL) configured on the device.
packets	(Optional) Specifies the total number of packets.
packets-per-flow	(Optional) Specifies the number of packets in a flow.
summary	Captures Network-Based Application Recognition (NBAR) classification summary.
number-of-flows	(Optional) Specifies the number of flows.

Command Default

Debugging is disabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
15.2(4)M	This command was introduced.

Usage Guidelines

An ACL name must be configured and NBAR must be enabled.

Examples

The following is sample output from the **debug ip nbar trace detail** command:

```
Device# debug ip nbar trace detail acl 100 200

Graph Id 1
Classification: 82, flag: 163
Packet No: 1
String: Searching Source V4 WKP
String: Searching Destination V4 WKP
String: Entering loop core from Heuristic Regex
State Node:http-verify-heuristic-entry-point-get
```

```
State Node:http-verify-heuristic-entry-point-get
State Node:HTTP-url-get-check
State Node:HTTP-url-get-check
State Node:HTTP-url-get-check
State Node:HTTP-url-get-check
State Node:youtube-found-url
State Node:http-check-url-fe
State Node:HTTP-request-advance-packet-pointer-to-next-http-header
State Node:HTTP-request-advance-packet-pointer-to-next-http-header
State Node:HTTP-request-advance-packet-pointer-to-next-http-header
State Node:HTTP-request-end-of-request-check
State Node:HTTP-request-check-end-of-packet
State Node:HTTP-request-check-end-of-packet
State Node:HTTP-request-headers-parser
State Node:HTTP-request-headers-parser
```

Related Commands

Command	Description
show ip nbar trace	Displays the path traversed by a packet.

debug ip nbar clients

To enable debugging of application programming interfaces (APIs) pertaining to Network-Based Application Recognition (NBAR) on a control plane, use the **debug ip nbar clients** command in privileged EXEC mode. To disable debugging, use the **no** form of the command.

debug ip nbar clients {**high**| **low**| **medium**}

no debug ip nbar clients

Syntax Description

high	Enables high-level debugging.
low	Enables low-, medium-, and high-level debugging.
medium	Enables medium- and low-level debugging.

Command Default

Debugging is disabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
15.2(4)M	This command was introduced.

Usage Guidelines

NBAR must be enabled for debugging.

Examples

The following is sample output from the **debug ip nbar clients low** command:

```
Device# debug ip nbar clients low
```

```
*May 14 08:33:37.468: STILE:CLIENT:LOW: intf list: Interface not found
*May 14 08:33:37.468: STILE:CLIENT:LOW: intf list: Interface not found
*May 14 08:33:37.468: STILE:CLIENT:LOW: intf list: Interface not found
*May 14 08:33:37.468: STILE:CLIENT:LOW: intf list: Interface not found
*May 14 08:33:37.468: STILE:CLIENT:LOW: intf list: Interface not found
*May 14 08:33:37.468: STILE:CLIENT:LOW: Fast flag: SET FLAG
*May 14 08:33:37.468: STILE:CLIENT:LOW: Fast flag: Client configs Fast Flag result end:1
```

debug ip nbar config

To enable debugging of all commands configured for the activation and deactivation of Network-Based Application Recognition (NBAR) on a control plane, use the **debug ip nbar config** command in privileged EXEC mode. To disable debugging, use the **no** form of this command.

debug ip nbar config {**high**| **low**| **medium**}

no debug ip nbar config

Syntax Description

high	Enables high-level debugging.
low	Enables low-, medium-, and high-level debugging.
medium	Enables medium- and low-level debugging.

Command Default

Debugging is disabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
15.2(4)M	This command was introduced.

Examples

The following is sample output from the **debug ip nbar config** command:

```
Device# debug ip nbar config high

*May 14 08:36:59.059: STILE:CONF:HIGH: Attempt to add branch to node that does not have
branches
*May 14 08:36:59.060: STILE:CONF:HIGH: Attempt to add branch to node that does not have
branches
*May 14 08:37:04.314: STILE:CONF:HIGH: Fast flag request for MQC is 1
*May 14 08:37:04.314: STILE:CONF:HIGH: Update fast flag
*May 14 08:37:04.314: STILE:CONF:HIGH: Fast flag request for MQC is 1
*May 14 08:37:04.314: STILE:CONF:HIGH: MQC or P.D set fast flag
```

debug ip nbar platform

To enable debugging of application programming interfaces (APIs) pertaining to Network-Based Application Recognition (NBAR) on a control plane, use the **debug ip nbar platform** command in privileged EXEC mode. To disable debugging, use the **no** form of this command.

debug ip nbar platform {**high**| **low**| **medium**}

no debug ip nbar platform

Syntax Description

high	Enables high-level debugging.
low	Enables low-, medium-, and high-level debugging.
medium	Enables medium- and low-level debugging.

Command Default

Debugging is disabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS XE 3.7S Release	This command was introduced.

Examples

The following is sample output from the **debug ip nbar platform** command:

```
Device# debug ip nbar platform low

*May 14 02:15:29.214: STILE:PLAT:HIGH: fs range: invalid id
*May 14 02:15:29.214: STILE:PLAT:HIGH: fs range: invalid id
*May 14 02:15:29.214: STILE:PLAT:HIGH: fs range: invalid id
*May 14 02:15:29.214: STILE:PLAT:HIGH: fs range: invalid id
```

debug ip ospf adj

To display information on adjacency events related to Open Shortest Path First (OSPF), such as packets being dropped due to a Time-to-Live (TTL) security check, use the **debug ip ospf adj c**ommand in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip ospf adj

no debug ip ospf adj

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
15.0(1)M	This command was integrated into Cisco IOS Release 15.0(1)M.

Examples

The following is sample output from the **debug ip ospf adj**command:

```
Router# debug ip ospf adj
Jan 31 00:13:05.175: OSPF: Drop packet on Serial2/0 from 10.1.1.1 with TTL: 1
Mar 27 23:15:03.175: OSPF Drop packet on OSPF_VL0 from 10.1.1.100 with TTL: 253
```

Information in the output includes the day and time the packet was dropped, protocol name, interface on which the packet was dropped, neighbor address, and TTL hop count.

Related Commands

Command	Description
debug ip ospf events	Displays information on OSPF-related events, such as adjacencies, flooding information, designated router selection, and SPF calculation.

debug ip ospf database-timer rate-limit

To display when link-state advertisement (LSA) rate-limiting timers will expire, use the **debug ip ospf database-timer rate-limit**command in privileged EXEC mode.

debug ip ospf database-timer rate-limit [*access-list-number*]

Syntax Description

access-list-number	(Optional) Number of the standard or expanded IP access list to apply to the debug output. Standard IP access lists are in the range 1 to 99. Expanded IP access lists are in the range 1300 to 1999.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(25)S	This command was introduced.
12.2(27)SBC	This command was integrated into Cisco IOS Release 12.2(27)SBC.
12.2(18)SXD	This command was integrated into Cisco IOS Release 12.2(18)SXD.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

Use this command if you need to see when the timers will expire per LSA. Use an access list if you want to limit the output.

Examples

The following is sample output from the **debug ip ospf database-timer rate-limit** command for an example configuration that includes the **timers throttle lsa all 100 10000 45000** command. Comments are inserted to explain the preceding output.

```
Router# debug ip ospf database-timer rate-limit
OSPF rate limit timer events debugging is on
*Mar 12 20:18:20.383:OSPF:Starting rate limit timer for 10.10.24.4
10.10.24.4 1 with 100ms delay
```

The interface is shut down, which causes OSPF to generate a new router LSA. The system starts a timer for 100 milliseconds.

```
*Mar 12 20:18:20.495:OSPF:Rate limit timer is expired for 10.10.24.4
10.10.24.4 1
```

The rate limit timer is expired after 100 milliseconds (a small delta is added to the timer).

```
*Mar 12 20:18:20.495:OSPF:For next LSA generation - wait :10000ms next:
```

```
20000ms
*Mar 12 20:18:20.495:OSPF:Build router LSA for area 24, router ID
10.10.24.4, seq 0x80000003
```

The system will generate update a router LSA after the timer expires.

debug ip ospf events

To display information on Open Shortest Path First (OSPF)-related events, such as adjacencies, flooding information, designated router selection, and shortest path first (SPF) calculation, use the **debug ip ospf events** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip ospf events

no debug ip ospf events

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Examples

The following is sample output from the **debug ip ospf events** command:

```
Router# debug ip ospf events
OSPF:hello with invalid timers on interface Ethernet0
hello interval received 10 configured 10
net mask received 255.255.255.0 configured 255.255.255.0
dead interval received 40 configured 30
```

The **debug ip ospf events** output shown might appear if any of the following situations occurs:

- The IP subnet masks for routers on the same network do not match.
- The OSPF hello interval for the router does not match that configured for a neighbor.
- The OSPF dead interval for the router does not match that configured for a neighbor.

If a router configured for OSPF routing is not seeing an OSPF neighbor on an attached network, perform the following tasks:

- Make sure that both routers have been configured with the same IP mask, OSPF hello interval, and OSPF dead interval.
- Make sure that both neighbors are part of the same area type.

In the following example line, the neighbor and this router are not part of a stub area (that is, one is a part of a transit area and the other is a part of a stub area, as explained in RFC 1247):

```
OSPF: hello packet with mismatched E bit
```

Related Commands

Command	Description
debug ip pgm host	Displays information about each OSPF packet received.

debug ip ospf mpls traffic-eng advertisements

To print information about traffic engineering advertisements in Open Shortest Path First (OSPF) link state advertisement (LSA) messages, use the **debug ip ospf mpls traffic-eng advertisements** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip ospf mpls traffic-eng advertisements

no debug ip ospf mpls traffic-eng advertisements

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(5)ST	This command was introduced.
12.1(3)T	This command was integrated into Cisco IOS Release 12.1(3)T.
12.0(22)S	This command was integrated into Cisco IOS Release 12.0(22)S.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Examples

In the following example, information about traffic engineering advertisements is printed in OSPF LSA messages:

```
Router# debug ip ospf mpls traffic-eng advertisements
OSPF:IGP delete router node 10.106.0.6 fragment 0 with 0 links
      TE Router ID 10.106.0.6
OSPF:IGP update router node 10.110.0.10 fragment 0 with 0 links
      TE Router ID 10.110.0.10
OSPF:MPLS announce router node 10.106.0.6 fragment 0 with 1 links
      Link connected to Point-to-Point network
      Link ID :10.110.0.10
      Interface Address :10.1.0.6
      Neighbor Address :10.1.0.10
      Admin Metric :10
      Maximum bandwidth :1250000
      Maximum reservable bandwidth :625000
      Number of Priority :8
      Priority 0 :625000      Priority 1 :625000
      Priority 2 :625000      Priority 3 :625000
      Priority 4 :625000      Priority 5 :625000
```

```
Priority 6 :625000      Priority 7 :625000
Affinity Bit :0x0
```

The table below describes the significant fields shown in the display.

Table 42: debug ip ospf mpls traffic-eng advertisements Field Descriptions

Field	Description
Link ID	Index of the link being described.
Interface Address	Address of the interface.
Neighbor Address	Address of the neighbor.
Admin Metric	Administrative weight associated with this link.
Maximum bandwidth	Bandwidth capacity of the link (kbps).
Maximum reservable bandwidth	Amount of reservable bandwidth on this link.
Number of Priority	Number of priority levels for which bandwidth is advertised.
Priority	Bandwidth available at indicated priority level.
Affinity Bit	Attribute flags of the link that are being flooded.

debug ip ospf nsf

To display debugging messages about Open Shortest Path First (OSPF) during a Cisco nonstop forwarding (NSF) restart, use the **debug ip ospf nsf** command in privileged EXEC mode. To disable the display of debugging output, use the **no** form of this command.

debug ip ospf nsf [detail]

no debug ip ospf nsf [detail]

Syntax Description

detail	(Optional) Displays detailed debug messages.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(22)S	This command was introduced.
12.2(18)S	This command was integrated into Cisco IOS Release 12.2(18)S.
12.2(20)S	Support for the Cisco 7304 router was added.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.

Usage Guidelines

Use the **debug ip ospf nsf** command to diagnose problems with OSPF link-state database (LSDB) resynchronization and NSF operations.

Examples

The following example shows that OSPF NSF events debugging is enabled:

```
Router# debug ip ospf nsf
```

Related Commands

Command	Description
nsf (OSPF)	Configures NSF operations for OSPF.
show ip ospf	Displays general information about OSPF routing processes.

Command	Description
show ip ospf neighbor	Displays OSPF-neighbor information on a per-interface basis.

debug ip ospf packet

To display information about each Open Shortest Path First (OSPF) packet received, use the **debug ip ospf packet** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip ospf packet

no debug ip ospf packet

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Examples

The following is sample output from the **debug ip ospf packet** command:

```
Router# debug ip ospf packet
OSPF: rcv. v:2 t:1 l:48 rid:200.0.0.117
      aid:0.0.0.0 chk:6AB2 aut:0 auk:
```

The **debug ip ospf packet** command produces one set of information for each packet received. The output varies slightly depending on which authentication is used. The following is sample output from the **debug ip ospf packet** command when message digest algorithm 5 (MD5) authentication is used.

```
Router# debug ip ospf packet
OSPF: rcv. v:2 t:1 l:48 rid:200.0.0.116
      aid:0.0.0.0 chk:0 aut:2 keyid:1 seq:0x0
```

The table below describes the significant fields shown in the display.

Table 43: debug ip ospf packet Field Descriptions

Field	Description
v:	OSPF version.
t:	OSPF packet type. Possible packet types follow: • 1--Hello • 2--Data description • 3--Link state request • 4--Link state update • 5--Link state acknowledgment
l:	OSPF packet length in bytes.
rid:	OSPF router ID.
aid:	OSPF area ID.

Field	Description
chk:	OSPF checksum.
aut:	OSPF authentication type. Possible authentication types follow: • 0--No authentication • 1--Simple password • 2--MD5
keyid:	MD5 key ID.
seq:	Sequence number.

Related Commands

Command	Description
debug ip http client	Displays information on OSPF-related events, such as adjacencies, flooding information, designated router selection, and SPF calculation.

debug ip ospf rib

To display debugging information for Open Shortest Path First (OSPF) Version 2 routes in the global or local Routing Information Base (RIB), use the **debug ip ospf rib**command in privileged EXEC mode. To disable the debugging of OSPF Version 2 routes, use the **no** form of this command.

debug ip ospf rib [**local**| [**redistribution**| **global** [*access-list-number*]]] **[detail]**

no debug ip ospf rib [**local**| [**redistribution**| **global** [*access-list-number*]]] **[detail]**

Syntax Description

local	(Optional) Displays debugging information for OSPF Version 2 routes in the local RIB.
redistribution	(Optional) Displays debugging information about redistributed OSPF Version 2 routes.
global	(Optional) Displays debugging information for OSPF Version 2 routes in the global RIB.
access-list-number	(Optional) Number of an access list. This is a decimal number from 1 to 199 or from 1300 to 2699.
detail	(Optional) Displays more detailed debug information.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.4(15)T	This command was introduced.
12.2(33)SRC	This command was integrated into Cisco IOS Release 12.2(33)SRC.
12.2(33)SB	This command was integrated into the Cisco IOS 12.2(33)SB release.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.

Usage Guidelines

You can use the output from the **debug ip ospf rib** command to learn about the function of the local RIB and the interaction between the route redistribution process and the global RIB. For example, you can learn why the routes that OSPF placed in the global RIB are not the same ones that you anticipated.

A Cisco Technical Assistance Center representative may ask you to turn on debugging using the **debug ip ospf rib** command as part of troubleshooting a problem.

To monitor updates from the OSPF database to the OSPF local RIB, use the **local** keyword, and to monitor updates from the OSPF database to the OSPF global RIB, use the **global** keyword.

It is highly recommended that you limit the debugging output to information specific to the IP prefix that is associated with a specific access list by entering the *access-list-number* argument.

Examples

The following is sample output from the **debug ip ospf rib**command with the *access-list-number* argument used in order to limit the debugging output to information specific to the IP prefix that is associated with the specific access list 1:

```
Router# show running-config | include access-list 1
access-list 112 permit 10.1.1.0 0.0.0.255
! access-list 1 is configured
Router# debug ip ospf rib local detail 1
*May 31 21:28:17.331: OSPF-RIB-LOCAL: Delete intra-area connected
route 192.168.130.2/255.255.255.0, area 1, dist 10, for interface
Ethernet0/0.1
*May 31 21:28:17.331: OSPF-RIB-LOCAL: Local RIB process OSPF-1
Router clear
*May 31 21:28:17.331: OSPF-RIB-LOCAL: Add intra-area connected
route 192.168.130.2/255.255.255.0, area 1, dist 10, for interface
Ethernet0/0.1
.
.
.
```

Related Commands

Command	Description
debug ip ospf events	Displays information on OSPF-related events, such as adjacencies, flooding information, designated router selection, and SPF calculation.

debug ip ospf spf statistic

To display statistical information while running the shortest path first (SPF) algorithm, use the **debug ip ospf spf statistic**command in privileged EXEC mode. To disable the debugging output, use the **no** form of this command.

debug ip ospf spf statistic

no debug ip ospf spf statistic

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(12)	This command was introduced.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

The **debug ip ospf spf statistic** command displays the SPF calculation times in milliseconds, the node count, and a time stamp.

Examples

The following is sample output from the **debug ip ospf spf statistic**command:

```
Router# debug ip ospf spf statistic
00:05:59: OSPF: Begin SPF at 359.216ms, process time 60ms
00:05:59: spf_time 00:05:59.216, wait_interval 0s
00:05:59: OSPF: End SPF at 359.216ms, Total elapsed time 0ms
00:05:59: Intra: 0ms, Inter: 0ms, External: 0ms
00:05:59: R: 4, N: 2, Stubs: 1
00:05:59: SN: 1, SA: 0, X5: 1, X7: 0
00:05:59: SPF suspends: 0 intra, 1 total
```

The table below describes the significant fields shown in the display.

Table 44: debug ip ospf spf statistic Field Descriptions

Field	Description
Begin SPF at	Absolute time in milliseconds when SPF is started.
process time	Cumulative time since the process has been created.
spf_time	Last time SPF was run or an event has happened to run SPF.

Field	Description
wait_interval	Time waited to run SPF.
End SPF at	Absolute time in milliseconds when SPF had ended.
Total elapsed time	Total time take to run SPF.
Intra:	Time taken to process intra-area link-state advertisements (LSAs).
Inter:	Time taken to process interarea LSAs.
External:	Time taken to process external LSAs.
R:	Number of router LSAs.
N:	Number of network LSAs.
Stubs:	Number of stub links.
SN:	Number of summary network LSAs.
SA:	Number of summary LSAs describing autonomous system boundary routers (ASBRs).
X5:	Number of external type 5 LSAs.
X7:	Number of external type 7 LSAs.
SPF suspends: intra	Number of times process is suspended during intra-area SPF run.
total	Total number of times process is suspended during SPF run.

debug ip packet

To display general IP debugging information and IP security option (IPSO) security transactions, use the **debug ip packet** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip packet [*access-list-number*] **[detail] [dump]**

no debug ip packet [*access-list-number*]

Syntax Description

access-list-number	(Optional) The IP access list number that you can specify. If the datagram is not permitted by that access list, the related debugging output is suppressed. Standard, extended, and expanded access lists are supported. The range of standard and extended access lists is from 1 to 199. The range of expanded access lists is from 1300 to 2699.
detail	(Optional) Displays detailed IP packet debugging information. This information includes the packet types and codes as well as source and destination port numbers.
dump	(Hidden) Displays IP packet debugging information along with raw packet data in hexadecimal and ASCII forms. This keyword can be enabled with individual access lists and also with the **detail** keyword. **Note** The **dump** keyword is not fully supported and should be used only in collaboration with Cisco Technical Support. See the caution notes below, in the usage guidelines, for more specific information.

Command Modes

Privileged EXEC

Usage Guidelines

If a communication session is closing when it should not be, an end-to-end connection problem can be the cause. The **debug ip packet** command is useful for analyzing the messages traveling between the local and remote hosts. IP packet debugging captures the packets that are process switched including received, generated and forwarded packets. IP packets that are switched in the fast path are not captured.

IPSO security transactions include messages that describe the cause of failure each time a datagram fails a security test in the system. This information is also sent to the sending host when the router configuration allows it.

Caution Because the **debug ip packet** command generates a substantial amount of output and uses a substantial amount of system resources, this command should be used with caution in production networks. It should only be enabled when traffic on the IP network is low, so other activity on the system is not adversely affected. Enabling the **detail** and **dump** keywords use the highest level of system resources of the available configuration options for this command, so a high level of caution should be applied when enabling either of these keywords.

Caution The **dump** keyword is not fully supported and should be used only in collaboration with Cisco Technical Support. Because of the risk of using significant CPU utilization, the dump keyword is hidden from the user and cannot be seen using the "?" prompt. The length of the displayed packet information may exceed the actual packet length and include additional padding bytes that do not belong to the IP packet. Also note that the beginning of a packet may start at different locations in the dump output depending on the specific router, interface type, and packet header processing that may have occurred before the output is displayed.

Examples

The following is sample output from the **debug ip packet** command:

```
Router# debug ip packet
IP packet debugging is on
IP: s=172.69.13.44 (Fddi0), d=10.125.254.1 (Serial2), g=172.69.16.2, forward
IP: s=172.69.1.57 (Ethernet4), d=10.36.125.2 (Serial2), g=172.69.16.2, forward
IP: s=172.69.1.6 (Ethernet4), d=255.255.255.255, rcvd 2
IP: s=172.69.1.55 (Ethernet4), d=172.69.2.42 (Fddi0), g=172.69.13.6, forward
IP: s=172.69.89.33 (Ethernet2), d=10.130.2.156 (Serial2), g=172.69.16.2, forward
IP: s=172.69.1.27 (Ethernet4), d=172.69.43.126 (Fddi1), g=172.69.23.5, forward
IP: s=172.69.1.27 (Ethernet4), d=172.69.43.126 (Fddi0), g=172.69.13.6, forward
IP: s=172.69.20.32 (Ethernet2), d=255.255.255.255, rcvd 2
IP: s=172.69.1.57 (Ethernet4), d=10.36.125.2 (Serial2), g=172.69.16.2, access denied
```

The output shows two types of messages that the **debug ip packet** command can produce; the first line of output describes an IP packet that the router forwards, and the third line of output describes a packet that is destined for the router. In the third line of output, rcvd 2 indicates that the router decided to receive the packet.

The table below describes the significant fields shown in the display.

Table 45: debug ip packet Field Descriptions

Field	Description
IP:	Indicates that this is an IP packet.
s=172.69.13.44 (Fddi0)	Indicates the source address of the packet and the name of the interface that received the packet.
d=10.125.254.1 (Serial2)	Indicates the destination address of the packet and the name of the interface (in this case, S2) through which the packet is being sent out on the network.
g=172.69.16.2	Indicates the address of the next-hop gateway.

Field	Description
forward	Indicates that the router is forwarding the packet. If a filter denies a packet, "access denied" replaces "forward," as shown in the last line of output.

The following is sample output from the **debug ip packet** command enabled with the **detail** keyword:

```
Router# debug ip packet detail

IP packet debugging is on (detailed)
001556: 19:59:30: CEF: Try to CEF switch 10.4.9.151 from FastEthernet0/0
001557: 19:59:30: IP: s=10.4.9.6 (FastEthernet0/0), d=10.4.9.151 (FastEthernet03
001558: 19:59:30:     TCP src=179, dst=11001, seq=3736598846, ack=2885081910, wH
001559: 20:00:09: CEF: Try to CEF switch 10.4.9.151 from FastEthernet0/0
001560: 20:00:09: IP: s=10.4.9.4 (FastEthernet0/0), d=10.4.9.151 (FastEthernet03
001561: 20:00:09:     TCP src=179, dst=11000, seq=163035693, ack=2948141027, wiH
001562: 20:00:14: CEF: Try to CEF switch 10.4.9.151 from FastEthernet0/0
001563: 20:00:14: IP: s=10.4.9.6 (FastEthernet0/0), d=10.4.9.151 (FastEthernet03
001564: 20:00:14:     ICMP type=8, code=0
001565: 20:00:14: IP: s=10.4.9.151 (local), d=10.4.9.6 (FastEthernet0/0), len 1g
001566: 20:00:14:     ICMP type=0, code=0
```

The format of the output with **detail keyword**provides additional information, such as the packet type, code, some field values, and source and destination port numbers.

The table below describes the significant fields shown in the display.

Table 46: debug ip packet detail Field Descriptions

Field	Description
CEF:	Indicates that the IP packet is being processed by CEF.
IP:	Indicates that this is an IP packet.
s=10.4.9.6 (FastEthernet0/0)	Indicates the source address of the packet and the name of the interface that received the packet.
d=10.4.9.151 (FastEthernet03)	Indicates the destination address of the packet and the name of the interface through which the packet is being sent out on the network.
TCP src=	Indicates the source TCP port number.
dst=	Indicates the destination TCP port number.
seq=	Value from the TCP packet sequence number field.
ack=	Value from the TCP packet acknowledgement field.
ICMP type=	Indicates ICMP packet type.
code=	Indicates ICMP return code.

The following is sample output from the **debug ip packet** command enabled with the **dump** keyword:

```
Router# debug ip packet dump
IP packet debugging is on (detailed) (dump)
21:02:42: IP: s=10.4.9.6 (FastEthernet0/0), d=10.4.9.4 (FastEthernet0/0), len 13
07003A00:                        0005 00509C08            ...P..
07003A10: 0007855B 4DC00800 45000064 001E0000  ...[M@..E..d....
07003A20: FE019669 0A040906 0A040904 0800CF7C  ~..i..........O|
07003A30: 0D052678 00000000 0A0B7145 ABCDABCD  ..&x......qE+M+M
07003A40: ABCDABCD ABCDABCD ABCDABCD ABCDABCD  +M+M+M+M+M+M+M+M
07003A50: ABCDABCD ABCDABCD ABCDABCD ABCDABCD  +M+M+M+M+M+M+M+M
07003A60: ABCDABCD ABCDABCD ABCDABCD ABCDABCD  +M+M+M+M+M+M+M+M
07003A70: ABCDABCD ABCDABCD ABCDABCD           +M+M+M+M+M+M
21:02:42: IP: s=10.4.9.4 (local), d=10.4.9.6 (FastEthernet0/0), len 100, sending
07003A00:                        0005 00509C08            ...P..
07003A10: 0007855B 4DC00800 45000064 001E0000  ...[M@..E..d....
07003A20: FF019569 0A040904 0A040906 0000D77C  ...i..........W|
07003A30: 0D052678 00000000 0A0B7145 ABCDABCD  ..&x......qE+M+M
07003A40: ABCDABCD ABCDABCD ABCDABCD ABCDABCD  +M+M+M+M+M+M+M+M
07003A50: ABCDABCD ABCDABCD ABCDABCD ABCDABCD  +M+M+M+M+M+M+M+M
07003A60: ABCDABCD ABCDABCD ABCDABCD ABCDABCD  +M+M+M+M+M+M+M+M
07003A70: ABCDABCD ABCDABCD ABCDABCD           +M+M+M+M+M+M
21:02:42: CEF: Try to CEF switch 10.4.9.4 from FastEthernet0/0
21:02:42: IP: s=10.4.9.6 (FastEthernet0/0), d=10.4.9.4 (FastEthernet0/0), len 13
07003380:                        0005 00509C08            ...P..
07003390: 0007855B 4DC00800 45000064 001F0000  ...[M@..E..d....
070033A0: FE019668 0A040906 0A040904 0800CF77  ~..h..........Ow
070033B0: 0D062678 00000000 0A0B7149 ABCDABCD  ..&x......qI+M+M
070033C0: ABCDABCD ABCDABCD ABCDABCD ABCDABCD  +M+M+M+M+M+M+M+M
070033D0: ABCDABCD ABCDABCD ABCDABCD ABCDABCD  +M+M+M+M+M+M+M+M
070033E0: ABCDABCD ABCDABCD ABCDABCD ABCDABCD  +M+M+M+M+M+M+M+M
070033F0: ABCDABCD ABCDABCD ABCDABCD           +M+M+M+M+M+M
```

Note The **dump** keyword is not fully supported and should be used only in collaboration with Cisco Technical Support. See the caution in the usage guidelines section of this command reference page for more specific information.

The output from the **debug ip packet** command, when the **dump** keyword is enabled, provides raw packet data in hexadecimal and ASCII forms. This addtional output is displayed in addition to the standard output. The **dump** keyword can be used with all of the available configuration options of this command.

The table below describes the significant fields shown in the display.

Table 47: debug ip packet dump Field Descriptions

Field	Description
IP:	Indicates that this is an IP packet.
s=10.4.9.6 (FastEthernet0/0)	Indicates the source address of the packet and the name of the interface that received the packet.
d=10.4.9.4 (FastEthernet0/0) len 13	Indicates destination address and length of the packet and the name of the interface through which the packet is being sent out on the network.
sending	Indicates that the router is sending the packet.

The calculation on whether to send a security error message can be somewhat confusing. It depends upon both the security label in the datagram and the label of the incoming interface. First, the label contained in the datagram is examined for anything obviously wrong. If nothing is wrong, assume the datagram to be correct. If something is wrong, the datagram is treated as *unclassified genser* . Then the label is compared with the interface range, and the appropriate action is taken, as the table below describes.

Table 48: Security Actions

Classification	Authorities	Action Taken
Too low	Too low Good Too high	No Response No Response No Response
In range	Too low Good Too high	No Response Accept Send Error
Too high	Too low In range Too high	No Response Send Error Send Error

The security code can only generate a few types of Internet Control Message Protocol (ICMP) error messages. The only possible error messages and their meanings follow:

- ICMP Parameter problem, code 0--Error at pointer
- ICMP Parameter problem, code 1--Missing option
- ICMP Parameter problem, code 2--See Note that follows
- ICMP Unreachable, code 10--Administratively prohibited

Note The message "ICMP Parameter problem, code 2" identifies a specific error that occurs in the processing of a datagram. This message indicates that the router received a datagram containing a maximum length IP header but no security option. After being processed and routed to another interface, it is discovered that the outgoing interface is marked with "add a security label." Because the IP header is already full, the system cannot add a label and must drop the datagram and return an error message.

When an IP packet is rejected due to an IP security failure, an audit message is sent via Department of Defense Intelligence Information System Network Security for Information Exchange (DNSIX) Network Address Translation (NAT). Also, any **debug ip packet** output is appended to include a description of the reason for rejection. This description can be any of the following:

- No basic

- No basic, no response
- Reserved class
- Reserved class, no response
- Class too low, no response
- Class too high
- Class too high, bad authorities, no response
- Unrecognized class
- Unrecognized class, no response
- Multiple basic
- Multiple basic, no response
- Authority too low, no response
- Authority too high
- Compartment bits not dominated by maximum sensitivity level
- Compartment bits do not dominate minimum sensitivity level
- Security failure: extended security disallowed
- NLESO source appeared twice
- ESO source not found
- Postroute, failed xfc out
- No room to add IPSO

debug ip pgm host

Note Support for the PGM Host feature has been removed. Use of this command is not recommended.

To display debug messages for the Pragmatic General Multicast (PGM) Host feature, use the **debug ip pgm host** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip pgm host [**data**| **nak**| **spm**]

no debug ip pgm host [**data**| **nak**| **spm**]

Syntax Description

data	(Optional) Enables debugging for PGM sent (ODATA) and re-sent (RDATA) data packets.
nak	(Optional) Enables debugging for PGM negative acknowledgment (NAK) data packets, NAK confirmation (NCF) data packets, and Null NAK (NNAK) data packets.
spm	(Optional) Enables debugging for PGM source path messages (SPMs).

Command Default

Debugging for PGM Host is not enabled. If the **debug ip pgm host** command is used with no additional keywords, debugging is enabled for all PGM Host message types.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.1(1)T	This command was introduced.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Examples

The following is sample output from the **debug ip pgm host** command:

```
Router# debug ip pgm host

Host SPM debugging is on
Host NAK/NCF debugging is on
Host ODATA/RDATA debugging is on
```

The following is sample output from the **debug ip pgm host** command when the **data** keyword is used:

```
Router# debug ip pgm host data

02:50:23:PGM Host:Received ODATA from 10.0.30.2 to 224.3.3.3 (74 bytes)
02:50:23:     ODATA TSI 00000A001E02-0401 data-dport BBBB csum 9317 tlen 74
02:50:23:     tsqn         31 dsqn         39
```

The following example shows output of the **debug ip pgm host**command when the **nak** keyword is used. In the following example, the host sends a NAK to the source for a missing packet and the source returns an NCF to the host followed by an RDATA data packet.

```
Router# debug ip pgm host nak

02:50:24:PGM Host:Sending NAK from 10.0.32.2 to 10.0.32.1 (36 bytes)
02:50:24:     NAK TSI 00000A001E02-0401 data-dport BBBB csum 04EC tlen 36
02:50:24:     dsqn         38 data source 10.0.30.2 group 224.3.3.3
02:50:24:PGM Host:Received NCF from 10.0.30.2 to 224.3.3.3 (36 bytes)
02:50:24:     NCF TSI 00000A001E02-0401 data-dport BBBB csum 02EC tlen 36
02:50:24:     dsqn         38 data source 10.0.30.2 group 224.3.3.3
02:50:24:PGM Host:Received RDATA from 10.0.30.2 to 224.3.3.3 (74 bytes)
02:50:24:     RDATA TSI 00000A001E02-0401 data-dport BBBB csum 9218 tlen 74
02:50:24:     tsqn         31 dsqn         38
```

The following is sample output from the **debug ip pgm host** command with the **spm** keyword is used:

```
Router# debug ip pgm host spm
02:49:39:PGM Host:Received SPM from 10.0.30.2 to 224.3.3.3 (36 bytes)
02:49:39:     SPM TSI 00000A001E02-0401 data-dport BBBB csum EA08 tlen 36
02:49:39:     dsqn        980 tsqn         31 lsqn         31   NLA 10.0.32.1
```

Related Commands

Command	Description
clear ip pgm host	Resets PGM Host connections to their default values and clears traffic statistics.
ip pgm host	Enables the PGM Host feature.
show ip pgm host defaults	Displays the default values for PGM Host traffic.
show ip pgm host sessions	Displays open PGM Host traffic sessions.
show ip pgm host traffic	Displays PGM Host traffic statistics.

debug ip pgm router

To display debug messages for Pragmatic General Multicast (PGM), use the **debug ip pgm router**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip pgm router [**spm**| **nak**| **data**]

no debug ip pgm router [**spm**| **nak**| **data**]

Syntax Description

spm	(Optional) Enables debugging for Source Path Messages (SPMs).
nak	(Optional) Enables debugging for negative acknowledgments (NAKs), NAK confirmations (NCFs), and Null NAKs (NNAKs).
data	(Optional) Enables debugging for Retransmissions (RDATA).

Command Default

Debugging for PGM is not enabled. If the **debug ip pgm router**command is used with no additional keywords, debugging is enabled for all PGM message types.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(5)T	This command was introduced.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Examples

The following shows sample output from the **debug ip pgm router**command:

```
Router# debug ip pgm router
SPM debugging is on
NAK/NNAK/NCF debugging is on
RDATA debugging is on
```

The following shows sample output from the **debug ip pgm router**command when the **spm** keyword is used:

```
Router# debug ip pgm router spm
PGM: Received SPM on Ethernet1/0/5 from 10.7.0.200 to 227.7.7.7 (52 bytes)
     SPM TSI 0A0700C85555-1000 data-dport 1001 csum CCCC tlen 52
     dsqn 3758096779 tsqn       1954 isqn       1979 lsqn       1990
     NLA 10.7.0.200
     SPM from source/RPF-neighbour 10.7.0.200 for 10.7.0.200 (SPT)
     Forwarded SPM from 10.7.0.200 to 227.7.7.7
```

The following is a debugging message for a selective SPM:

```
Router# debug ip pgm router spm
PGM: Received SPM on Ethernet1/0/5 from 10.7.0.200 to 234.4.3.2 (52 bytes)
     SPM TSI 0A0700C85555-2000 data-dport 2001 csum CCCC tlen 52    Options  P N O
     dsqn 3758096768 tsqn       1986 isqn       1994 lsqn       2006
     NLA 10.7.0.200
     SPM from source/RPF-neighbour 10.7.0.200 for 10.7.0.200 (SPT)
     Forwarded SPM from 10.7.0.200 to 227.7.7.7
```

The “P N O” flags indicate which options are present in this packet:

- P indicates that this is a parity packet.
- N indicates that options are network significant.
- O indicates that options are present.

The following shows sample output from the **debug ip pgm router**command when the **nak** keyword is used:

```
Router# debug ip pgm router nak
PGM: Received NAK on Ethernet1/0/0 from 10.1.0.4 to 10.1.0.2 (36 bytes)
     NAK TSI 0A0700C85555-1000 data-dport 1001 csum CCCC tlen 36
     dsqn       1990 data source 10.7.0.200 group 227.7.7.7
     NAK unicast routed to RPF neighbour 10.4.0.1
     Forwarding NAK from 10.1.0.4 to 10.4.0.1 for 10.7.0.200
PGM: Received NCF on Ethernet1/0/5 from 10.7.0.200 to 227.7.7.7 (36 bytes)
     NCF TSI 0A0700C85555-1000 data-dport 1001 csum CACC tlen 36
     dsqn       1990 data source 10.7.0.200 group 227.7.7.7
     NAK retx canceled for TSI 0A0700C85555-1000 dsqn       1990
     NAK elimination started for TSI 0A0700C85555-1000 dsqn       1990
PGM: Received NCF on Ethernet1/0/5 from 10.7.0.200 to 227.7.7.7 (36 bytes)
     NCF TSI 0A0700C85555-1000 data-dport 1001 csum CACC tlen 36
     dsqn       1991 data source 10.7.0.200 group 227.7.7.7
     No NAK retx outstanding for TSI 0A0700C85555-1000 dsqn       1991
     NAK anticipated for TSI 0A0700C85555-1000 dsqn       1991
```

The following example shows output of the **debug ip pgm router**command with the **data**keyword. The debugging message is for an RDATA packet for which the router has only anticipated state, sqn 1991. Because it did not actually get a NAK, this RDATA is not forwarded by the PGM router.

```
Router# debug ip pgm router data
PGM: Received RDATA on Ethernet1/0/5 from 10.7.0.200 to 227.7.7.7 (70 bytes)
     RDATA TSI 0A0700C85555-1000 data-dport 1001 csum CCCC tlen 32
     tsqn       1954 dsqn       1990
     Marking Ethernet1/0/0 for forwarding
     Marking Serial5/0 for skipping
     Forwarded RDATA from 10.7.0.200 to 227.7.7.7
Debug message for RDATA packet corresponding to a NAK for sqn
1990. Since the NAK was received on Ethernet1/0/0, RDATA is forwarded
out only that interface and another interface in the multicast olist
Serial5/0 is skipped.
PGM: Received RDATA on Ethernet1/0/5 from 10.7.0.200 to 227.7.7.7 (70 bytes)
     RDATA TSI 0A0700C85555-1000 data-dport 1001 csum CCCC tlen 32
     tsqn       1954 dsqn       1991
     Eliminated RDATA (null oif) from 10.7.0.200 to 227.7.7.7
```

Related Commands

Command	Description
ip pgm router	Enables the PGM Router Assist feature for the interface.
show ip pgm router	Displays PGM traffic statistics and TSI state.

debug ip pim

To display Protocol Independent Multicast (PIM) packets received and sent, and to display PIM-related events, use the **debug ip pim**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip pim [**vrf** *vrf-name*] [*group-address*| **atm**| **auto-rp**| **bsr**| **df** [*rp-address*]| **hello**| **tag**]

no debug ip pim [**vrf** *vrf-name*] [*group-address*| **atm**| **auto-rp**| **bsr**| **df** [*rp-address*]| **hello**| **tag**]

Syntax Description

vrf *vrf-name*	(Optional) Displays PIM-related events associated with the Multicast Virtual Private Network (MVPN) routing and forwarding (MVRF) instance specified for the *vrf-name* argument.
group-address	(Optional) IP address or Domain Name System (DNS) name of a multicast group. Entering a multicast group address restricts the output to display only PIM-related events associated with the multicast group address specified for the optional *group-address* argument.
atm	(Optional) Displays PIM ATM signaling activity.
auto-rp	(Optional) Displays the contents of each PIM packet used in the automatic discovery of group-to-rendezvous point (RP) mapping and the actions taken on the address-to-RP mapping database.
bsr	(Optional) Displays candidate-RPs and Bootstrap Router (BSR) activity.
df	(Optional) When bidirectional PIM is used, displays all designated forwarder (DF) election messages.
rp-address	(Optional) The rendezvous point IP address.
hello	(Optional) Displays events associated with PIM hello messages.
tag	(Optional) Displays tag-switching-related activity.

Command Default

All PIM packets are displayed.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
10.2	This command was introduced.
11.1	The **auto-rp** keyword was added.
11.3	The **atm** and **tag** keywords were added.
12.1(2)T	The **df** keyword was added.
12.1(3)T	The **bsr** keyword was added.
12.0(22)S	The **vrf** keyword, *vrf-name* argument, and **hello** keyword were added.
12.2(13)T	The **vrf** keyword and *vrf-name* argument were added.
12.2(14)S	This command was integrated into Cisco IOS Release 12.2(14)S.
12.2(15)T	The **hello** keyword was added.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.

Usage Guidelines

PIM uses Internet Group Management Protocol (IGMP) packets to communicate with routers and advertise reachability information.

Use this command with the **debug ip igmp** and **debug ip mrouting** commands to display additional multicast routing information.

Examples

The following is sample output from the **debug ip pim** command:

```
Router# debug ip pim 224.2.0.1

PIM: Received Join/Prune on Ethernet1 from 172.16.37.33
PIM: Received Join/Prune on Ethernet1 from 172.16.37.33
PIM: Received Join/Prune on Tunnel0 from 10.3.84.1
PIM: Received Join/Prune on Ethernet1 from 172.16.37.33
PIM: Received Join/Prune on Ethernet1 from 172.16.37.33
PIM: Received RP-Reachable on Ethernet1 from 172.16.20.31
PIM: Update RP expiration timer for 224.2.0.1
PIM: Forward RP-reachability packet for 224.2.0.1 on Tunnel0
PIM: Received Join/Prune on Ethernet1 from 172.16.37.33
PIM: Prune-list (10.221.196.51/32, 224.2.0.1)
PIM: Set join delay timer to 2 seconds for (10.221.0.0/16, 224.2.0.1) on Ethernet1
PIM: Received Join/Prune on Ethernet1 from 172.16.37.6
PIM: Received Join/Prune on Ethernet1 from 172.16.37.33
PIM: Received Join/Prune on Tunnel0 from 10.3.84.1
PIM: Join-list: (*, 224.2.0.1) RP 172.16.20.31
PIM: Add Tunnel0 to (*, 224.2.0.1), Forward state
PIM: Join-list: (10.0.0.0/8, 224.2.0.1)
```

```
PIM: Add Tunnel0 to (10.0.0.0/8, 224.2.0.1), Forward state
PIM: Join-list: (10.4.0.0/16, 224.2.0.1)
PIM: Prune-list (172.16.84.16/28, 224.2.0.1) RP-bit set RP 172.16.84.16
PIM: Send Prune on Ethernet1 to 172.16.37.6 for (172.16.84.16/28, 224.2.0.1), RP
PIM: For RP, Prune-list: 10.9.0.0/16
PIM: For RP, Prune-list: 10.16.0.0/16
PIM: For RP, Prune-list: 10.49.0.0/16
PIM: For RP, Prune-list: 10.84.0.0/16
PIM: For RP, Prune-list: 10.146.0.0/16
PIM: For 10.3.84.1, Join-list: 172.16.84.16/28
PIM: Send periodic Join/Prune to RP via 172.16.37.6 (Ethernet1)
```

The following lines appear periodically when PIM is running in sparse mode and indicate to this router the multicast groups and multicast sources in which other routers are interested:

```
PIM: Received Join/Prune on Ethernet1 from 172.16.37.33
PIM: Received Join/Prune on Ethernet1 from 172.16.37.33
```

The following lines appear when a rendezvous point (RP) message is received and the RP timer is reset. The expiration timer sets a checkpoint to make sure the RP still exists. Otherwise, a new RP must be discovered.

```
PIM: Received RP-Reachable on Ethernet1 from 172.16.20.31
PIM: Update RP expiration timer for 224.2.0.1
PIM: Forward RP-reachability packet for 224.2.0.1 on Tunnel0
```

The prune message in the following line states that this router is not interested in the Source-Active (SA) information. This message tells an upstream router to stop forwarding multicast packets from this source. The address 10.221.196.51/32 indicates a host route with 32 bits of mask.

```
PIM: Prune-list (10.221.196.51/32, 224.2.0.1)
```

In the following line, a second router on the network wants to override the prune message that the upstream router just received. The timer is set at a random value so that if additional routers on the network still want to receive multicast packets for the group, only one will actually send the message. The other routers will receive the join message and then suppress sending their own message.

```
PIM: Set join delay timer to 2 seconds for (10.221.0.0/16, 224.2.0.1) on Ethernet1
```

In the following line, a join message is sent toward the RP for all sources:

```
PIM: Join-list: (*, 224.2.0.1) RP 172.16.20.31
```

In the following lines, the interface is being added to the outgoing interface (OIF) of the (*, G) and (S, G) multicast route (mroute) table entry so that packets from the source will be forwarded out that particular interface:

```
PIM: Add Tunnel0 to (*, 224.2.0.1), Forward state
PIM: Add Tunnel0 to (10.0.0.0/8, 224.2.0.1), Forward state
```

The following line appears in sparse mode only. There are two trees on which data may be received: the RP tree and the source tree. In dense mode there is no RP. After the source and the receiver have discovered one another at the RP, the first-hop router for the receiver will usually join to the source tree rather than the RP tree.

```
PIM: Prune-list (172.16.84.16/28, 224.2.0.1) RP-bit set RP 172.16.84.16
```

The send prune message in the next line shows that a router is sending a message to a second router saying that the first router should no longer receive multicast packets for the (S, G). The RP at the end of the message indicates that the router is pruning the RP tree and is most likely joining the source tree, although the router may not have downstream members for the group or downstream routers with members of the group. The output shows the specific sources from which this router no longer wants to receive multicast messages.

```
PIM: Send Prune on Ethernet1 to 172.16.37.6 for (172.16.84.16/28, 224.2.0.1), RP
```

The following lines indicate that a prune message is sent toward the RP so that the router can join the source tree rather than the RP tree:

```
PIM: For RP, Prune-list: 10.9.0.0/16
PIM: For RP, Prune-list: 10.16.0.0/16
PIM: For RP, Prune-list: 10.49.0.0/16
```

In the following line, a periodic message is sent toward the RP. The default period is once per minute. Prune and join messages are sent toward the RP or source rather than directly to the RP or source. It is the responsibility of the next hop router to take proper action with this message, such as continuing to forward it to the next router in the tree.

```
PIM: Send periodic Join/Prune to RP via 172.16.37.6 (Ethernet1)
```

Related Commands

Command	Description
debug ip dvmrp	Displays information on DVMRP packets received and sent.
debug ip igmp	Displays IGMP packets received and sent, and displays IGMP host-related events.
debug ip igrp transactions	Displays transaction information on IGRP routing transactions.
debug ip mrouting	Displays changes to the IP multicast routing table.
debug ip sd	Displays all SD announcements received.

debug ip pim atm

To log Protocol Independent Multicast (PIM) ATM signalling activity, use the **debug ip pim atm** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip pim atm

no debug ip pim atm

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Examples

The following sample output shows a new group being created and the router toward the rendezvous point (RP) opening a new virtual circuit (VC). Because there are now two groups on this router, there are two VCs open, as reflected by the "current count."

The following is sample output from the **debug ip pim atm**command:

```
Router# debug ip pim atm
Jan 28 19:05:51: PIM-ATM: Max VCs 200, current count 1
Jan 28 19:05:51: PIM-ATM: Send SETUP on ATM2/0 for 239.254.254.253/171.69.214.43
Jan 28 19:05:51: PIM-ATM: Received CONNECT on ATM2/0 for 239.254.254.253, vcd 19
Jan 28 19:06:35: PIM-ATM: Max VCs 200, current count 2
```

The table below describes the significant fields shown in the display.

Table 49: debug ip pim atm Field Descriptions

Field	Description
Jan 28 19:05:51	Current date and time (in hours:minutes:seconds).
PIM-ATM	Indicates what PIM is doing to set up or monitor an ATM connection (vc).
current count	Current number of open virtual circuits.

The resulting **show ip mroute** output follows:

```
Router# show ip mroute 239.254.254.253
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, C - Connected, L - Local, P - Pruned
       R - RP-bit set, F - Register flag, T - SPT-bit set, J - Join SPT
Timers: Uptime/Expires
Interface state: Interface, Next-Hop or VCD, State/Mode
(*, 239.254.254.253), 00:00:04/00:02:53, RP 171.69.214.50, flags: S
  Incoming interface: Ethernet1/1, RPF nbr 171.69.214.50
  Outgoing interface list:
    ATM2/0, VCD 19, Forward/Sparse-Dense, 00:00:04/00:02:52
```

debug ip pim auto-rp

To display the contents of each Protocol Independent Multicast (PIM) packet used in the automatic discovery of group-to-rendezvous point (RP) mapping and the actions taken on the address-to-RP mapping database, use the **debug ip pim auto-rp** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip pim auto-rp [**vrf** *vrf-name*]

no debug ip pim auto-rp [**vrf** *vrf-name*]

Syntax Description

vrf	(Optional) Supports the Multicast Virtual Private Network (VPN) routing and forwarding (VRF) instance.
vrf-name	(Optional) Name assigned to the VRF.

Command Default

No default behavior or values.

Command Modes

Privileged EXEC

Command History

Release	Modification
11.3	This command was introduced.
12.0(23)S	The **vrf** keyword and *vrf-name* argument were added.
12.2(13)T	This command was integrated into Cisco IOS Release 12.2(13)T.
12.2(14)S	This command was integrated into Cisco IOS Release 12.2(14)S.
12.2(27)SBC	This command was integrated into Cisco IOS Release 12.2(27)SBC.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Examples

The following is sample output from the **debug ip pim auto-rp** command:

```
Router# debug ip pim auto-rp
Auto-RP: Received RP-announce, from 172.16.214.66, RP_cnt 1, holdtime 180 secs
Auto-RP:  update (192.168.248.0/24, RP:172.16.214.66)
Auto-RP: Build RP-Discovery packet
Auto-RP:  Build mapping (192.168.248.0/24, RP:172.16.214.66),
Auto-RP:  Build mapping (192.168.250.0/24, RP:172.16.214.26).
Auto-RP:  Build mapping (192.168.254.0/24, RP:172.16.214.2).
```

```
Auto-RP: Send RP-discovery packet (3 RP entries)
Auto-RP: Build RP-Announce packet for 172.16.214.2
Auto-RP:  Build announce entry for (192.168.254.0/24)
Auto-RP: Send RP-Announce packet, IP source 172.16.214.2, ttl 8
```

The first two lines show a packet received from 172.16.214.66 announcing that it is the RP for the groups in 192.168.248.0/24. This announcement contains one RP address and is valid for 180 seconds. The RP-mapping agent then updates its mapping database to include the new information.

```
Auto-RP: Received RP-announce, from 172.16.214.66, RP_cnt 1, holdtime 180 secs
Auto-RP:  update (192.168.248.0/24, RP:172.16.214.66)
```

In the next five lines, the router creates an RP-discovery packet containing three RP mapping entries. The packet is sent to the well-known CISCO-RP-DISCOVERY group address (224.0.1.40).

```
Auto-RP: Build RP-Discovery packet
Auto-RP:  Build mapping (192.168.248.0/24, RP:172.16.214.66),
Auto-RP:  Build mapping (192.168.250.0/24, RP:172.16.214.26).
Auto-RP:  Build mapping (192.168.254.0/24, RP:172.16.214.2).
Auto-RP: Send RP-discovery packet (3 RP entries)
```

The final three lines show the router announcing that it intends to be an RP for the groups in 192.168.254.0/24. Only routers inside the scope “ttl 8” receive the advertisement and use the RP for these groups.

```
Auto-RP: Build RP-Announce packet for 172.16.214.2
Auto-RP:  Build announce entry for (192.168.254.0/24)
Auto-RP: Send RP-Announce packet, IP source 172.16.214.2, ttl 8
```

The following is sample output from the **debug ip pim auto-rp** command when a router receives an update. In this example, the packet contains three group-to-RP mappings, which are valid for 180 seconds. The RP-mapping agent then updates its mapping database to include the new information.

```
Router# debug ip pim auto-rp
Auto-RP: Received RP-discovery, from 172.16.214.17, RP_cnt 3, holdtime 180 secs
Auto-RP:  update (192.168.248.0/24, RP:172.16.214.66)
Auto-RP:  update (192.168.250.0/24, RP:172.16.214.26)
Auto-RP:  update (192.168.254.0/24, RP:172.16.214.2
```

debug ip policy

To display IP policy routing packet activity, use the **debug ip policy**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip policy [*access-list-name*]

no debug ip policy [*access-list-name*]

Syntax Description

access-list-name	(Optional) The name of the access list. Displays packets permitted by the access list that are policy routed in process level, Cisco Express Forwarding (CEF), and distributed CEF (DCEF) with NetFlow enabled or disabled. If no access list is specified, information about all policy-matched and policy-routed packets is displayed.

Command Modes

Privileged EXEC

Command History

Release	Command
12.0(3)T	This command was introduced.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

After you configure IP policy routing with the **ip policy** and **route-map** commands, use the **debug ip policy** command to ensure that the IP policy is configured correctly.

Policy routing looks at various parts of the packet and then routes the packet based on certain user-defined attributes in the packet.

The **debug ip policy** command helps you determine what policy routing is following. It displays information about whether a packet matches the criteria, and if so, the resulting routing information for the packet.

Caution Because the **debug ip policy** command generates a substantial amount of output, use it only when traffic on the IP network is low, so other activity on the system is not adversely affected.

Examples

The following is sample output from the **debug ip policy**command:

```
Router# debug ip policy 3
```

```
IP: s=30.0.0.1 (Ethernet0/0/1), d=40.0.0.7, len 100,FIB flow policy match
IP: s=30.0.0.1 (Ethernet0/0/1), d=40.0.0.7, len 100,FIB PR flow accelerated!
IP: s=30.0.0.1 (Ethernet0/0/1), d=40.0.0.7, g=10.0.0.8, len 100, FIB policy routed
```

The table below describes the significant fields shown in the display.

Table 50: debug ip policy Field Descriptions

Field	Description
IP: s=	IP source address and interface of the packet being routed.
d=	IP destination address of the packet being routed.
len	Length of the packet.
g=	IP gateway address of the packet being routed.

debug ip rbscp

To display general error messages about access list-based Rate-Based Satellite Control Protocol (RBSCP), use the **debug ip rbscp**command in privileged EXEC mode. To disable debug output, use the **no** form of this command.

debug ip rbscp

no debug ip rbscp

Syntax Description

This command has no arguments or keywords.

Command Default

RBSCP debugging is disabled by default.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.4(9)T	This command was introduced.

Usage Guidelines

Caution Using this command will impact the router's forwarding performance.

Examples

The following is sample output from the **debug ip rbscp** command. The hexadecminal number is the sequence number to keep track of the flow.

```
Router# debug ip rbscp
*May 11 02:17:01.407: RBSCP process: 0x662852D0 passed access list
```

Related Commands

Command	Description
debug ip rbscp ack-split	Displays information about TCP ACK splitting done in conjunction with RBSCP.
ip rbscp ack-split	Configures the TCP ACK splitting feature of RBSCP on an outgoing interface for packets that are permitted by a specified access list.

debug ip rbscp ack-split

To display information about TCP ACK splitting done in conjunction with Rate-Based Satellite Control Protocol (RBSCP), use the **debug ip rbscp ack-split**command in privileged EXEC mode. To disable debug output, use the **no** form of this command.

debug ip rbscp ack-split

no debug ip rbscp ack-split

Syntax Description

This command has no arguments or keywords.

Command Default

RBSCP debugging for TCP ACKs is disabled by default.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.4(9)T	This command was introduced.

Usage Guidelines

Caution Using this command will impact the router's forwarding performance.

Examples

The following is sample output from the **debug ip rbscp ack-split**command when the packets match the access list applied to RBSCP. The output includes the source and destination IP addresses and port numbers, the hexadecimal sequence number, and the cumulative ACK that acknowledges bytes up to that number.

```
Router# debug ip rbscp ack-split
*May 11 02:17:01.407: RBSCP ACK split: 0x662852D0, input FastEthernet1/0 -> output
FastEthernet1/1
*May 11 02:17:01.407: RBSCP ACK split: rcvd src 1.1.1.1:38481 -> dst 3.3.3.1:21, cumack
2336109115
*May 11 02:17:01.407: RBSCP ACK split: generated 0x65FC0874 cumack 2336109112
*May 11 02:17:01.407: RBSCP ACK split: generated 0x66762A78 cumack 2336109113
*May 11 02:17:01.407: RBSCP ACK split: generated 0x6676442C cumack 2336109114
*May 11 02:17:01.407: RBSCP ACK split: releasing original ACK 2336109115
*May 11 02:17:01.415: RBSCP process: 0x662852D0 passed access list
*May 11 02:17:01.415: RBSCP ACK split: 0x662852D0, input FastEthernet1/0 -> output
FastEthernet1/1
*May 11 02:17:01.415: RBSCP ACK split: rcvd src 1.1.1.1:36022 -> dst 3.3.3.1:20240, cumack
 4024420742
*May 11 02:17:01.415: RBSCP ACK split: generated 0x65FC1E7C cumack 4024420739
*May 11 02:17:01.415: RBSCP ACK split: generated 0x65FC2980 cumack 4024420740
*May 11 02:17:01.415: RBSCP ACK split: generated 0x65FC3484 cumack 4024420741
*May 11 02:17:01.415: RBSCP ACK split: releasing original ACK 4024420742
*May 11 02:17:01.419: RBSCP process: 0x662852D0 passed access list
```

```
*May 11 02:17:01.419: RBSCP ACK split: 0x662852D0, input FastEthernet1/0 -> output
FastEthernet1/1
```

Related Commands

Command	Description
debug ip rbscp	Displays general error messages about access list-based RBSCP.
ip rbscp ack-split	Configures the TCP ACK splitting feature of RBSCP on an outgoing interface for packets that are permitted by a specified access list.

debug ip rgmp

To log debugging messages sent by a Router-Port Group Management Protocol (RGMP)-enabled router, use the **debug ip rgmp** command in privileged EXEC mode. To disable debugging outut, use the **no** form of this command.

debug ip rgmp [*group-name*| *group-address*]

no debug ip rgmp

Syntax Description

group-name	(Optional) The name of a specific IP multicast group.
group-address	(Optional) The IP address of a specific IP multicast group.

Command Default

Debugging for RGMP is not enabled. If the **debug ip rgmp**command is used without arguments, debugging is enabled for all RGMP message types.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(10)S	This command was introduced.
12.1(1)E	The command was integrated into Cisco IOS Release 12.1(1)E.
12.1(5)T	The command was integrated into Cisco IOS Release 12.1(5)T.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Examples

The following shows sample output from the **debug ip rgmp**command:

```
Router# debug ip rgmp

   RGMP: Sending a Hello packet on Ethernet1/0
   RGMP: Sending a Join packet on Ethernet1/0 for group 224.1.2.3
   RGMP: Sending a Leave packet on Ethernet1/0 for group 224.1.2.3
   RGMP: Sending a Bye packet on Ethernet1/0
```

Related Commands

Command	Description
ip rgmp	Enables the RGMP on IEEE 802.3 Ethernet interfaces.

Command	Description
show ip igmp interface	Displays multicast-related information about an interface.

debug ip rip

To display information on Routing Information Protocol (RIP) routing transactions, use the **debug ip rip** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip rip [**bfd events**]

no debug ip rip [**bfd events**]

Syntax Description

bfd events	(Optional) Displays information on RIP Bidirectional Forwarding Detection (BFD)-related events.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.0(21)M	This command was introduced in a release earlier than Cisco IOS Release 12.0(21)M.
Cisco IOS XE Release 3.3S	This command was modified. The **bfd** keyword was added.
15.1(2)S	This command was integrated into Cisco IOS Release 15.1(2)S.

Examples

In the following example, the router being debugged has received updates from a router at source address 10.89.80.28. In this scenario, information has been sent to about five destinations in the routing table update. Notice that the fourth destination address in the update,172.31.0.0, is inaccessible because it is more than 15 hops away from the router from which the update was sent. The router being debugged also sends updates, in both cases to broadcast address 255.255.255.255 as the destination.

```
Router# debug ip rip
RIP: received update from 10.89.80.28 on Ethernet0
  10.89.95.0 in 1 hops
  10.89.81.0 in 1 hops
  10.89.66.0 in 2 hops
  172.31.0.0 in 16 hops (inaccessible)
  0.0.0.0 in 7 hop
RIP: sending update to 255.255.255.255 via Ethernet0 (10.89.64.31)
  subnet 10.89.94.0, metric 1
  172.31.0.0 in 16 hops (inaccessible)
RIP: sending update to 255.255.255.255 via Serial1 (10.89.94.31)
  subnet 10.89.64.0, metric 1
  subnet 10.89.66.0, metric 3
  172.31.0.0 in 16 hops (inaccessible)
  default 0.0.0.0, metric 8
```

The second line is an example of a routing table update. It shows the number of hops between a given Internet address and the router.

The entries show that the router is sending updates that are similar, except that the number in parentheses is the source address encapsulated into the IP header.

The following are examples for the **debug ip rip** command of entries that appear at startup, during an interface transition event, or when a user manually clears the routing table:

```
RIP: broadcasting general request on Ethernet0
RIP: broadcasting general request on Ethernet1
```

The following entry is most likely caused by a malformed packet from the sender:

```
RIP: bad version 128 from 160.89.80.43
```

Related Commands

Command	Description
show ip rip neighbor	Displays RIP neighbors for which BFD sessions are created.

debug ip routing

To display information on Routing Information Protocol (RIP) routing table updates and route cache updates, use the **debug ip routing** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip routing

no debug ip routing

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(13) T	Support for Interior Gateway Routing Protocol (IGRP) was removed.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Examples

The following is sample output from the **debug ip routing** command:

```
Router# debug ip routing
RT: add 172.25.168.0 255.255.255.0 via 172.24.76.30, igrp metric [100/3020]
RT: metric change to 172.25.168.0 via 172.24.76.30, igrp metric [100/3020]
        new metric [100/2930]
IP: cache invalidation from 0x115248 0x1378A, new version 5736
RT: add 172.26.219.0 255.255.255.0 via 172.24.76.30, igrp metric [100/16200]
RT: metric change to 172.26.219.0 via 172.24.76.30, igrp metric [100/16200]
        new metric [100/10816]
RT: delete route to 172.26.219.0 via 172.24.76.30, igrp metric [100/10816]
RT: no routes to 172.26.219.0, entering holddown
IP: cache invalidation from 0x115248 0x1378A, new version 5737
RT: 172.26.219.0 came out of holddown
RT: garbage collecting entry for 172.26.219.0
IP: cache invalidation from 0x115248 0x1378A, new version 5738
RT: add 172.26.219.0 255.255.255.0 via 172.24.76.30, igrp metric [100/10816]
RT: delete route to 172.26.219.0 via 172.24.76.30, igrp metric [100/10816]
RT: no routes to 172.26.219.0, entering holddown
IP: cache invalidation from 0x115248 0x1378A, new version 5739
RT: 172.26.219.0 came out of holddown
RT: garbage collecting entry for 172.26.219.0
IP: cache invalidation from 0x115248 0x1378A, new version 5740
RT: add 172.26.219.0 255.255.255.0 via 172.24.76.30, igrp metric [100/16200]
RT: metric change to 172.26.219.0 via 172.24.76.30, igrp metric [100/16200]
        new metric [100/10816]
RT: delete route to 172.26.219.0 via 172.24.76.30, igrp metric [100/10816]
RT: no routes to 172.26.219.0, entering holddown
IP: cache invalidation from 0x115248 0x1378A, new version 5741
```

In the following lines, a newly created entry has been added to the IP routing table. The “metric change” indicates that this entry existed previously, but its metric changed and the change was reported by means of IGRP. The metric could also be reported via RIP, OSPF, or another IP routing protocol. The numbers inside the brackets report the administrative distance and the actual metric.

```
RT: add 172.25.168.0 255.255.255.0 via 172.24.76.30, igrp metric [100/3020]
RT: metric change to 172.25.168.0 via 172.24.76.30, igrp metric [100/3020]
        new metric [100/2930]
IP: cache invalidation from 0x115248 0x1378A, new version 5736
```

“Cache invalidation” means that the fast-switching cache was invalidated due to a routing table change. “New version” is the version number of the routing table. When the routing table changes, this number is incriminated. The hexadecimal numbers are internal numbers that vary from version to version and software load to software load.

In the following output, the “holddown” and “cache invalidation” lines are displayed. Most of the distance vector routing protocols use “holddown” to avoid typical problems like counting to infinity and routing loops. If you look at the output of the **show ip protocols** command you will see the timer values for “holddown” and “cache invalidation.” “Cache invalidation” corresponds to “came out of holddown.” “Delete route” is triggered when a better path appears. It removes the old inferior path.

```
RT: delete route to 172.26.219.0 via 172.24.76.30, igrp metric [100/10816]
RT: no routes to 172.26.219.0, entering holddown
IP: cache invalidation from 0x115248 0x1378A, new version 5737
RT: 172.26.219.0 came out of holddown
```

debug ip routing static bfd

To enable debugging output on IP static Bidirectional Forwarding Detection (BFD) neighbor events, use the **debug ip routing static bfd**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip routing static bfd

no debug ip routing static bfd

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(33)SRC	This command was introduced.

Examples

The following is sample output from the **debug ip routing static bfd**command:

```
Router# debug ip routing static bfd
*Dec 18 19:01:48.416: IP-ST-BFD(default): queued Config BFD neighbor command: intf
Ethernet1/1, gw 10.1.1.1 *Dec 18 19:01:48.416: IP-ST: Entering ipstatic_bfd_neighbor_add
Router(config)# ip route 10.2.0.0 255.255.0.0 Ethernet1/1 10.1.1.1
Router(config)# *Dec 18 19:02:06.348: IP-ST: head_gwif: NULL *Dec 18 19:02:06.348: IP-ST:
Inserted to GWIF tree (head): 10.2.0.0/16 Et1/1 10.1.1.1 *Dec 18 19:02:16.852: RT: updating
 static 10.2.0.0/16 (0x0) via 10.1.1.1 Et1/1 *Dec 18 19:02:16.856: RT: add 10.2.0.0/16 via
 10.1.1.1, static metric [1/0] RtrB(config)#end RouterB#
```

debug ip rsvp

Caution Use this command with a small number of tunnels or Resource Reservation Protocol (RSVP) reservations. Too much data can overload the CPU.

To display debug messages for RSVP categories, use the **debug ip rsvp** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip rsvp [**all**| **api**| **authentication**| **cli**| **data-pkts**| **database**| **detail**| **dump-messages**| **errors**| **events**| **fast-reroute**| **filter** [**acl**| **vrf** {*****| **vrf-name** [**acl**]}]| **function**| **handles**| **hello**| **messages**| **msg-mgr**| **path**| **policy**| **proxy**| **rate-limit**| **reliable-msg**| **resv**| **routing**| **sbm**| **signalling**| **snmp**| **summary-refresh**| **svc**| **timeouts**| **timer**| **traffic-control**| **wfq**]

no debug ip rsvp

Syntax Description

all	(Optional) RSVP messages for all categories.
api	(Optional) RSVP application programming interface (API) events.
authentication	(Optional) RSVP authentication.
cli	(Optional) RSVP command-line interface (CLI).
data-pkts	(Optional) RSVP data processing.
database	(Optional) RSVP database debugging.
detail	(Optional) RSVP packet content.
dump-messages	(Optional) Dump RSVP message content.
errors	(Optional) Informational debugging messages and messages about irregular events.
events	(Optional) RSVP process events.
fast-reroute	(Optional) RSVP fast-reroute support for label-switched paths (LSPs).
filter	(Optional) RSVP debug message filter.
acl	(Optional) Number (1 to 199) of the access control list (ACL).

vrf *	(Optional) A virtual routing and forwarding (VFR) instance. * = A wildcard to display all VRFs.
vrf *vrf-name*	(Optional) A VFR instance. *vrf-name* = The name of a VRF.
acl	(Optional) Number (1 to 199) of the ACL for the VRF.
function	(Optional) RSVP function names.
handles	(Optional) RSVP database handles event.
hello	(Optional) RSVP hello events.
messages	(Optional) Brief information about all RSVP messages that are sent and received via IP debugging.
msg-mgr	(Optional) RSVP message-manager events.
path	(Optional) RSVP PATH messages.
policy	(Optional) RSVP policy information.
proxy	(Optional) Proxy API trace.
rate-limit	(Optional) RSVP rate-limiting events.
reliable-msg	(Optional) RSVP reliable messages events.
resv	(Optional) RSVP RESV messages.
routing	(Optional) RSVP routing messages.
sbm	(Optional) RSVP subnet bandwidth manager (SBM) messages.
signalling	(Optional) RSVP signalling (PATH and RESV) messages.
snmp	(Optional) RSVP Simple Network Management Protocol (SNMP) events.
sso	(Optional) RSVP stateful switchover (SSO) events.
summary-refresh	(Optional) RSVP summary refresh and bundle messages events.
svc	(Optional) Switched virtual circuit (SVC) events.

timeouts	(Optional) RSVP refresh timeouts.
timer	(Optional) RSVP timer events.
traffic-control	(Optional) RSVP traffic control events.
wfq	(Optional) RSVP weighted fair queueing (WFQ) events.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.0(5)T	This command was introduced.
12.2(13)T	The **dump-messages**, **msg-mgr**, **proxy**, **rate-limit**, **reliable-msg**, and **summary-refresh** keywords were added.
12.0(23)S	The **timeouts** keyword was added.
12.0(24)S	The **hello** keyword was added.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(31)SB2	This command was integrated into Cisco IOS Release 12.2(31)SB2.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.
12.4(20)T	The command output was modified to display RSVP source address and interface information.
15.0(1)M	This command was modified. The optional **vrf** and *****keywords and *vrf-name*argument were added.
12.2(33)SRE	This command was modified. For point-to-multipoint traffic engineering tunnels, the output displays the destination address of the sub-label switched path (LSP).

Examples

Examples

The following output appears in **source-address**: *source-address* format after you configure a source address and enable the **debug ip rsvp cli** command:

Router# debug ip rsvp cli

```
RSVP cli debugging is on

*Sep 11 06:33:27.203: RSVP: RSVP source-address is enabled on interface Ethernet1/0.
source-address: 10.1.3.13
```

The following output appears in **source-interface::address**: *source-interface::address* format after you configure a source interface address and enable the **debug ip rsvp cli** command:

```
*Sep 11 06:33:27.203: RSVP: RSVP source-interface is enabled on interface Ethernet1/0.
source-interface::address: Loopback0::10.1.1.1
```

The following output appears when you enable the **debug ip rsvp path** command and configure a source address in the HOP object of PATH, PATHTEAR, or PATHERROR messages:

```
*Sep 12 08:56:46.267: RSVP: 10.1.1.1_200->10.4.4.4_100[0.0.0.0]: building hop object with
src addr: 10.2.3.23
```

Examples

The following commands show how to enable debugging for RSVP signaling and messages:

```
Router# debug ip rsvp signalling

RSVP signalling messages (Summary) debugging is on
Router# debug ip rsvp messages

RSVP messages (sent/received via IP) debugging is on
```

The following output displays RSVP signaling-related events that include sending and receiving PATH and RESV messages, admitting new reservations, establishing sessions, sending and receiving acknowledgments (ACKs), and sending and receiving summary refresh messages:

```
01:14:56:RSVP 10.20.1.1_19->10.75.1.1_100[10.20.1.1]:Received Path message from 10.20.1.1
(on sender host)
01:14:56:RSVP:new path message passed parsing, continue...
01:14:56:RSVP 10.20.1.1_19->10.75.1.1_100[10.20.1.1]:Refresh Path psb = 61646BB0 refresh
interval = 0mSec
01:14:56:RSVP 10.20.1.1_19->10.75.1.1_100[10.20.1.1]:Sending Path message to 10.4.4.2
01:14:56:RSVP session 10.75.1.1_100[10.20.1.1]:Path sent by IP to 10.4.4.2 length=216
checksum=B1E4 TOS=0xC0 prerouted=YES
router_alert=YES udp=NO (Ethernet1)
01:14:56:RSVP:Resv received from IP layer  (IP HDR 10.4.4.2->10.4.4.1)
01:14:56:RSVP session 10.75.1.1_100[10.20.1.1]:Received RESV for 10.75.1.1 (Ethernet1) from
 10.4.4.2
01:14:56:RSVP 10.20.1.1_19->10.75.1.1_100[10.20.1.1]:reservation not found--new one
01:14:56:RSVP-RESV:Admitting new reservation:6165D0E4
01:14:56:RSVP 10.20.1.1_19->10.75.1.1_100[10.20.1.1]:RSVP bandwidth is available
01:14:56:RSVP-RESV:reservation was installed:6165D0E4
01:14:57:RSVP:Sending Unknown message to 10.4.4.2
01:14:57:RSVP:Ack sent by IP to 10.4.4.2 length=20 checksum=34A7 TOS=0x00 prerouted=NO
router_alert=NO udp=NO (Ethernet1)
01:14:57:RSVP 10.20.1.1_19->10.75.1.1_100[10.20.1.1]:Refresh Path psb = 61646BB0 refresh
interval = 937mSec
01:14:58:%LINK-3-UPDOWN:Interface Tunnel100, changed state to up
01:14:59:%LINEPROTO-5-UPDOWN:Line protocol on Interface Tunnel100, changed state to up
01:15:26:RSVP 10.20.1.1_19->10.75.1.1_100[10.20.1.1]:Refresh Path psb = 61646BB0 refresh
interval = 30000mSec
01:15:26:RSVP 10.20.1.1_19->10.75.1.1_100[10.20.1.1]:Sending Path message to 10.4.4.2
01:15:26:RSVP session 10.75.1.1_100[10.20.1.1]:Path sent by IP to 10.4.4.2 length=216
checksum=B1E4 TOS=0xC0 prerouted=YES
router_alert=YES udp=NO (Ethernet1)
01:15:26:RSVP:Resv received from IP layer  (IP HDR 10.4.4.2->10.4.4.1)
01:15:26:RSVP session 10.75.1.1_100[10.20.1.1]:Received RESV for 10.75.1.1 (Ethernet1) from
 10.4.4.2
01:15:26:RSVP 10.20.1.1_19->10.75.1.1_100[10.20.1.1]:reservation found--processing possible
 change:6165D0E4
01:15:26:RSVP 10.20.1.1_19->10.75.1.1_100[10.20.1.1]:No change in reservation
01:15:27:RSVP:Sending Ack message to 10.4.4.2
```

```
01:15:27:RSVP:Ack sent by IP to 10.4.4.2 length=20 checksum=34A7 TOS=0x00 prerouted=NO
router_alert=NO udp=NO (Ethernet1)
01:15:56:RSVP:Sending Srefresh message to 10.4.4.2
01:15:56:RSVP:Srefresh sent by IP to 10.4.4.2 length=32 checksum=CA0D TOS=0x00 prerouted=NO
 router_alert=NO udp=NO (Ethernet1)
01:15:56:RSVP:Ack received from IP layer  (IP HDR 10.4.4.2->10.4.4.1)
01:15:56:RSVP:Srefresh received from IP layer  (IP HDR 10.4.4.2->10.4.4.1)
01:15:56:RSVP-RESV:Resv state is being refreshed for 0x91
01:15:56:RSVP:Sending Ack message to 10.4.4.2
01:15:56:RSVP:Ack sent by IP to 10.4.4.2 length=20 checksum=34A5 TOS=0x00 prerouted=NO
router_alert=NO udp=NO (Ethernet1)
01:16:26:RSVP:Sending Srefresh message to 10.4.4.2
01:16:26:RSVP:Srefresh sent by IP to 10.4.4.2 length=32 checksum=CA0C TOS=0x00 prerouted=NO
 router_alert=NO udp=NO (Ethernet1)
01:16:26:RSVP:Ack received from IP layer  (IP HDR 10.4.4.2->10.4.4.1)
01:16:26:RSVP:Srefresh received from IP layer  (IP HDR 10.4.4.2->10.4.4.1)
01:16:26:RSVP-RESV:Resv state is being refreshed for 0x91
01:16:26:RSVP:Sending Ack message to 10.4.4.2
01:16:26:RSVP:Ack sent by IP to 10.4.4.2 length=20 checksum=34A3 TOS=0x00 prerouted=NO
router_alert=NO udp=NO (Ethernet1)
```

Related Commands

Command	Description
show debug	Displays active debug output.

debug ip rsvp aggregation

To display debugging output for Resource Reservation Protocol (RSVP) aggregation sessions, use the **debug ip rsvp aggregation**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip rsvp aggregation

no debug ip rsvp aggregation

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(33)SRC	This command was introduced.
Cisco IOS XE Release 2.6	This command was integrated into Cisco IOS XE Release 2.6.

Usage Guidelines

This command displays information about RSVP aggregation sessions.

RSVP aggregation maintains a Finite State Machine (FSM) for each aggregate session. The RSVP code uses the FSM to maintain aggregate states and transition between the states of an aggregate. For example, after the aggregator sends out the aggregate PATH message, a new state will be entered for the aggregate session (RESV_WAIT) to reflect that an aggregate RESV message is expected. If an aggregate RESV message is received, the session enters the ESTABLISHED state. If an aggregate RESV is not received within a timeout, the aggregate session is cleaned and the process starts again.

Each aggregate reservation can be in one of the following states:

- PATH_WAIT--Valid at the deaggregator only. The aggregate reservation at the deaggregator enters this state after the deaggregator has sent a PATHERROR message requesting a new aggregate needed.
- RESV_WAIT--Valid at the aggregator only. The aggregate reservation at the aggregator enters this state after the aggregator has sent a PATH message for the aggregate reservation.
- RESVCONF_WAIT--Valid at the deaggregator only. The aggregate reservation at the deaggregator enters this state after the deaggregator has sent a RESV message for the aggregate reservation.
- ESTABLISHED--Valid at both the aggregator and the deaggregator. The aggregator enters this state after a RESVCONF message has been sent. The deaggregator enters this state after it receives a RESVCONF message for the aggregate reservation.
- SHUT_DELAY--Valid at both the aggregator and the deaggregator. The aggregator and the deaggregator enter this state after the last end-to-end (E2E) reservation has been removed.

There are timers associated with the PATH_WAIT, RESV_WAIT, RESVCONF_WAIT, and SHUT_DELAY states. For example, if an event that is needed to move the FSM out of the PATH_WAIT, RESV_WAIT, or RESVCONF_WAIT state does not occur, (that is, an aggregate PATH message is not received when in the PATH_WAIT state), the timer expires and the aggregate is cleared.

In the successful scenario, the aggregate stays in the ESTABLISHED state as long as some E2E flows are aggregated. Both the aggregator and the deaggregator stay in the SHUT_DELAY state until the timer expires or an aggregate RESVTEAR or PATHTEAR message is received.

Examples

The following example shows output from the **debug ip rsvp aggregation**command taken at an aggregator:

```
Router# debug ip rsvp aggregation
RSVP aggregation debugging is on
*Jan 25 18:40:03.385: RSVP-AGG-3175: 10.3.3.3->10.4.4.4_46[A][4AB8208]:
event=NEW_AGG_NEEDED, current state=START *Jan 25 18:40:03.385: RSVP-AGG-3175:
10.3.3.3->10.4.4.4_46[A][4AB8208]: triggered Aggregate Path to 10.4.4.4 *Jan 25 18:40:03.385:
 RSVP-AGG-3175: 10.3.3.3->10.4.4.4_46[A][4AB8208]: new state=RESV_WAIT *Jan 25 18:40:03.441:
 RSVP-AGG-3175: 10.3.3.3->10.4.4.4_46[A][4AB8208]:
event=AGG_RESV_STATE_CREATED, current state=RESV_WAIT *Jan 25 18:40:03.441: RSVP-AGG-3175:
 10.3.3.3->10.4.4.4_46[A][4AB8208]: new state=ESTABLISHED *Jan 25 18:40:03.465: RSVP-AGG-3175:
 10.3.3.3->10.4.4.4_46[A][4AB8208]:
event=E2E_RESV_STATE_CREATED, current state=ESTABLISHED *Jan 25 18:40:03.465: RSVP-AGG-3175:
 10.3.3.3->10.4.4.4_46[A][4AB8208]:
event=E2E_RESV_STATE_ADMITTED, current state=ESTABLISHED
```

Related Commands

Command	Description
show debugging	Displays active debug output.

debug ip rsvp authentication

To display debugging output related to Resource Reservation Protocol (RSVP) authentication, use the debug **ip rsvp authentication**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip rsvp authentication

no debug ip rsvp authentication

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(15)T	This command was introduced.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.

Usage Guidelines

After you enable RSVP authentication, RSVP logs system error events whenever an authentication check fails. These events are logged instead of just being displayed when debugging is enabled because they may indicate potential security attacks. The events are generated when:

- RSVP receives a message that does not contain the correct cryptographic signature. This could be due to misconfiguration of the authentication key or algorithm on one or more RSVP neighbors, but it may also indicate an (unsuccessful) attack.
- RSVP receives a message with the correct cryptographic signature, but with a duplicate authentication sequence number. This may indicate an (unsuccessful) message replay attack.
- RSVP receives a message with the correct cryptographic signature, but with an authentication sequence number that is outside the receive window. This could be due to a reordered burst of valid RSVP messages, but it may also indicate an (unsuccessful) message replay attack.
- Failed challenges result from timeouts or bad challenge responses.

Examples

The following example shows output from the **debug ip rsvp authentication**command in which the authentication type (digest) and the sequence number have been validated:

```
Router# debug ip rsvp authentication
RSVP authentication debugging is on
Router# show debugging
*Jan 30 08:10:46.335:RSVP_AUTH:Resv integrity digest from 192.168.101.2 valid
*Jan 30 08:10:46.335:RSVP_AUTH:Resv integrity sequence number 13971113505298841601 from
```

```
192.168.101.2 valid
*Jan 30 08:10:46.335:RSVP_AUTH:Resv from 192.168.101.2 passed all authentication checks
```

Note Cisco routers using RSVP authentication on Cisco IOS software ideally should have clocks that can be accurately restored to the correct time when the routers boot. This capability is available on certain Cisco routers that have clocks with battery backup. For those platforms that do not have battery backup, consider configuring the router to keep its clock synchronized with a Network Time Protocol (NTP) time server. Otherwise, if two adjacent routers have been operating with RSVP authentication enabled and one of them reboots such that its clock goes backward in time, it is possible (but unlikely) the router that did not reboot will log RSVP authentication sequence number errors.

Related Commands

Command	Description
ip rsvp authentication	Activates RSVP cryptographic authentication.
show debugging	Displays active debug output.

debug ip rsvp detail

To display detailed information about Resource Reservation Protocol (RSVP)-enabled and Subnetwork Bandwidth Manager (SBM) message processing, use the **debug ip rsvp detail**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip rsvp detail

no debug ip rsvp detail

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(5)T	This command was introduced.
12.0(23)S	This command was integrated into Cisco IOS Release 12.0(23)S.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Examples

The following example shows the detailed debug information about RSVP and SBM that is available when you enable debug mode through the **debug ip rsvp detail** command:

```
Router# debug ip rsvp detail
RSVP debugging is on
router2#u
*Dec 31 16:44:29.651: RSVP: send I_AM_DSBM message from 145.2.2.150
*Dec 31 16:44:29.651: RSVP: IP to 224.0.0.17 length=88 checksum=43AF
(Ethernet2)
*Dec 31 16:44:29.651: RSVP: version:1 flags:0000 type:I_AM_DSBM cksum:43AF
                        ttl:254 reserved:0 length:88
*Dec 31 16:44:29.651:  DSBM_IP_ADDR       type 1 length 8 : 91020296
*Dec 31 16:44:29.651:  HOP_L2             type 1 length 12: 00E01ECE
*Dec 31 16:44:29.651:                                      : 0F760000
*Dec 31 16:44:29.651:  SBM_PRIORITY       type 1 length 8 : 00000064
*Dec 31 16:44:29.651:  DSBM_TIMERS        type 1 length 8 : 00000F05
*Dec 31 16:44:29.651:  SBM_INFO           type 1 length 44: 00000000
*Dec 31 16:44:29.651:                                      : 00240C02 00000007
*Dec 31 16:44:29.651:                                      : 01000006 7F000005
*Dec 31 16:44:29.651:                                      : 00000000 00000000
*Dec 31 16:44:29.655:                                      : 00000000 00000000
*Dec 31 16:44:29.655:                                      : 00000000
```

Related Commands

Command	Description
debug ip rsvp	Displays information about SBM message processing, the DSBM election process, and RSVP message processing.
debug ip rsvp detail sbm	Displays detailed information about the contents of SMB messages only, and SBM and DSBM state transitions.
ip rsvp dsbm-candidate	Configures an interface as a DSBM candidate.
show ip rsvp sbm	Displays information about SBM configured for a specific RSVP-enabled interface or all RSVP-enabled interfaces on the router.

debug ip rsvp dump-messages

Caution Use this command with a small number of tunnels or Resource Reservation Protocol (RSVP) reservations. Too much data can overload the console.

To display debugging messages for all RSVP events, use the **debug ip rsvp dump-messages** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip rsvp dump-messages [**hex**| **path**| **resv**| **sbm**| **signalling**]

no debug ip rsvp dump-messages

Syntax Description

hex	(Optional) Hex dump of packet contents.
path	(Optional) Contents of Path messages.
resv	(Optional) Contents of Resv messages.
sbm	(Optional) Contents of SBM messages.
signalling	(Optional) Contents of all signaling (Path and Resv) messages.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(13)T	This command was introduced.
12.0(24)S	This command was integrated into Cisco IOS Release 12.0(24)S.

Examples

The following command shows how to enable debugging for RSVP events:

```
Router# debug ip rsvp dump-messages
RSVP message dump debugging is on
```

In the following display, notice that a Path message is transmitted and an ACK_DESIRED flag is set for ID: 0x26 Epoch: 0x76798A. In response, a Resv message is sent and an acknowledgment (ACK) is issued for ID: 0x26 Epoch: 0x76798A indicating the RSVP state is established on the neighboring router:

```
00:37:15:RSVP:version:1 flags:0000 type:PROXY_PATH cksum:0000 ttl:255 reserved:0 length:212
00:37:15: SESSION              type 7 length 16:
00:37:15:      Destination 140.75.1.1, TunnelId 100, Source 140.20.1.1, Protocol 0, Flags
0000
00:37:15: HOP                  type 1 length 12:
00:37:15:      Neighbor 140.20.1.1, LIH 0x00000000
00:37:15: TIME_VALUES          type 1 length 8 :
00:37:15:      Refresh period is 30000 msecs
00:37:15: SENDER_TEMPLATE      type 7 length 12:
00:37:15:      Source 140.20.1.1, tunnel_id 9
00:37:15: SENDER_TSPEC         type 2 length 36:
00:37:15:      version=0, length in words=7
00:37:15:      Token bucket fragment (service_id=1, length=6 words
00:37:15:          parameter id=127, flags=0, parameter length=5
00:37:15:          average rate=1250 bytes/sec, burst depth=1000 bytes
00:37:15:          peak rate   =1250 bytes/sec
00:37:15:          min unit=0 bytes, max pkt size=4294967295 bytes
00:37:15: ADSPEC               type 2 length 48:
00:37:15: version=0  length in words=10
00:37:15: General Parameters  break bit=0  service length=8
00:37:15:                                            IS Hops:0
00:37:15:                 Minimum Path Bandwidth (bytes/sec):2147483647
00:37:15:                        Path Latency (microseconds):0
00:37:15:                                           Path MTU:-1
00:37:15: Controlled Load Service  break bit=0  service length=0
00:37:15: LABEL_REQUEST        type 1 length 8 :
00:37:15:      Layer 3 protocol ID:2048
00:37:15: EXPLICIT_ROUTE       type 1 length 36:
00:37:15:      (#1) Strict IPv4 Prefix, 8 bytes, 140.20.1.1/32
00:37:15:      (#2) Strict IPv4 Prefix, 8 bytes, 140.4.4.2/32
00:37:15:      (#3) Strict IPv4 Prefix, 8 bytes, 140.70.1.1/32
00:37:15:      (#4) Strict IPv4 Prefix, 8 bytes, 140.70.1.2/32
00:37:15: SESSION_ATTRIBUTE    type 7 length 28:
00:37:15:        Session name:tagsw4500-21_t100
00:37:15:        Setup priority:7, reservation priority:7
00:37:15:        Status:May-Reroute
00:37:15:
00:37:15:RSVP:version:1 flags:0001 type:Path cksum:D61E ttl:255 reserved:0 length:216
00:37:15: MESSAGE_ID           type 1 length 12:
00:37:15:      ID:0x26 Epoch:0x76798A
00:37:15:      Flags:ACK_DESIRED
00:37:15: SESSION              type 7 length 16:
00:37:15:      Destination 140.75.1.1, TunnelId 100, Source 140.20.1.1, Protocol 0, Flags
0000
00:37:15: HOP                  type 1 length 12:
00:37:15:      Neighbor 140.4.4.1, LIH 0x10000401
00:37:15: TIME_VALUES          type 1 length 8 :
00:37:15:      Refresh period is 30000 msecs
00:37:15: EXPLICIT_ROUTE       type 1 length 28:
00:37:15:      (#1) Strict IPv4 Prefix, 8 bytes, 140.4.4.2/32
00:37:15:      (#2) Strict IPv4 Prefix, 8 bytes, 140.70.1.1/32
00:37:15:      (#3) Strict IPv4 Prefix, 8 bytes, 140.70.1.2/32
00:37:15: LABEL_REQUEST        type 1 length 8 :
00:37:15:      Layer 3 protocol ID:2048
00:37:15: SESSION_ATTRIBUTE    type 7 length 28:
00:37:15:        Session name:tagsw4500-21_t100
00:37:15:        Setup priority:7, reservation priority:7
00:37:15:        Status:May-Reroute
00:37:15: SENDER_TEMPLATE      type 7 length 12:
00:37:15:      Source 140.20.1.1, tunnel_id 9
00:37:15: SENDER_TSPEC         type 2 length 36:
00:37:15:      version=0, length in words=7
00:37:15:      Token bucket fragment (service_id=1, length=6 words
00:37:15:          parameter id=127, flags=0, parameter length=5
00:37:15:          average rate=1250 bytes/sec, burst depth=1000 bytes
00:37:15:          peak rate   =1250 bytes/sec
00:37:15:          min unit=0 bytes, max pkt size=4294967295 bytes
```

```
00:37:15: ADSPEC               type 2 length 48:
00:37:15: version=0  length in words=10
00:37:15: General Parameters  break bit=0  service length=8
00:37:15:                                            IS Hops:1
00:37:15:                 Minimum Path Bandwidth (bytes/sec):1250000
00:37:15:                        Path Latency (microseconds):0
00:37:15:                                           Path MTU:1500
00:37:15: Controlled Load Service  break bit=0  service length=0
00:37:15:
00:37:15:RSVP:version:1 flags:0001 type:Resv cksum:DADF ttl:255 reserved:0 length:132
00:37:15: MESSAGE_ID_ACK       type 1 length 12:
00:37:15:      Type:ACK
00:37:15:      ID:0x26 Epoch:0x76798A
00:37:15:      Flags:None
00:37:15: MESSAGE_ID           type 1 length 12:
00:37:15:      ID:0x43 Epoch:0xE1A1B7
00:37:15:      Flags:ACK_DESIRED
00:37:15: SESSION              type 7 length 16:
00:37:15:      Destination 140.75.1.1, TunnelId 100, Source 140.20.1.1, Protocol 0, Flags
0000
00:37:15: HOP                  type 1 length 12:
00:37:15:      Neighbor 140.4.4.2, LIH 0x10000401
00:37:15: TIME_VALUES          type 1 length 8 :
00:37:15:      Refresh period is 30000 msecs
00:37:15: STYLE                type 1 length 8 :
00:37:15:      Shared-Explicit (SE)
00:37:15: FLOWSPEC             type 2 length 36:
00:37:15:      version = 0 length in words = 7
00:37:15:      service id = 5, service length = 6
00:37:15:      tspec parameter id = 127, flags = 0, length = 5
00:37:15:      average rate = 1250 bytes/sec, burst depth = 1000 bytes
00:37:15:      peak rate    = 1250 bytes/sec
00:37:15:      min unit = 0 bytes, max pkt size = 0 bytes
00:37:15: FILTER_SPEC          type 7 length 12:
00:37:15:      Source 140.20.1.1, tunnel_id 9
00:37:15: LABEL                type 1 length 8 :
00:37:15:      Labels:16
00:37:15:
00:37:15:RSVP:version:1 flags:0001 type:Ack cksum:34F5 ttl:255 reserved:0 length:20
00:37:15: MESSAGE_ID_ACK       type 1 length 12:
00:37:15:      Type:ACK
00:37:15:      ID:0x43 Epoch:0xE1A1B7
00:37:15:      Flags:None
00:37:15:
00:37:17:%LINK-3-UPDOWN:Interface Tunnel100, changed state to up
00:37:18:%LINEPROTO-5-UPDOWN:Line protocol on Interface Tunnel100, changed state to up
```

Related Commands

Command	Description
ip rsvp signalling refresh reduction	Enables refresh reduction.
show debug	Displays active debug output.

debug ip rsvp errors

To display informational debugging messages and messages about irregular events, use the **debug ip rsvp errors**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip rsvp errors

no debug ip rsvp errors

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(13)T	This command was introduced.
12.0(29)S	This command was integrated into Cisco IOS Release 12.0(29)S.

Usage Guidelines

Use the **debug ip rsvp errors** command to display informational messages and messages about irregular events such as an incomplete setup or breakdown of an RSVP session. Informational messages do not necessarily indicate problems. It is useful to use this command if something has gone wrong, but you do not know what.

If you enter a different debug command, such as **debug ip rsvp signalling**, all the signalling errors and the normal signalling events are displayed. You do not have to also enter the **debug ip rsvp errors** command.

If there are many active RSVP sessions, enter the following configuration command to activate ACL filtering so that you will view only relevant debugging messages.

```
Router(config)# access-list
 number
 permit
 udp
 src_ip
 src_port
 dst_ip
 dst_port
```

Where

- *number* --Access list number, from 100 to 199
- *src_ip* --The tunnel headend
- *src_port* --The link-state packet (LSP) ID
- *dst_ip* --The tunnel tailend

- *dst_port* --The tunnel ID, where the tunnel ID is the tunnel interface number

Then enter the following command to turn on ACL filtering:

```
Router# debug ip rsvp filter
```

In the following example, debugging is allowed only when the session is initiated from 192.168.1.4 toward 192.168.1.8, for Tunnel8 on the headend.

Note This ACL will capture both PATH and RESV messages for the session from 192.168.1.4 to 192.168.1.8, but not any tunnels originating from 1.8 going to 1.4. You can also specify the LSP ID, but that is less useful because it changes all the time, and the combination of the head, tail, and tunnel ID is generally enough to limit the output to what you want.

```
Router(config)# access-list 101 permit udp host 192.168.1.4 host 192.168.1.8 eq 8
Router# debug ip rsvp filter
```

Examples

The following is sample output from the **debug ip rsvp errors**command:

```
Router# debug ip rsvp errors
*May 21 08:54:31.918: RSVP: 5.1.1.1_68->7.1.1.1_3[5.1.1.1]: Problem parsing PATH message:
MISFORMATTED object (13) C-Type (2)
```

debug ip rsvp hello

To verify that a Hello instance has been created,that a Hello instance has been deleted, or that communication with a neighbor has been lost, use the **debug ip rsvp hello** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip rsvp hello [client] [detail] [messages] [stats]

no debug ip rsvp hello [client] [detail] [messages] [stats]

Syntax Description

client	(Optional) Indicates whether clients are enabled or disabled.
detail	(Optional) Indicates whether detailed output is enabled or disabled.
messages	(Optional) Indicates whether messages are enabled or disabled.
stats	(Optional) Indicates whether statistics are enabled or disabled.

Command Default

Debugging activity for the Hello instance or communication with a neighbor does not occur.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(22)S	This command was introduced.
12.2(18)SXD1	This command was integrated into Cisco IOS Release 12.2(18)SXD1.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.4(20)T	This command was integrated into Cisco IOS Release 12.4(20)T.

Usage Guidelines

When you enter the **debug ip rsvp hello** command, Resource Reservation Protocol (RSVP) signaling messages are shown, but RSVP hello messages are excluded because of the large number of hello messages that are sent.

Examples

Following is sample output from the **debug ip rsvp hello** command. The first portion of the output is for serial interface 2/0 when Hello is created.

```
Router# debug ip rsvp hello
00:22:03: RSVP-HELLO: rsvp_hello_inst_init: Initializing ACTIVE hello inst 10.0.0.2->10.0.0.3

00:22:03: RSVP-HELLO: rsvp_hello_create_instance_from_psb: Next hop Se2/0 is adjacent
00:22:03: RSVP-HELLO: rsvp_hello_create_instance_from_psb: Create hello instance for
10.0.0.2->10.0.0.3 on Se2/0 (psb=61BC5F60)
00:22:03: RSVP-HELLO: rsvp_hello_find_instance: psb_cnt=2 for hello inst 10.0.0.2->10.0.0.3

00:22:03: RSVP-HELLO: rsvp_hello_incoming_message: Neighbor 10.0.0.3 state changed to UP
00:22:05: %LINK-3-UPDOWN: Interface Tunnel1, changed state to up
00:22:06: %LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel1, changed state to up
Router(config-if)#
Router(config-if)# shut

Router(config-if)#
```

The following output shows that Hello has been deleted:

```
00:25:19: RSVP-HELLO: rsvp_hello_path_delete: psb for hello inst 10.0.0.2->10.0.0.3 exited
 READY state (psb_cnt=1)
00:25:19: RSVP-HELLO: rsvp_hello_path_delete: psb for hello inst 10.0.0.2->10.0.0.3 exited
 READY state (psb_cnt=0)
00:25:19: RSVP-HELLO: rsvp_hello_path_delete: Last psb deleted, hello inst for
10.0.0.2->10.0.0.3 ACTIVE->PASSIVE
00:25:19: RSVP-HELLO: rsvp_hello_path_delete: psb for hello inst 10.0.0.2->10.0.0.3 exited
 READY state (psb_cnt=0)
00:25:19: RSVP-HELLO: rsvp_hello_path_delete: Last psb deleted, hello inst for
10.0.0.2->10.0.0.3 ACTIVE->PASSIVE
00:25:21: %LINK-5-CHANGED: Interface Tunnel1, changed state to administratively down
00:25:22: %LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel1,
changed state to down
00:05:51: RSVP-HELLO: Communication lost with 10.0.0.2
00:05:51: RSVP-HELLO: rsvp_hello_communication_lost: Neighbor 10.0.0.2 was reset (src_inst)
```

Following is sample output from the **debug ip rsvp hello stats**command:

```
Router(config)# debug ip rsvp hello stats
Router#
00:32:28: RSVP-HELLO: rsvp_hello_stats_init: Hello stats is being configured
```

Related Commands

Command	Description
ip rsvp signalling hello (configuration)	Enables Hello globally on the router.
ip rsvp signalling hello dscp	Sets the DSCP value that is in the IP header of the Hello message sent out from an interface.
ip rsvp signalling hello (interface)	Enables Hello on an interface where you need Fast Reroute protection.
ip rsvp signalling hello refresh interval	Configures the Hello request interval.
ip rsvp signalling hello refresh misses	Specifies how many Hello acknowledgments a node can miss in a row before the node considers that communication with its neighbor is down.

Command	Description
ip rsvp signalling hello statistics	Enables Hello statistics on the router.

debug ip rsvp high-availability

To display debugging output for Resource Reservation Protocol traffic engineering (RSVP-TE) high availability (HA) activities that improve the accessibility of network resources, use the **debug ip rsvp high-availability**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip rsvp high-availability {**all**| **database**| **errors**| **events**| **fsm**| **issu**| **messages**}

no debug ip rsvp high-availability {**all**| **database**| **errors**| **events**| **fsm**| **issu**| **messages**}

Syntax Description

all	Displays debugging output for all RSVP-TE HA categories except for the dumping of messages.
database	Displays information about read and write operations to and from the checkpointed database during the RSVP-TE HA activities.
errors	Displays errors encountered by RSVP-TE during HA activities.
events	Displays significant RSVP-TE stateful switchover (SSO) events during RSVP-TE HA activities, such as: • RSVP-TE process events • RSVP-TE Route Processor (RP) state (active, standby, and recovery) changes • Recovery period beginning and end • Redundant Facility (RF) events handled by RSVP-TE
fsm	Displays significant events for the RSVP-TE checkpointed database finite state machine (fsm) during the RSVP-TE HA activities.
issu	Displays information about RSVP-TE In-Service Software Upgrade (ISSU) activity.
messages	Displays information about Checkpointing Facility (CF) messages sent by RSVP-TE between the active RP and the standby RP.

Command Default

Debugging is not enabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(33)SRA	This command was introduced.
12.2(33)SRB	Support for ISSU was added.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.

Usage Guidelines

This command displays information about RSVP-TE activities, before and after SSO, that improve the availability of network resources and services.

Examples

The following example is sample output from the **debug ip rsvp high-availability all** command, which turns on debugging for IP RSVP-TE HA events, messages, database, errors, fsm, and ISSU:

```
Router# debug ip rsvp high-availability all
RSVP HA all debugging is on
Router# show debug        <---- This command displays the debugging that is enabled.
IP RSVP HA debugging is on for:
   events
   messages
   database
   errors
   fsm
   issu
```

This sample debugging output is displayed as an SSO recovery begins on the standby router in the process of the standby router becoming active.

Note The prefix in the debug output is composed of label switched path (LSP) 5-tuples in the following format: 10.0.0.3_61->10.0.0.9_10[10.0.0.3]. The 10.0.0.3 represents the source address, the 61 represents the LSP ID, the 10.0.0.9 represents the tunnel destination (tunnel tail), the10 represents the tunnel ID, and the [10.0.0.3] represents the extended tunnel ID.

```
*May 12 19:46:14.267: RSVP-HA: session 65.39.97.4_18698[0.0.0.0]:rsvp_ha_read_lsp_head_info:
 Read LSP Head info: tun_id: 10
*May 12 19:46:14.267: RSVP-HA: session 10.0.0.1_10[0.0.0.0]: rsvp_ha_db_entry_find: lsp_head
 entry found
*May 12 19:46:14.267: rsvp_ha_read_lsp_head_info: entry found
*May 12 19:46:14.267: RSVP-HA:rsvp_ha_read_lsp_head_info: Read LSP Head info: tun_id: 10
*May 12 19:46:14.267: RSVP-HA: session 10.221.123.48_10[0.0.0.0]: rsvp_ha_db_entry_find:
lsp_head entry found
*May 12 19:46:14.267: rsvp_ha_read_lsp_head_info: entry found
*May 12 19:46:15.995: %SYS-5-CONFIG_I: Configured from console by console
*May 12 19:46:20.803: RSVP-HA: 10.0.0.3_61->10.0.0.9_10[10.0.0.3]: rsvp_ha_db_entry_find:
lsp entry found
*May 12 19:46:20.803: rsvp_ha_read_generic_info: lsp entry found
*May 12 19:46:20.807: RSVP-HA: session 10.0.0.9_10[0.0.0.0]:rsvp_ha_write_generic_info:
Writing lsp_head info
*May 12 19:46:20.807: RSVP-HA: session 10.0.0.9_10[0.0.0.0]: rsvp_ha_db_entry_find: lsp_head
 entry not found
```

```
*May 12 19:46:20.807: RSVP-HA: session 10.0.0.9_10[0.0.0.0]:
rsvp_ha_handle_wr_entry_not_found:
entry not found, type =lsp_head, action: Add
*May 12 19:46:20.807: RSVP-HA: session 10.0.0.9_10[0.0.0.0]: rsvp_ha_db_entry_create: Created
 lsp_head entry
*May 12 19:46:20.807: RSVP-HA: session 10.0.0.9_10[0.0.0.0]:rsvp_ha_set_entry_state: None
-> Send-Pending
*May 12 19:46:20.807: RSVP-HA: session 10.0.0.9_10[0.0.0.0]: rsvp_ha_db_wavl_entry_insert:
 Inserted entry into lsp_head Write DB, Send_Pending tree
*May 12 19:46:20.807: RSVP-HA: session 10.0.0.9_10[0.0.0.0]:rsvp_ha_fsm_wr_event_add_entry:
 add lsp_head entry to Write DB
*May 12 19:46:20.807: RSVP-HA: 10.0.0.3_61->10.0.0.9_10[10.0.0.3]:
rsvp_ha_write_generic_info: Writing lsp info
*May 12 19:46:20.807: RSVP-HA: 10.0.0.3_61->10.0.0.9_10[10.0.0.3]: rsvp_ha_db_entry_find:
lsp entry not found
*May 12 19:46:20.807: RSVP-HA: 10.0.0.3_61->10.0.0.9_10[10.0.0.3]:
rsvp_ha_handle_wr_entry_not_found: entry not found, type =lsp, action: Add
*May 12 19:46:20.807: RSVP-HA: 10.0.0.3_61->10.0.0.9_10[10.0.0.3]: rsvp_ha_db_entry_create:
 Created lsp entry
*May 12 19:46:20.807: RSVP-HA:10.0.0.3_61->10.0.0.9_10[10.0.0.3]:
rsvp_ha_set_entry_state: None -> Send-Pending
*May 12 19:46:20.807: RSVP-HA: 10.0.0.3_61->10.0.0.9_10[10.0.0.3]:
rsvp_ha_db_wavl_entry_insert: Inserted entry into lsp Write DB, Send_Pending tree
*May 12 19:46:20.807: RSVP-HA: 10.0.0.3_61->10.0.0.9_10[10.0.0.3]:
rsvp_ha_fsm_wr_event_add_entry: add lsp entry to Write DB
*May 12 19:46:20.807: rsvp_ha_rd_remove_lsp_head_info: Event RD: remove lsp_head_info
*May 12 19:46:20.807: RSVP-HA: session 10.27.90.140_10[0.0.0.0]:
rsvp_ha_db_entry_find: lsp_head entry found
*May 12 19:46:20.807: RSVP-HA: session 10.0.0.9_10[0.0.0.0]: rsvp_ha_db_wavl_entry_remove:
 Removed entry from lsp_head Read DB, Checkpointed tree
*May 12 19:46:20.807: RSVP-HA: session 10.0.0.9_10[0.0.0.0]: rsvp_ha_db_entry_free: Freeing
 lsp_head entry
*May 12 19:46:20.807: RSVP-HA: session 10.0.0.9_10[0.0.0.0]:rsvp_ha_set_entry_state:
Checkpointed -> None
.
.
.
```

The following example shows how to turn debugging off for this command:

```
Router# no debug ip rsvp high-availability all
RSVP HA all debugging is off
```

Related Commands

Command	Description
debug ip rsvp sso	Displays debugging output for RSVP signalling when the graceful restart feature is configured.
debug mpls traffic-eng ha sso	Displays debugging output for MPLS traffic engineering HA activities during the graceful switchover from an active RP to a redundant standby RP.

debug ip rsvp p2mp

To display status messages for Resource Reservation Protocol (RSVP) point-to-multipoint (P2MP) events, use the **debug ip rsvp p2mp**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip rsvp p2mp

no debug ip rsvp p2mp

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(33)SRE	This command was introduced. For P2MP traffic engineering tunnels, the output displays the status of the sublabel switched paths (sub-LSPs).

Usage Guidelines

If the P2MP tunnel is not up, issue this command and the **debug ip rsvp signalling** command and examine the output to determine if there is a problem with the configuration.

Use this command with a small number of tunnels or RSVP reservations or use the RSVP debug message filter to limit the amount of data. Too much data can overload the CPU.

Examples

The following example shows status messages as a P2MP sub-LSP is signaled:

```
Router# debug ip rsvp p2mp
RSVP p2mp debugging is on
IP RSVP debugging is on for:
   p2mp
Router (config)# interface tunnel100
Router (config-if)# no shutdown
06:56:21: RSVP: 10.1.0.1_134[Src/1]->10.2.0.1_100[Src] {13}: First Sub-LSP, accept Path.
06:56:21: RSVP: 10.1.0.1_134[Src/2]->10.3.0.1_100[Src] {13}: Sibling Sub-LSP received with
 consistent signalling attributes, accept Path
06:56:21: RSVP: 10.1.0.1_134[Src/3]->10.4.0.1_100[Src] {13}: Sibling Sub-LSP received with
 consistent signalling attributes, accept Path
06:56:22: RSVP: 10.1.0.1_134[Src/1]->10.2.0.1_100[Src] {13}: First Sub-LSP, accept Resv.
06:56:22: %LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel100, changed state to up
06:56:22: RSVP: 10.1.0.1_134[Src/3]->10.4.0.1_100[Src] {13}: Sibling Sub-LSP received with
 consistent signalling attributes, accept Resv
06:56:22: RSVP: 10.1.0.1_134[Src/2]->10.3.0.1_100[Src] {13}: Sibling Sub-LSP received with
 consistent signalling attributes, accept Resv
```

Related Commands

Command	Description
debug ip rsvp signalling	Displays RSVP signalling (PATH and RESV) messages.
show ip rsvp reservation	Displays RSVP PATH-related receiver information currently in the database.
show ip rsvp sender	Displays RSVP RESV-related receiver information currently in the database.

debug ip rsvp policy

To display debugging messages for Resource Reservation Protocol (RSVP) policy processing, use the **debug ip rsvp policy**command in privileged EXEC mode. To disable debugging output, use the **no**form of this command.

debug ip rsvp policy

no debug ip rsvp policy

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging for RSVP policy processing is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.1(1)T	This command was introduced.
12.0(23)S	This command was integrated into Cisco IOS Release 12.0(23)S.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

You might find it useful to enable the debug cops command when you are using the debug ip rsvp policy command. Together, these commands generate a complete record of the policy process.

Examples

The following example uses only the **debug ip rsvp policy** command:

```
Router-1# debug ip rsvp policy
RSVP_POLICY debugging is on
02:02:14:RSVP-POLICY:Creating outbound policy IDB entry for Ethernet2/0 (61E6AB38)
02:02:14:RSVP-COPS:COPS query for Path message, 10.31.0.1_44->10.33.0.1_44
02:02:14:RSVP-POLICY:Building incoming Path context
02:02:14:RSVP-POLICY:Building outgoing Path context on Ethernet2/0
02:02:14:RSVP-POLICY:Build REQ message of 216 bytes
02:02:14:RSVP-POLICY:Message sent to PDP
02:02:14:RSVP-COPS:COPS engine called us with reason2, handle 6202A658
02:02:14:RSVP-COPS:Received decision message
02:02:14:RSVP-POLICY:Received decision for Path message
02:02:14:RSVP-POLICY:Accept incoming message
02:02:14:RSVP-POLICY:Send outgoing message to Ethernet2/0
02:02:14:RSVP-POLICY:Replacement policy object for path-in context
02:02:14:RSVP-POLICY:Replacement TSPEC object for path-in context
02:02:14:RSVP-COPS:COPS report for Path message, 10.31.0.1_44->10.33.0.1_44
02:02:14:RSVP-POLICY:Report sent to PDP
02:02:14:RSVP-COPS:COPS report for Path message, 10.31.0.1_44->10.33.0.1_44
```

The following example uses both the **debug ip rsvp policy** and the **debug cops** commands:

```
Router-1# debug ip rsvp policy
RSVP_POLICY debugging is on
Router-1# debug cops
COPS debugging is on
02:15:14:RSVP-POLICY:Creating outbound policy IDB entry for Ethernet2/0 (61E6AB38)
02:15:14:RSVP-COPS:COPS query for Path message, 10.31.0.1_44->10.33.0.1_44
02:15:14:RSVP-POLICY:Building incoming Path context
02:15:14:RSVP-POLICY:Building outgoing Path context on Ethernet2/0
02:15:14:RSVP-POLICY:Build REQ message of 216 bytes
02:15:14:COPS:** SENDING MESSAGE **
    COPS HEADER:Version 1, Flags 0, Opcode 1 (REQ), Client-type:1, Length:216
    HANDLE (1/1) object. Length:8.    00 00 22 01
    CONTEXT (2/1) object. Length:8.   R-type:5.     M-type:1
    IN_IF (3/1) object. Length:12.   Address:10.1.2.1.    If_index:4
    OUT_IF (4/1) object. Length:12.   Address:10.33.0.1.    If_index:3
    CLIENT SI (9/1) object. Length:168.   CSI data:
02:15:14: SESSION              type 1 length 12:
02:15:14:      Destination 10.33.0.1, Protocol_Id 17, Don't Police , DstPort 44
02:15:14: HOP                  type 1 length 12:0A010201
02:15:14:                                      :00000000
02:15:14: TIME_VALUES          type 1 length 8 :00007530
02:15:14: SENDER_TEMPLATE      type 1 length 12:
02:15:14:      Source 10.31.0.1, udp_source_port 44
02:15:14: SENDER_TSPEC         type 2 length 36:
02:15:14:      version=0, length in words=7
02:15:14:      Token bucket fragment (service_id=1, length=6 words
02:15:14:          parameter id=127, flags=0, parameter length=5
02:15:14:          average rate=1250 bytes/sec, burst depth=10000 bytes
02:15:14:          peak rate   =1250000 bytes/sec
02:15:14:          min unit=0 bytes, max unit=1514 bytes
02:15:14: ADSPEC               type 2 length 84:
02:15:14: version=0  length in words=19
02:15:14: General Parameters  break bit=0  service length=8
02:15:14:                                              IS Hops:1
02:15:14:                 Minimum Path Bandwidth (bytes/sec):1250000
02:15:14:                        Path Latency (microseconds):0
02:15:14:                                             Path MTU:1500
02:15:14: Guaranteed Service  break bit=0  service length=8
02:15:14:                          Path Delay (microseconds):192000
02:15:14:                         Path Jitter (microseconds):1200
02:15:14:            Path delay since shaping (microseconds):192000
02:15:14:           Path Jitter since shaping (microseconds):1200
02:15:14: Controlled Load Service  break bit=0  service length=0
02:15:14:COPS:Sent 216 bytes on socket,
02:15:14:RSVP-POLICY:Message sent to PDP
02:15:14:COPS:Message event!
02:15:14:COPS:State of TCP is 4
02:15:14:In read function
02:15:14:COPS:Read block of 96 bytes, num=104 (len=104)
02:15:14:COPS:** RECEIVED MESSAGE **
    COPS HEADER:Version 1, Flags 1, Opcode 2 (DEC), Client-type:1, Length:104
    HANDLE (1/1) object. Length:8.    00 00 22 01
    CONTEXT (2/1) object. Length:8.   R-type:1.     M-type:1
    DECISION (6/1) object. Length:8.   COMMAND cmd:1, flags:0
    DECISION (6/3) object. Length:56.   REPLACEMENT  00 10 0E 01 61 62 63 64 65 66 67
68 69 6A 6B 6C 00 24 0C 02 00
00 00 07 01 00 00 06 7F 00 00 05 44 9C 40 00 46 1C 40 00 49 98
96 80 00 00 00 C8 00 00 01 C8
    CONTEXT (2/1) object. Length:8.   R-type:4.     M-type:1
    DECISION (6/1) object. Length:8.   COMMAND cmd:1, flags:0
02:15:14:Notifying client (callback code 2)
02:15:14:RSVP-COPS:COPS engine called us with reason2, handle 6202A104
02:15:14:RSVP-COPS:Received decision message
02:15:14:RSVP-POLICY:Received decision for Path message
02:15:14:RSVP-POLICY:Accept incoming message
02:15:14:RSVP-POLICY:Send outgoing message to Ethernet2/0
02:15:14:RSVP-POLICY:Replacement policy object for path-in context
02:15:14:RSVP-POLICY:Replacement TSPEC object for path-in context
02:15:14:RSVP-COPS:COPS report for Path message, 10.31.0.1_44->10.33.0.1_44
02:15:14:COPS:** SENDING MESSAGE **
```

```
    COPS HEADER:Version 1, Flags 1, Opcode 3 (RPT), Client-type:1, Length:24
    HANDLE (1/1) object. Length:8.    00 00 22 01
    REPORT (12/1) object. Length:8.   REPORT type COMMIT (1)
02:15:14:COPS:Sent 24 bytes on socket,
02:15:14:RSVP-POLICY:Report sent to PDP
02:15:14:Timer for connection entry is zero
02:15:14:RSVP-COPS:COPS report for Path message, 10.31.0.1_44->10.33.0.1_44
```

Related Commands

Command	Description
debug cops	Displays debugging messages for COPS processing.

debug ip rsvp rate-limit

To display debugging messages for Resource Reservation Protocol (RSVP) rate-limiting events, use the **debug ip rsvp rate-limit**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip rsvp rate-limit

no debug ip rsvp rate-limit

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(13)T	This command was introduced.
12.0(24)S	This command was integrated into Cisco IOS Release 12.0(24)S.

Examples

The following command shows how to enable debugging for RSVP rate-limiting and message manager events:

```
Router# debug ip rsvp rate-limit
RSVP rate-limit debugging is on
Router# debug ip rsvp msg-mgr
RSVP msg-mgr debugging is on
```

In the following display, RSVP process information including messages, timers, neighbors IP addresses, and message IDs, appear:

```
01:00:19:RSVP-RATE-LIMIT:rsvp_msg_pacing_send_message
01:00:19:RSVP-MSG-MGR (140.4.4.2):Starting timer msg-pacing interval 20
01:00:19:RSVP-MSG-MGR (140.4.4.2):Enqueue element 27000405 of type 3 on msg-pacing TAIL
01:00:19:RSVP-RATE-LIMIT:rsvp_msg_pacing_timer - timer expired
01:00:19:RSVP-MSG-MGR (140.4.4.2):Dequeueing element 27000405 of type 3 from msg-pacing
01:00:19:RSVP-RATE-LIMIT:rsvp_msg_pacing_send_qe:sending psb (qe 27000405)
01:00:21:%LINK-3-UPDOWN:Interface Tunnel100, changed state to up
01:00:22:%LINEPROTO-5-UPDOWN:Line protocol on Interface Tunnel100, changed state to up
01:01:03:RSVP-RATE-LIMIT:rsvp_msg_pacing_send_message
01:01:03:RSVP-MSG-MGR (140.4.4.2):Starting timer msg-pacing interval 20
01:01:03:RSVP-MSG-MGR (140.4.4.2):Enqueue element 27000405 of type 3 on msg-pacing TAIL
01:01:03:RSVP-RATE-LIMIT:rsvp_msg_pacing_timer - timer expired
01:01:03:RSVP-MSG-MGR (140.4.4.2):Dequeueing element 27000405 of type 3 from msg-pacing
01:01:03:RSVP-RATE-LIMIT:rsvp_msg_pacing_send_qe:sending psb (qe 27000405)
01:01:42:RSVP-RATE-LIMIT:rsvp_msg_pacing_send_message
01:01:42:RSVP-MSG-MGR (140.4.4.2):Starting timer msg-pacing interval 20
01:01:42:RSVP-MSG-MGR (140.4.4.2):Enqueue element 27000405 of type 3 on msg-pacing TAIL
01:01:42:RSVP-RATE-LIMIT:rsvp_msg_pacing_timer - timer expired
01:01:42:RSVP-MSG-MGR (140.4.4.2):Dequeueing element 27000405 of type 3 from msg-pacing
```

```
01:01:42:RSVP-RATE-LIMIT:rsvp_msg_pacing_send_qe:sending psb (qe 27000405)
01:02:09:RSVP-RATE-LIMIT:rsvp_msg_pacing_send_message
01:02:09:RSVP-MSG-MGR (140.4.4.2):Starting timer msg-pacing interval 20
01:02:09:RSVP-MSG-MGR (140.4.4.2):Enqueue element 27000405 of type 3 on msg-pacing TAIL
01:02:09:RSVP-RATE-LIMIT:rsvp_msg_pacing_timer - timer expired
01:02:09:RSVP-MSG-MGR (140.4.4.2):Dequeueing element 27000405 of type 3 from msg-pacing
01:02:09:RSVP-RATE-LIMIT:rsvp_msg_pacing_send_qe:sending psb (qe 27000405)
```

Related Commands

Command	Description
ip rsvp signalling rate-limit	Controls the transmission rate for RSVP messages sent to a neighboring router during a specified interval.
show debug	Displays active debug output.

debug ip rsvp reliable-msg

To display debugging messages for Resource Reservation Protocol (RSVP) reliable messages events, use the **debug ip rsvp reliable-msg**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip rsvp reliable-msg

no debug ip rsvp reliable-msg

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(13)T	This command was introduced.
12.0(24)S	This command was integrated into Cisco IOS Release 12.0(24)S.

Examples

The following command shows how to enable debugging for RSVP reliable messages events:

```
Router# debug ip rsvp reliable-msg
RSVP reliable-msg debugging is on
```

In the following display, message IDs, acknowledgments (ACKs), and message processes including retransmissions, appear:

```
01:07:37:RSVP-RMSG:Inserted msg id(0x46, 0x48000403) on local msgid db
01:07:37:RSVP-RMSG:rsvp_rmsg_process_acks, Handle:000C1701 neighbor:140.4.4.2
01:07:37:RSVP-RMSG:max_ids:1 q_sz:1 msg_sz:1500 ids_len:1432 num_objs:0 obj_len:0
nbr:140.4.4.2
01:07:39:%LINK-3-UPDOWN:Interface Tunnel100, changed state to up
01:07:40:%LINEPROTO-5-UPDOWN:Line protocol on Interface Tunnel100, changed state to up
01:08:07:RSVP-RMSG:rsvp_rmsg_process_acks, Handle:000C1701 neighbor:140.4.4.2
01:08:07:RSVP-RMSG:max_ids:1 q_sz:1 msg_sz:1500 ids_len:1432 num_objs:0 obj_len:0
nbr:140.4.4.2
01:08:37:RSVP-RMSG:max_ids:1 q_sz:1 msg_sz:1500 ids_len:1424 num_objs:1 obj_len:8
nbr:140.4.4.2
01:08:37:RSVP-RMSG:rsvp_rmsg_process_immediate_tmb, Handle:2D000404 neighbor:140.4.4.2
01:08:37:RSVP-RMSG:Inserted msg id(0x47, 0x2D000404) on local msgid db
01:08:37:RSVP-RMSG:current queue:immed next_queue:rxmt-1 (qe 2D000404s)
01:08:37:RSVP-RMSG:rsvp_rmsg_process_acks, Handle:000C1701 neighbor:140.4.4.2
01:08:37:RSVP-RMSG:max_ids:1 q_sz:1 msg_sz:1500 ids_len:1432 num_objs:0 obj_len:0
nbr:140.4.4.2
01:08:38:RSVP-RMSG:rsvp_rmsg_process_rxmt_tmb, Handle:2D000404 neighbor:140.4.4.2
01:08:38:RSVP-RMSG:An ack was received for tmb 2D000404 on neighbor 140.4.4.2
01:09:07:RSVP-RMSG:max_ids:1 q_sz:1 msg_sz:1500 ids_len:1424 num_objs:1 obj_len:8
nbr:140.4.4.2
```

```
01:09:07:RSVP-RMSG:rsvp_rmsg_process_immediate_tmb, Handle:2E000404 neighbor:140.4.4.2
01:09:07:RSVP-RMSG:Inserted msg id(0x48, 0x2E000404) on local msgid db
01:09:07:RSVP-RMSG:current queue:immed next_queue:rxmt-1 (qe 2E000404s)
01:09:07:RSVP-RMSG:rsvp_rmsg_process_acks, Handle:000C1701 neighbor:140.4.4.2
01:09:07:RSVP-RMSG:max_ids:1 q_sz:1 msg_sz:1500 ids_len:1432 num_objs:0 obj_len:0
nbr:140.4.4.2
01:09:08:RSVP-RMSG:rsvp_rmsg_process_rxmt_tmb, Handle:2E000404 neighbor:140.4.4.2
01:09:08:RSVP-RMSG:An ack was received for tmb 2E000404 on neighbor 140.4.4.2
01:09:37:RSVP-RMSG:max_ids:1 q_sz:1 msg_sz:1500 ids_len:1424 num_objs:1 obj_len:8
nbr:140.4.4.2
01:09:37:RSVP-RMSG:rsvp_rmsg_process_immediate_tmb, Handle:2F000404 neighbor:140.4.4.2
01:09:37:RSVP-RMSG:Inserted msg id(0x49, 0x2F000404) on local msgid db
01:09:37:RSVP-RMSG:current queue:immed next_queue:rxmt-1 (qe 2F000404s)
01:09:37:RSVP-RMSG:rsvp_rmsg_process_acks, Handle:000C1701 neighbor:140.4.4.2
01:09:37:RSVP-RMSG:max_ids:1 q_sz:1 msg_sz:1500 ids_len:1432 num_objs:0 obj_len:0
nbr:140.4.4.2
01:09:38:RSVP-RMSG:rsvp_rmsg_process_rxmt_tmb, Handle:2F000404 neighbor:140.4.4.2
01:09:38:RSVP-RMSG:An ack was received for tmb 2F000404 on neighbor 140.4.4.2
```

Related Commands

Command	Description
ip rsvp signalling refresh reduction	Enables refresh reduction.
show debug	Displays active debug output.

debug ip rsvp sbm

To display detailed information about Subnetwork Bandwidth Manager (SBM) messages only, and SBM and Designated Subnetwork Bandwidth Manager (DSBM) state transitions, use the **debug ip rsvp sbm**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip rsvp sbm

no debug ip rsvp sbm

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(5)T	This command was introduced.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

The **debug ip rsvp sbm**command provides information about messages received, minimal detail about the content of these messages, and information about state transitions.

Examples

The following example shows the detailed debug information about SBM and the SBM and DSBM state transitions that is available when you enable debug mode through the **debug ip rsvp sbm** command:

```
Router# debug ip rsvp sbm
RSVP debugging is on
router2#
*Dec 31 16:45:34.659: RSVP: send I_AM_DSBM message from 145.2.2.150
*Dec 31 16:45:34.659: RSVP: IP to 224.0.0.17 length=88 checksum=9385 (Ethernet2)
*Dec 31 16:45:34.659:  RSVP: version:1 flags:0000 type:I_AM_DSBM cksum:9385
                         ttl:254    reserved:0 length:88
*Dec 31 16:45:34.659:  DSBM_IP_ADDR       type 1 length 8 : 91020296
*Dec 31 16:45:34.659:  HOP_L2             type 1 length 12: 00E01ECE
*Dec 31 16:45:34.659:                                     : 0F760000
*Dec 31 16:45:34.659:  SBM_PRIORITY       type 1 length 8 : 0029B064
*Dec 31 16:45:34.659:  DSBM_TIMERS        type 1 length 8 : 00000F05
*Dec 31 16:45:34.659:  SBM_INFO           type 1 length 44: 00000000
*Dec 31 16:45:34.659:                                     : 00240C02 00000007
*Dec 31 16:45:34.659:                                     : 01000006 7F000005
*Dec 31 16:45:34.659:                                     : 00000000 00000000
*Dec 31 16:45:34.663:                                     : 00000000 00000000
*Dec 31 16:45:34.663:                                     : 00000000
*Dec 31 16:45:34.663:
```

Related Commands

Command	Description
debug ip rsvp	Displays information about SBM message processing, the DSBM election process, and RSVP message processing.
debug ip rsvp authentication	Displays detailed information about RSVP and SBM.
ip rsvp dsbm-candidate	Configures an interface as a DSBM candidate.

debug ip rsvp sso

To display debugging output for Resource Reservation Protocol (RSVP) signaling when the graceful restart feature is configured, use the **debug ip rsvp sso**command in privileged EXEC mode. To disable debugging, use the **no** form of this command.

debug ip rsvp sso

no debug ip rsvp sso

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging is disabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(33)SRA	This command was introduced.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.

Usage Guidelines

This command displays debugging output from RSVP signaling during and after the Route Processor (RP) stateful switchover when system control and routing protocol execution is transferred from the active RP to the redundant standby RP. The SSO process occurs when the active router becomes unavailable, so that no interruption of network services occurs. The command displays information about the activities that RSVP performs when you configure a graceful restart, such as:

- Writing checkpointing information into the write database when a new traffic engineering (TE) label switched path (LSP) is signaled on the active RP
- Recovering the LSP checkpointed information from the read database after SSO
- Displaying information about LSPs not recovered

Examples

The following is sample output from the **debug ip rsvp sso**command that was displayed during a successful SSO on the standby router as it became active:

```
Router# debug ip rsvp sso
RSVP sso debugging is on
Router#
```

Note The prefix in the debug output is composed of LSP 5-tuples in the following format: 10.0.0.3_61->10.0.0.9_10[10.0.0.3]. The 10.0.0.3 represents the source address, the 61 represents the LSP ID, the 10.0.0.9 represents the tunnel destination (tunnel tail), the10 represents the tunnel ID, and the [10.0.0.3] represents the extended tunnel ID.

```
*May 12 20:12:38.175: RSVP-HA: begin recovery, send msg to RSVP
*May 12 20:12:38.175: RSVP: 10.0.0.3_61->10.0.0.9_10[10.0.0.3]: event: new Path received
during RSVP or IGP recovery period
*May 12 20:12:38.175: RSVP: 10.0.0.3_61->10.0.0.9_10[10.0.0.3]:
rsvp_ha_sb_event_new_path_received: lsp_info found, attempt to recover lsp
*May 12 20:12:38.175: RSVP: 10.0.0.3_61->10.0.0.9_10[10.0.0.3]: set psb_is_recovering flag
*May 12 20:12:38.179: RSVP: 10.0.0.3_61->10.0.0.9_10[10.0.0.3]:rsvp_ha_sb_set_path_info:
Recovering: Set next_hop and next_idb in psb
*May 12 20:12:38.179: RSVP:
10.0.0.3_61->10.0.0.9_10[10.0.0.3]:rsvp_ha_mark_lsp_if_recoverable: LSP is recoverable (ERO
 expansion. not needed)
*May 12 20:12:38.179: RSVP-HA: rsvp_ha_sb_handle_recovery_start: Recovery period start: set
 GR recovery time.
*May 12 20:12:38.179: RSVP_HA: checkpoint hello_globals_info
*May 12 20:12:38.179: RSVP-HELLO: rsvp_ha_update_all_gr_hi: Updating all GR HIs with new
src_instance
*May 12 20:12:38.183: RSVP: 10.0.0.3_61->10.0.0.9_10[10.0.0.3]: prevent populating output;
 LSP is recovering
*May 12 20:12:38.187: RSVP: 10.0.0.3_61->10.0.0.9_10[10.0.0.3]: prevent populating output;
 LSP is recovering
*May 12 20:12:38.939: RSVP: 10.0.0.3_61->10.0.0.9_10[10.0.0.3]:
rsvp_ha_sb_event_new_resv_received: event: Resv for LSP received during recovery period
*May 12 20:12:38.943: RSVP: 10.0.0.3_61->10.0.0.9_10[10.0.0.3]:
rsvp_ha_event_lsp_create_head: psb found
*May 12 20:12:38.943: RSVP: 10.0.0.3_61->10.0.0.9_10[10.0.0.3]:
rsvp_ha_event_lsp_create_head: event: LSP created at head-end, try to checkpoint it
*May 12 20:12:38.943: RSVP: 10.0.0.3_61->10.0.0.9_10[10.0.0.3]: LSP was checkpointed
*May 12 20:12:38.943: RSVP-HA: 10.0.0.3_61->10.0.0.9_10[10.0.0.3]:
rsvp_ha_sb_event_lsp_head_recovered: event: LSP head was recovered
*May 12 20:12:38.943: RSVP-HA: recovery period over, send msg to RSVP
*May 12 20:12:38.947: RSVP-HA: rsvp_ha_sb_handle_recovery_end: Deleting state for LSPs not
 recovered
Router#
```

The following example shows how to turn debugging off for this command:

```
Router# no debug ip rsvp sso
RSVP sso debugging is off
```

Related Commands

Command	Description
debug ip rsvp high-availability	Displays debugging output for RSVP-TE HA activities that improve the accessibility of network resources.
debug mpls traffic-eng ha sso	Displays debugging output for MPLS traffic engineering HA activities during the graceful switchover from an active RP to a redundant standby RP.

debug ip rsvp summary-refresh

To display debugging messages for Resource Reservation Protocol (RSVP) summary-refresh messages events, use the **debug ip rsvp summary-refresh**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip rsvp summary-refresh

no debug ip rsvp summary-refresh

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(13)T	This command was introduced.
12.0(24)S	This command was integrated into Cisco IOS Release 12.0(24)S.

Examples

The following command shows how to enable debugging for RSVP summary-refresh messages events:

```
Router# debug ip rsvp summary-refresh
RSVP summary-refresh debugging is on
```

In the following output, the IP addresses, the interfaces, the types of RSVP messages (Path and Resv), message IDs, and epoch identifiers (for routers) for which RSVP summary-refresh events occur are shown:

```
01:11:00:RSVP-SREFRESH:Incoming message from nbr 140.4.4.2 with epoch:0xE1A1B7 msgid:0x84
on Ethernet1
01:11:00:RSVP-SREFRESH 140.20.1.1_18->140.75.1.1_100[140.20.1.1]:Created msgid 0x84 for nbr
 140.4.4.2
01:11:02:%LINK-3-UPDOWN:Interface Tunnel100, changed state to up
01:11:03:%LINEPROTO-5-UPDOWN:Line protocol on Interface Tunnel100, changed state to up
01:11:30:RSVP-SREFRESH:140.20.1.1_18->140.75.1.1_100[140.20.1.1]:Path, ID:0x4C :Start using
 Srefresh to 140.4.4.2
01:11:31:RSVP-SREFRESH:Incoming message from nbr 140.4.4.2 with epoch:0xE1A1B7 msgid:0x84
on Ethernet1
01:11:31:RSVP-SREFRESH:State exists for nbr:140.4.4.2 epoch:0xE1A1B7 msgid:0x84
01:12:00:RSVP-SREFRESH:Preparing to Send Srefresh(es) to 140.4.4.2, 1 IDs Total
01:12:00:RSVP-SREFRESH:Sending 1 IDs in this Srefresh
01:12:00:RSVP-SREFRESH:140.20.1.1_18->140.75.1.1_100[140.20.1.1]:Path, ID:0x4C
01:12:01:RSVP-SREFRESH:Incoming message from nbr 140.4.4.2 with epoch:0xE1A1B7 msgid:0x86
on Ethernet1
01:12:01:RSVP-SREFRESH:Rec'd 1 IDs in Srefresh from 140.4.4.2 (on Ethernet1), epoch:0xE1A1B7
 msgid:0x86
01:12:01:RSVP-SREFRESH:140.20.1.1_18->140.75.1.1_100[140.20.1.1]:Resv, ID:0x84
01:12:30:RSVP-SREFRESH:Preparing to Send Srefresh(es) to 140.4.4.2, 1 IDs Total
01:12:30:RSVP-SREFRESH:Sending 1 IDs in this Srefresh
```

```
01:12:30:RSVP-SREFRESH:140.20.1.1_18->140.75.1.1_100[140.20.1.1]:Path, ID:0x4C
01:12:31:RSVP-SREFRESH:Incoming message from nbr 140.4.4.2 with epoch:0xE1A1B7 msgid:0x88
on Ethernet1
01:12:31:RSVP-SREFRESH:Rec'd 1 IDs in Srefresh from 140.4.4.2 (on Ethernet1), epoch:0xE1A1B7
 msgid:0x88
01:12:31:RSVP-SREFRESH:140.20.1.1_18->140.75.1.1_100[140.20.1.1]:Resv, ID:0x84
01:13:00:RSVP-SREFRESH:Preparing to Send Srefresh(es) to 140.4.4.2, 1 IDs Total
01:13:00:RSVP-SREFRESH:Sending 1 IDs in this Srefresh
01:13:00:RSVP-SREFRESH:140.20.1.1_18->140.75.1.1_100[140.20.1.1]:Path, ID:0x4C
01:13:01:RSVP-SREFRESH:Incoming message from nbr 140.4.4.2 with epoch:0xE1A1B7 msgid:0x8A
on Ethernet1
01:13:01:RSVP-SREFRESH:Rec'd 1 IDs in Srefresh from 140.4.4.2 (on Ethernet1), epoch:0xE1A1B7
 msgid:0x8A
01:13:01:RSVP-SREFRESH:140.20.1.1_18->140.75.1.1_100[140.20.1.1]:Resv, ID:0x84
```

Note In the preceding output, notice the message IDs that correspond to Path or Resv state being refreshed. Because the entire message does not have to be transmitted, there is less data and network performance is improved.

Related Commands

Command	Description
ip rsvp signalling refresh reduction	Enables refresh reduction.
show debug	Displays active debug output.

debug ip rsvp traffic-control

To display debugging messages for compression-related events, use the **debug ip rsvp traffic-control command in**privilegedEXEC mode. To disable debugging output, use the **no** form of this command.

debug ip rsvp traffic-control

no debug ip rsvp traffic-control

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.0	This command was introduced.
12.2(15)T	This command was modified. The command output was modified to include compression-related events.
12.0(24)S	This command was integrated into Cisco IOS Release 12.0(24)S.
12.2(14)S	This command was integrated into Cisco IOS Release 12.2(14)S.
12.2(18)SXF2	This command was integrated into Cisco IOS Release 12.2(18)SXF2.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
Cisco IOS XE Release 2.6	This command was integrated into Cisco IOS XE Release 2.6.

Usage Guidelines

Use the **debug ip rsvp traffic-control** command to troubleshoot compression-related problems.

Examples

The following example from the **debug ip rsvp traffic-control** command shows that compression was successfully predicted:

```
Router# debug ip rsvp traffic-control
RSVP debugging is on
Router# show debugging
00:44:49: RSVP-TC: Attempting to install QoS for rsb 62CC66F0
00:44:49: RSVP-TC: Adding new tcsb 02000406 for rsb 62CC66F0
00:44:49: RSVP-TC: Assigning WFQ QoS (on FR VC 101) to tcsb 02000406
00:44:49: RSVP-TC: Predicted compression for TCSB 2000406:
00:44:49: RSVP-TC:   method      = rtp
00:44:49: RSVP-TC:   context ID  = 2
00:44:49: RSVP-TC:   factor      = 82 percent
00:44:49: RSVP-TC:   bytes-saved = 36 bytes
00:44:49: RSVP-TC: Bandwidth check: requested bw=65600 old bw=0
00:44:49: RSVP-TC: RSVP bandwidth is available
```

```
00:44:49: RSVP-TC: Consulting policy for tcsb 02000406
00:44:49: RSVP-TC: Policy granted QoS for tcsb 02000406
00:44:49: RSVP-TC: Requesting QoS for tcsb 02000406
00:44:49: RSVP-TC:    ( r = 8200       bytes/s   M = 164        bytes
00:44:49: RSVP-TC:      b = 328        bytes     m = 164        bytes )
00:44:49: RSVP-TC:      p = 10000      bytes/s   Service Level = priority
00:44:49: RSVP-WFQ: Update for tcsb 02000406 on FR PVC dlci 101 on Se3/0
00:44:49: RSVP-WFQ: Admitted 66 kbps of bandwidth
00:44:49: RSVP-WFQ: Allocated PRIORITY queue 24
00:44:49: RSVP-TC: Allocation succeeded for tcsb 02000406
```

The following example from the **debug ip rsvp traffic-control** command shows that compression was unsuccessfully predicted because no compression context IDs were available:

```
Router# debug ip rsvp traffic-control
RSVP debugging is on
Router# show debugging
00:10:16:RSVP-TC:Attempting to install QoS for rsb 62CED62C
00:10:16:RSVP-TC:Adding new tcsb 01000421 for rsb 62CED62C
00:10:16:RSVP-TC:Assigning WFQ QoS (on FR VC 101) to tcsb 01000421
00:10:16:RSVP-TC:sender's flow is not rtp compressible for TCSB 1000421
00:10:16:         reason: no contexts available
00:10:16:RSVP-TC:sender's flow is not udp compressible for TCSB 1000421
00:10:16:         reason: no contexts available
00:10:16:RSVP-TC:Bandwidth check:requested bw=80000 old bw=0
00:10:16:RSVP-TC:RSVP bandwidth is available
00:10:16:RSVP-TC:Consulting policy for tcsb 01000421
00:10:16:RSVP-TC:Policy granted QoS for tcsb 01000421
00:10:16:RSVP-TC:Requesting QoS for tcsb 01000421
00:10:16:RSVP-TC:   ( r = 10000      bytes/s   M = 200        bytes
00:10:16:RSVP-TC:     b = 400        bytes     m = 200        bytes )
00:10:16:RSVP-TC:     p = 10000      bytes/s   Service Level = priority
00:10:16:RSVP-WFQ:Update for tcsb 01000421 on FR PVC dlci 101 on Se3/0
00:10:16:RSVP-WFQ:Admitted 80 kbps of bandwidth
00:10:16:RSVP-WFQ:Allocated PRIORITY queue 24
00:10:16:RSVP-TC:Allocation succeeded for tcsb 01000421
```

Related Commands

Command	Description
show debugging	Displays active debugging output.

debug ip rsvp wfq

To display debugging messages for the weighted fair queue (WFQ), use the **debug ip rsvp wfq**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip rsvp wfq

no debug ip rsvp wfq

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.1(3)T	This command was introduced.
12.0(24)S	This command was integrated into Cisco IOS Release 12.0(24)S.
12.2(14)S	This command was integrated into Cisco IOS Release 12.2(14)S.
12.2(18)SXF2	This command was integrated into Cisco IOS Release 12.2(18)SXF2.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
Cisco IOS XE Release 2.6	This command was integrated into Cisco IOS XE Release 2.6.

Examples

The following is sample output from the **debug ip rsvp wfq** command:

```
Router# debug ip rsvp wfq
RSVP debugging is on
Router# show debugging
IP RSVP debugging is on
IP RSVP debugging (Traffic Control events) is on
IP RSVP debugging (WFQ events) is on
Router#
03:03:23:RSVP-TC:Attempting to install QoS for rsb 6268A538
03:03:23:RSVP-TC:Adding new tcsb 00001A01 for rsb 6268A538
03:03:23:RSVP-TC:Assigning WFQ QoS to tcsb 00001A01
03:03:23:RSVP-TC:Consulting policy for tcsb 00001A01
03:03:23:RSVP-TC:Policy granted QoS for tcsb 00001A01
03:03:23:RSVP-TC:Requesting QoS for tcsb 00001A01
03:03:23:RSVP-TC:   ( r = 12500      bytes/s   M = 1514       bytes
03:03:23:RSVP-TC:     b = 1000       bytes     m = 0          bytes )
03:03:23:RSVP-TC:     p = 12500      bytes/s   Service Level = non-priority
03:03:23:RSVP-WFQ:Requesting a RESERVED queue on Et0/1 for tcsb 00001A01
03:03:23:RSVP-WFQ:Queue 265 allocated for tcsb 00001A01
03:03:23:RSVP-TC:Allocation succeeded for tcsb 00001A01
Router#
Router# no debug ip rsvp wfq
RSVP debugging is off
```

Related Commands

Command	Description
show debugging	Displays active debugging output.

debug ip rtp header-compression through debug ipv6 icmp

debug ip rtp header-compression through debug ipv6 icmp

debug ip rtp header-compression

To display events specific to Real-Time Transport Protocol (RTP) header compression, use the **debug ip rtp header-compression**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip rtp header-compression

no debug ip rtp header-compression

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Examples

The following is sample output from the **debug ip rtp header-compression**command:

```
Router# debug ip rtp header-compression
RHC BRI0: rcv compressed rtp packet
RHC BRI0: context0: expected sequence 0, received sequence 0
RHC BRI0: rcv compressed rtp packet
RHC BRI0: context0: expected sequence 1, received sequence 1
RHC BRI0: rcv compressed rtp packet
RHC BRI0: context0: expected sequence 2, received sequence 2
RHC BRI0: rcv compressed rtp packet
RHC BRI0: context0: expected sequence 3, received sequence 3
```

The table below describes the significant fields shown in the display.

Table 51: debug ip rtp header-compression Field Descriptions

Field	Description
context0	Compression state for a connection 0.
expected sequence	RTP header compression link sequence (expected).
received sequence	RTP header compression link sequence (actually received).

Related Commands

Command	Description
debug ip rtp packets	Displays a detailed dump of packets specific to RTP header compression.

debug ip rtp packets

To display a detailed dump of packets specific to Real-Time Transport Protocol (RTP) header compression, use the **debug ip rtp packets**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip rtp packets

no debug ip rtp packets

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Examples

The following is sample output from the **debug ip rtp packets**command:

```
Router# debug ip rtp packets
RTP packet dump:
  IP:  source: 171.68.8.10, destination: 224.2.197.169, id: 0x249B, ttl: 9,
       TOS: 0 prot: 17,
  UDP: source port: 1034, destination port: 27404, checksum: 0xB429,len: 152
  RTP: version: 2, padding: 0, extension: 0, marker: 0,
       payload: 3, ssrc 2369713968,
       sequence: 2468, timestamp: 85187180, csrc count: 0
```

The table below describes the significant fields shown in the display.

Table 52: debug ip rtp packets Field Descriptions

Field	Description
id	IP identification.
ttl	IP time to live (TTL).
len	Total UDP length.

Related Commands

Command	Description
debug ip rtp header-compression	Displays events specific to RTP header compression.

debug ip scp

To troubleshoot secure copy (SCP) authentication problems, use the **debug ip scp** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip scp

no debug ip scp

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(2)T	This command was introduced.
12.0(21)S	This command was integrated into Cisco IOS Release 12.0(21)S.
12.2(22)S	This command was integrated into Cisco IOS Release 12.2(22)S.
12.2(25)S	This command was integrated into Cisco IOS Release 12.2(25)S.
12.2(18)SXD	This command was integrated into Cisco IOS Release 12.2(18)SXD.

Examples

The following example is output from the **debug ip scp** command. In this example, a copy of the file scptest.cfg from a UNIX host running configuration of the router was successful.

```
Router# debug ip scp
4d06h:SCP:[22 -> 10.11.29.252:1018] send <OK>
4d06h:SCP:[22 <- 10.11.29.252:1018] recv C0644 20 scptest.cfg
4d06h:SCP:[22 -> 10.11.29.252:1018] send <OK>
4d06h:SCP:[22 <- 10.11.29.252:1018] recv 20 bytes
4d06h:SCP:[22 <- 10.11.29.252:1018] recv <OK>
4d06h:SCP:[22 -> 10.11.29.252:1018] send <OK>
4d06h:SCP:[22 <- 10.11.29.252:1018] recv <EOF>
```

The following example is also output from the **debug ip scp** command, but in this example, the user has privilege 0 and is therefore denied:

```
Router# debug ip scp
4d06h:SCP:[22 -> 10.11.29.252:1018] send Privilege denied.
```

Related Commands

Command	Description
ip scp server enable	Enables SCP server-side functionality.

debug ip sctp api

To provide diagnostic information about Stream Control Transmission Protocol (SCTP) application programming interfaces (APIs), use the **debug ip sctp api**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip sctp api

no debug ip sctp api

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(4)T	This command was introduced.

Usage Guidelines

In a live system, the debugging messages for performance, state, signal, and warnings are the most useful. These show any association or destination address failures and can be used to monitor the stability of any established associations.

Caution The **debug ip sctp api** command should not be used in a live system that has any significant amount of traffic running because it can generate a lot of traffic, which can cause associations to fail.

Examples

The following example shows SCTP calls to the API that are being executed and the parameters associated with these calls:

```
Router# debug ip sctp api
*Mar  1 00:31:14.211: SCTP: sctp_send: Assoc ID: 1
*Mar  1 00:31:14.211: SCTP:            stream num: 10
*Mar  1 00:31:14.211: SCTP:            bptr: 62EE332C, dptr: 4F7B598
*Mar  1 00:31:14.211: SCTP:            datalen: 100
*Mar  1 00:31:14.211: SCTP:            context: 1
*Mar  1 00:31:14.211: SCTP:            lifetime: 0
*Mar  1 00:31:14.211: SCTP:            unorder flag: FALSE
*Mar  1 00:31:14.211: SCTP:            bundle flag: TRUE
*Mar  1 00:31:14.211: SCTP: sctp_send successful return
*Mar  1 00:31:14.211: SCTP: sctp_receive: Assoc ID: 1
*Mar  1 00:31:14.215: SCTP:               max data len: 100
*Mar  1 00:31:14.215: SCTP: sctp_receive successful return
*Mar  1 00:31:14.215: SCTP: Process Send Request
*Mar  1 00:31:14.951: SCTP: sctp_receive: Assoc ID: 0
```

```
*Mar  1 00:31:14.951: SCTP:               max data len: 100
*Mar  1 00:31:14.951: SCTP: sctp_receive successful return
.
.
.
```

The table below describes the significant fields shown in the display.

Table 53: debug ip sctp api Field Descriptions

Field	Description
Assoc ID	Association identifier.
stream num	SCTP stream number.
bptr, dptr	Address of the buffer that contains the data, and address of the start of the data.
datalen	Length of the data that the application is sending (the datagram).
context	A value that is meaningful to the application. Returned with the datagram if the datagram ever needs to be retrieved.
lifetime	Not used.
unorder flag	Specifies that the datagram should be sent as unordered data.
bundle flag	Indicates whether the application wants the datagram to be delayed slightly, trying to bundle it with other data being sent.
max data len	Maximum length of data that can be received--the size of the receive buffer.

Related Commands

Command	Description
clear ip sctp statistics	Empties the buffer that holds SCTP statistics.
debug ip sctp congestion	Shows a list of all current SCTP associations.
show ip sctp association parameters	Shows the parameters configured for the association defined by the association identifier.
show ip sctp association statistics	Shows the current statistics for the association defined by the association identifier.
show ip sctp errors	Shows error counts logged by SCTP.

Command	Description
show ip sctp instances	Shows all currently defined SCTP instances.
show ip sctp statistics	Shows overall statistics counts for SCTP.
show iua as	Shows information about the current condition of an application server.
show iua asp	Shows information about the current condition of an application server process.

debug ip sctp congestion

To provide diagnostic information about Stream Control Transmission Protocol (SCTP) congestion parameters, use the **debug ip sctp congestion**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip sctp congestion

no debug ip sctp congestion

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(4)T	This command was introduced.

Usage Guidelines

In a live system, the debugging messages for performance, state, signal, and warnings are the most useful. These show any association or destination address failures and can be used to monitor the stability of any established associations.

Debug commands other than those for performance, state, signal, and warnings can generate a great deal of output and therefore can cause associations to fail. These commands should be used only in test environments or when there are very low amounts of traffic.

Examples

The following example shows parameters used to calculate SCTP congestion:

```
Router# debug ip sctp congestion
SCTP: Assoc 0: Slow start 10.6.0.4, cwnd 3000
SCTP: Assoc 0: Data chunks rcvd, local rwnd 7800
SCTP: Assoc 0: Free chunks, local rwnd 9000
SCTP: Assoc 0: Data chunks rcvd, local rwnd 8200
SCTP: Assoc 0: Add Sack, local a_rwnd 8200
SCTP: Assoc 0: Free chunks, local rwnd 9000
SCTP: Assoc 0: Data chunks rcvd, local rwnd 7800
SCTP: Assoc 0: Data chunks rcvd, local rwnd 7000
SCTP: Assoc 0: Add Sack, local a_rwnd 7000
SCTP: Assoc 0: Free chunks, local rwnd 9000
SCTP: Assoc 0: Bundle for 10.5.0.4, rem rwnd 14000, cwnd 19500, outstand 0
SCTP: Assoc 0: Bundled 12 chunks, remote rwnd 12800, outstand 1200
SCTP: Assoc 0: Bundling data, next chunk dataLen (100) > remaining mtu size
SCTP: Assoc 0: Bundle for 10.5.0.4, rem rwnd 12800, cwnd 19500, outstand 1200
SCTP: Assoc 0: Bundled 12 chunks, remote rwnd 11600, outstand 2400
SCTP: Assoc 0: Bundling data, next chunk dataLen (100) > remaining mtu size
SCTP: Assoc 0: Bundle for 10.5.0.4, rem rwnd 11600, cwnd 19500, outstand 2400
SCTP: Assoc 0: Bundled 12 chunks, remote rwnd 10400, outstand 3600
```

```
SCTP: Assoc 0: Bundling data, next chunk dataLen (100) > remaining mtu size
SCTP: Assoc 0: Bundle for 10.5.0.4, rem rwnd 10400, cwnd 19500, outstand 3600
SCTP: Assoc 0: Bundled 4 chunks, remote rwnd 10000, outstand 4000
SCTP: Assoc 0: No additional chunks waiting.
SCTP: Assoc 0: Data chunks rcvd, local rwnd 7800
SCTP: Assoc 0: Data chunks rcvd, local rwnd 7000
SCTP: Assoc 0: Add Sack, local a_rwnd 7000
SCTP: Assoc 0: Chunk A22F3B45 ack'd, dest 10.5.0.4, outstanding 3900
SCTP: Assoc 0: Chunk A22F3B46 ack'd, dest 10.5.0.4, outstanding 3800
SCTP: Assoc 0: Chunk A22F3B47 ack'd, dest 10.5.0.4, outstanding 3700
SCTP: Assoc 0: Chunk A22F3B48 ack'd, dest 10.5.0.4, outstanding 3600
SCTP: Assoc 0: Chunk A22F3B49 ack'd, dest 10.5.0.4, outstanding 3500
SCTP: Assoc 0: Chunk A22F3B4A ack'd, dest 10.5.0.4, outstanding 3400
SCTP: Assoc 0: Chunk A22F3B4B ack'd, dest 10.5.0.4, outstanding 3300
SCTP: Assoc 0: Chunk A22F3B4C ack'd, dest 10.5.0.4, outstanding 3200
SCTP: Assoc 0: Chunk A22F3B4D ack'd, dest 10.5.0.4, outstanding 3100
SCTP: Assoc 0: Chunk A22F3B4E ack'd, dest 10.5.0.4, outstanding 3000
SCTP: Assoc 0: Chunk A22F3B4F ack'd, dest 10.5.0.4, outstanding 2900
SCTP: Assoc 0: Chunk A22F3B50 ack'd, dest 10.5.0.4, outstanding 2800
SCTP: Assoc 0: Chunk A22F3B51 ack'd, dest 10.5.0.4, outstanding 2700
SCTP: Assoc 0: Chunk A22F3B52 ack'd, dest 10.5.0.4, outstanding 2600
SCTP: Assoc 0: Chunk A22F3B53 ack'd, dest 10.5.0.4, outstanding 2500
SCTP: Assoc 0: Chunk A22F3B54 ack'd, dest 10.5.0.4, outstanding 2400
SCTP: Assoc 0: Chunk A22F3B55 ack'd, dest 10.5.0.4, outstanding 2300
SCTP: Assoc 0: Chunk A22F3B56 ack'd, dest 10.5.0.4, outstanding 2200
```

The table below describes the significant fields shown in the display.

Table 54: debug ip sctp congestion Field Descriptions

Field	Description
cwnd	Congestion window values for destination address.
rwnd, a_rwnd	Receiver window values as defined in RFC 2960.
outstanding	Number of bytes outstanding.

Related Commands

Command	Description
clear ip sctp statistics	Empties the buffer that holds SCTP statistics.
show ip sctp association parameters	Shows the parameters configured for the association defined by the association identifier.
show ip sctp association statistics	Shows the current statistics for the association defined by the association identifier.
show ip sctp errors	Shows error counts logged by SCTP.
show ip sctp instances	Shows all currently defined SCTP instances.
show ip sctp statistics	Shows overall statistics counts for SCTP.
show iua as	Shows information about the current condition of an application server.

Command	Description
show iua asp	Shows information about the current condition of an application server process.

debug ip sctp init

To show datagrams and other information related to the initializing of new Stream Control Transmission Protocol (SCTP) associations, use the **debug ip sctp init**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip sctp init

no debug ip sctp init

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(4)T	This command was introduced.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.

Usage Guidelines

All initialization chunks are shown, including the INIT, INIT_ACK, COOKIE_ECHO, and COOKIE_ACK chunks. This debug command can be used to see the chunks associated with any initialization sequence but does not display data chunks sent once the association is established. Therefore, it is safe to use in a live system that has traffic flowing when you have trouble with associations failing and being reestablished.

Examples

The following example shows initialization chunks for SCTP associations:

```
Router# debug ip sctp init
*Mar  1 00:53:07.279: SCTP Test: Attempting to open assoc to remote port 8787...assoc ID
is 0
*Mar  1 00:53:07.279: SCTP: Process Assoc Request
*Mar  1 00:53:07.279: SCTP: Assoc 0: dest addr list:
*Mar  1 00:53:07.279: SCTP:                 addr 10.5.0.4
*Mar  1 00:53:07.279: SCTP:                 addr 10.6.0.4
*Mar  1 00:53:07.279:
...
*Mar  1 00:53:13.279: SCTP: Assoc 0: Send Init
*Mar  1 00:53:13.279: SCTP:        INIT_CHUNK, len 42
*Mar  1 00:53:13.279: SCTP:        Initiate Tag: B4A10C4D, Initial TSN: B4A10C4D, rwnd 9000
*Mar  1 00:53:13.279: SCTP:        Streams Inbound: 13, Outbound: 13
*Mar  1 00:53:13.279: SCTP:        IP Addr: 10.1.0.2
*Mar  1 00:53:13.279: SCTP:        IP Addr: 10.2.0.2
*Mar  1 00:53:13.279: SCTP:        Supported addr types: 5
*Mar  1 00:53:13.307: SCTP: Process Init
```

```
*Mar  1 00:53:13.307: SCTP:        INIT_CHUNK, len 42
*Mar  1 00:53:13.307: SCTP:        Initiate Tag: 3C2D8327, Initial TSN: 3C2D8327, rwnd 18000
*Mar  1 00:53:13.307: SCTP:        Streams Inbound: 13, Outbound: 13
*Mar  1 00:53:13.307: SCTP:        IP Addr: 10.5.0.4
*Mar  1 00:53:13.307: SCTP:        IP Addr: 10.6.0.4
*Mar  1 00:53:13.307: SCTP:        Supported addr types: 5
*Mar  1 00:53:13.307: SCTP: Assoc 0: Send InitAck
*Mar  1 00:53:13.307: SCTP:        INIT_ACK_CHUNK, len 124
*Mar  1 00:53:13.307: SCTP:        Initiate Tag: B4A10C4D, Initial TSN: B4A10C4D, rwnd 9000
*Mar  1 00:53:13.307: SCTP:        Streams Inbound: 13, Outbound: 13
*Mar  1 00:53:13.307: SCTP:        Responder cookie len 88
*Mar  1 00:53:13.307: SCTP:        IP Addr: 10.1.0.2
*Mar  1 00:53:13.307: SCTP:        IP Addr: 10.2.0.2
*Mar  1 00:53:13.311: SCTP: Assoc 0: Process Cookie
*Mar  1 00:53:13.311: SCTP:        COOKIE_ECHO_CHUNK, len 88
*Mar  1 00:53:13.311: SCTP: Assoc 0: dest addr list:
*Mar  1 00:53:13.311: SCTP:                 addr 10.5.0.4
*Mar  1 00:53:13.311: SCTP:                 addr 10.6.0.4
*Mar  1 00:53:13.311:
*Mar  1 00:53:13.311: SCTP: Instance 0 dest addr list:
*Mar  1 00:53:13.311: SCTP:                 addr 10.5.0.4
*Mar  1 00:53:13.311: SCTP:                 addr 10.6.0.4
*Mar  1 00:53:13.311:
*Mar  1 00:53:13.311: SCTP: Assoc 0: Send CookieAck
*Mar  1 00:53:13.311: SCTP:        COOKIE_ACK_CHUNK
```

The table below describes the significant fields shown in the display.

Table 55: debug ip sctp init Field Descriptions

Field	Description
Initiate Tag	Initiation chunk identifier.
Initial TSN	Initial transmission sequence number.
rwnd	Receiver window values.

Related Commands

Command	Description
clear ip sctp statistics	Empties the buffer that holds SCTP statistics.
debug ip sctp congestion	Shows a list of all current SCTP associations.
show ip sctp association parameters	Shows the parameters configured for the association defined by the association identifier.
show ip sctp association statistics	Shows the current statistics for the association defined by the association identifier.
show ip sctp errors	Shows error counts logged by SCTP.
show ip sctp instances	Shows all currently defined SCTP instances.
show ip sctp statistics	Shows overall statistics counts for SCTP.

Command	Description
show iua as	Shows information about the current condition of an application server.
show iua asp	Shows information about the current condition of an application server process.

debug ip sctp multihome

To show the source and destination of datagrams in order to monitor the use of the multihome addresses for Stream Control Transmission Protocol (SCTP), use the **debug ip sctp multihome**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip sctp multihome

no debug ip sctp multihome

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(4)T	This command was introduced.

Usage Guidelines

More than one IP address parameter can be included in an initialization (INIT) chunk when the INIT sender is multihomed. Datagrams should be sent to the primary destination addresses unless the network is experiencing problems, in which case the datagrams should be sent to secondary addresses.

Caution The **debug ip sctp multihome** command generates one debug line for each datagram sent or received. It should be used with extreme caution in a live network.

Examples

The following example shows source and destination for multihomed addresses:

```
Router# debug ip sctp multihome
SCTP: Rcvd s=10.5.0.4  8787, d=10.1.0.2  8787, len 1404
SCTP: Rcvd s=10.5.0.4  8787, d=10.1.0.2  8787, len 476
SCTP: Sent:  Assoc 0: s=10.1.0.2  8787, d=10.5.0.4  8787, len 28
SCTP: Assoc 0: Send Data to dest 10.5.0.4
SCTP: Sent:  Assoc 0: s=10.1.0.2  8787, d=10.5.0.4  8787, len 1404
SCTP: Sent:  Assoc 0: s=10.1.0.2  8787, d=10.5.0.4  8787, len 1404
SCTP: Sent:  Assoc 0: s=10.1.0.2  8787, d=10.5.0.4  8787, len 1404
SCTP: Sent:  Assoc 0: s=10.1.0.2  8787, d=10.5.0.4  8787, len 476
SCTP: Rcvd s=10.5.0.4  8787, d=10.1.0.2  8787, len 28
SCTP: Rcvd s=10.5.0.4  8787, d=10.1.0.2  8787, len 28
SCTP: Rcvd s=10.5.0.4  8787, d=10.1.0.2  8787, len 1404
SCTP: Rcvd s=10.5.0.4  8787, d=10.1.0.2  8787, len 1404
SCTP: Sent:  Assoc 0: s=10.1.0.2  8787, d=10.5.0.4  8787, len 28
SCTP: Rcvd s=10.5.0.4  8787, d=10.1.0.2  8787, len 1404
SCTP: Rcvd s=10.5.0.4  8787, d=10.1.0.2  8787, len 476
SCTP: Sent:  Assoc 0: s=10.1.0.2  8787, d=10.5.0.4  8787, len 28
```

```
SCTP: Assoc 0: Send Data to dest 10.5.0.4
SCTP: Sent:  Assoc 0: s=10.1.0.2  8787, d=10.5.0.4  8787, len 1404
SCTP: Sent:  Assoc 0: s=10.1.0.2  8787, d=10.5.0.4  8787, len 1404
SCTP: Sent:  Assoc 0: s=10.1.0.2  8787, d=10.5.0.4  8787, len 1404
SCTP: Sent:  Assoc 0: s=10.1.0.2  8787, d=10.5.0.4  8787, len 476
SCTP: Rcvd s=10.6.0.4  8787, d=10.2.0.2  8787, len 44
SCTP: Sent:  Assoc 0: s=10.2.0.2  8787, d=10.6.0.4  8787, len 44
SCTP: Rcvd s=10.5.0.4  8787, d=10.1.0.2  8787, len 28
SCTP: Rcvd s=10.5.0.4  8787, d=10.1.0.2  8787, len 28
SCTP: Rcvd s=10.5.0.4  8787, d=10.1.0.2  8787, len 1404
SCTP: Rcvd s=10.5.0.4  8787, d=10.1.0.2  8787, len 1404
SCTP: Sent:  Assoc 0: s=10.1.0.2  8787, d=10.5.0.4  8787, len 28
SCTP: Rcvd s=10.5.0.4  8787, d=10.1.0.2  8787, len 1404
SCTP: Rcvd s=10.5.0.4  8787, d=10.1.0.2  8787, len 476
```

The table below describes the significant fields shown in the display.

Table 56: debug ip sctp multihome Field Descriptions

Field	Description
s	Source address and port.
d	Destination address and port.

Related Commands

Command	Description
clear ip sctp statistics	Empties the buffer that holds SCTP statistics.
debug ip sctp congestion	Shows a list of all current SCTP associations.
show ip sctp association parameters	Shows the parameters configured for the association defined by the association identifier.
show ip sctp association statistics	Shows the current statistics for the association defined by the association identifier.
show ip sctp errors	Shows error counts logged by SCTP.
show ip sctp instances	Shows all currently defined SCTP instances.
show ip sctp statistics	Shows overall statistics counts for SCTP.
show iua as	Shows information about the current condition of an application server.
show iua asp	Shows information about the current condition of an application server process.

debug ip sctp performance

To display the average number of Stream Control Transmission Protocol (SCTP) chunks and datagrams being sent and received per second, use the **debug ip sctp performance**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip sctp performance

no debug ip sctp performance

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(4)T	This command was introduced.

Usage Guidelines

In a live system, the debugging messages for performance, state, signal, and warnings are the most useful. These show any association or destination address failures and can be used to monitor the stability of any established associations.

Once enabled, the **debug ip sctp performance** command displays the average number of chunks and datagrams being sent and received per second once every 10 seconds. Note that the averages are cumulative since the last time the statistics were cleared using the **clear ip sctp statistics** command and may not accurately reflect the number of datagrams and chunks currently being sent and received at that particular moment.

Examples

The following example shows a low rate of traffic:

```
Router# debug ip sctp performance

SCTP Sent: SCTP Dgrams 5, Chunks 28, Data Chunks 29, ULP Dgrams 29
SCTP Rcvd: SCTP Dgrams 7, Chunks 28, Data Chunks 29, ULP Dgrams 29
Chunks Discarded: 0,  Retransmitted 0
SCTP Sent: SCTP Dgrams 6, Chunks 29, Data Chunks 30, ULP Dgrams 30
SCTP Rcvd: SCTP Dgrams 7, Chunks 29, Data Chunks 30, ULP Dgrams 30
Chunks Discarded: 0,  Retransmitted 0
```

The table below describes the significant fields shown in the display.

Table 57: debug ip sctp performance Field Descriptions

Field	Description
SCTP Dgrams	Datagram sent to or received from the network.

Field	Description
Chunks	Includes data chunks and control chunks sent or received.
Data Chunks	Data chunks sent or received.
ULP Dgrams	Upper-layer protocol (ULP) datagrams, which are datagrams sent to or received from the ULP or application.

Related Commands

Command	Description
clear ip sctp statistics	Empties the buffer that holds SCTP statistics.
debug ip sctp congestion	Shows a list of all current SCTP associations.
show ip sctp association parameters	Shows the parameters configured for the association defined by the association identifier.
show ip sctp association statistics	Shows the current statistics for the association defined by the association identifier.
show ip sctp errors	Shows error counts logged by SCTP.
show ip sctp instances	Shows all currently defined SCTP instances.
show ip sctp statistics	Shows overall statistics counts for SCTP.
show iua as	Shows information about the current condition of an application server.
show iua asp	Shows information about the current condition of an application server process.

debug ip sctp rcvchunks

To provide diagnostic information about chunks received with Stream Control Transmission Protocol (SCTP), use the **debug ip sctp rcvchunks**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip sctp rcvchunks

no debug ip sctp rcvchunks

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(4)T	This command was introduced.

Usage Guidelines

The **debug ip sctp rcvchunks** command shows the following information about received chunks:

- Whether the chunk is for a new datagram or is part of a datagram that is being reassembled
- Whether the datagram is complete after receiving this chunk
- If the datagram is complete, whether the datagram is in sequence within the specified stream and can be delivered to the upper-layer protocol (ULP)
- The selective acknowledgments (SACKs) that are returned to the remote SCTP peer
- The cumulative transmission sequence number (Cum TSN) that was acknowledged and the number of fragments included
- Whether the datagram is received by the ULP

Caution The **debug ip sctp rcvchunks** command generates multiple debug lines for each chunk received. It should be used with extreme caution in a live network.

Examples

In the following example, a segmented datagram is received in two chunks for stream 0 and sequence number 0. The length of the first chunk is 1452 bytes, and the second is 1 byte. The first chunk indicates that it is for a new datagram, but the second chunk indicates that it is part of an existing datagram that is already being reassembled. When the first chunk is processed, it is noted to be in sequence, but is not complete and so cannot be delivered yet. When the second chunk is received, the datagram is both in sequence and complete. The application receives the datagram, and a SACK is shown to acknowledge that both chunks were received with no missing chunks indicated (that is, with no fragments).

```
Router# debug ip sctp rcvchunks
SCTP: Assoc 0: New chunk (0/0/1452/2C33D822) for new dgram (0)
SCTP: Assoc 0: dgram (0) is in seq
SCTP: Assoc 0: Add Sack Chunk, CumTSN=2C33D822, numFrags=0
SCTP: Assoc 0: New chunk (0/0/1/2C33D823) for existing dgram (0)
SCTP: Assoc 0: dgram (0) is complete
SCTP: Assoc 0: ApplRecv chunk 0/0/1452/2C33D822
SCTP: Assoc 0: ApplRecv chunk 0/0/1/2C33D823
SCTP: Assoc 0: Add Sack Chunk, CumTSN=2C33D823, numFrags=0
```

The table below describes the significant fields shown in the display.

Table 58: debug ip sctp rcvchunks Field Descriptions

Field	Description
0 / 0 / 1452 / 2C33D822	Stream number / datagram sequence number / chunk length, in bytes / chunk transmission sequence number.
Sack Chunk	Selective acknowledgment chunk.
ApplRecv	Application has received the chunk.
CumTSN	Cumulative transmission sequence number that is being acknowledged.
numFrags	Number of fragments, or missing chunks.

Related Commands

Command	Description
clear ip sctp statistics	Empties the buffer that holds SCTP statistics.
debug ip sctp congestion	Shows a list of all current SCTP associations.
show ip sctp association parameters	Shows the parameters configured for the association defined by the association identifier.
show ip sctp association statistics	Shows the current statistics for the association defined by the association identifier.
show ip sctp errors	Shows error counts logged by SCTP.
show ip sctp instances	Shows all currently defined SCTP instances.
show ip sctp statistics	Shows overall statistics counts for SCTP.

Command	Description
show iua as	Shows information about the current condition of an application server.
show iua asp	Shows information about the current condition of an application server process.

debug ip sctp rto

To show adjustments that are made to the retransmission timeout (RTO) value when using Stream Control Transmission Protocol (SCTP), use the **debug ip sctp rto**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip sctp rto

no debug ip sctp rto

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(4)T	This command was introduced.

Usage Guidelines

The **debug ip sctp rto** command shows adjustments that are made to the retransmission timeout value (shown as retrans in the command output) because of either retransmission of data chunks or unacknowledged heartbeats.

Caution The **debug ip sctp rto** command can generate a great deal of output. It should be used with extreme caution in a live network.

Examples

In the following example, there is only one destination address available. Each time the chunk needs to be retransmitted, the RTO value is doubled.

```
Router# debug ip sctp rto
SCTP: Assoc 0: destaddr 10.5.0.4, retrans timeout on chunk 942BAC55
SCTP: Assoc 0: destaddr 10.5.0.4, rto backoff 2000 ms
SCTP: Assoc 0: destaddr 10.5.0.4, retrans timeout on chunk 942BAC55
SCTP: Assoc 0: destaddr 10.5.0.4, rto backoff 4000 ms
SCTP: Assoc 0: destaddr 10.5.0.4, retrans timeout on chunk 942BAC55
SCTP: Assoc 0: destaddr 10.5.0.4, rto backoff 8000 ms
SCTP: Assoc 0: destaddr 10.5.0.4, retrans timeout on chunk 942BAC55
SCTP: Assoc 0: destaddr 10.5.0.4, rto backoff 16000 ms
SCTP: Assoc 0: destaddr 10.5.0.4, retrans timeout on chunk 942BAC55
SCTP: Assoc 0: destaddr 10.5.0.4, rto backoff 32000 ms
```

Related Commands

Command	Description
clear ip sctp statistics	Empties the buffer that holds SCTP statistics.
debug ip sctp congestion	Shows a list of all current SCTP associations.
show ip sctp association parameters	Shows the parameters configured for the association defined by the association identifier.
show ip sctp association statistics	Shows the current statistics for the association defined by the association identifier.
show ip sctp errors	Shows error counts logged by SCTP.
show ip sctp instances	Shows all currently defined SCTP instances.
show ip sctp statistics	Shows overall statistics counts for SCTP.
show iua as	Shows information about the current condition of an application server.
show iua asp	Shows information about the current condition of an application server process.

debug ip sctp segments

To show short diagnostics for every datagram that is sent or received with Stream Control Transmission Protocol (SCTP), use the **debug ip sctp segments**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip sctp segments

no debug ip sctp segments

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(4)T	This command was introduced.

Usage Guidelines

The **debug ip sctp segments** command provides the short form of the output about datagrams. For the verbose form, use the **debug ip sctp segmentv** command.

Caution The **debug ip sctp segments** command generates several lines of output for each datagram sent or received. It should be used with extreme caution in a live network.

Examples

The following output shows an example in which an association is established, a few heartbeats are sent, the remote endpoint fails, and the association is restarted.

```
Router# debug ip sctp segments
SCTP: Sent:  Assoc 0: s=10.1.0.2  8787, d=10.5.0.4  8787, len 56
SCTP:        INIT_CHUNK, Tag: 3C72A02A, TSN: 3C72A02A
SCTP: Recv:  Assoc 0: s=10.5.0.4  8787, d=10.1.0.2  8787, len 56
SCTP:        INIT_CHUNK, Tag: 13E5AD6C, TSN: 13E5AD6C
SCTP: Sent:  Assoc NULL: s=10.1.0.2  8787, d=10.5.0.4  8787, len 136
SCTP:        INIT_ACK_CHUNK, Tag: 3C72A02A, TSN: 3C72A02A
SCTP: Recv:  Assoc 0: s=10.5.0.4  8787, d=10.1.0.2  8787, len 100
SCTP:        COOKIE_ECHO_CHUNK, len 88
SCTP: Sent:  Assoc NULL: s=10.1.0.2  8787, d=10.5.0.4  8787, len 16
SCTP:        COOKIE_ACK_CHUNK
SCTP: Sent:  Assoc 0: s=10.1.0.2  8787, d=10.5.0.4  8787, len 52
SCTP:        HEARTBEAT_CHUNK
SCTP: Sent:  Assoc 0: s=10.1.0.2  8787, d=10.5.0.4  8787, len 52
SCTP:        HEARTBEAT_CHUNK
SCTP: Sent:  Assoc 0: s=10.1.0.2  8787, d=10.5.0.4  8787, len 52
SCTP:        HEARTBEAT_CHUNK
```

```
SCTP: Recv:  Assoc 0: s=10.5.0.4  8787, d=10.1.0.2  8787, len 56
SCTP:        INIT_CHUNK, Tag: 4F2D8235, TSN: 4F2D8235
SCTP: Sent:  Assoc NULL: s=10.1.0.2  8787, d=10.5.0.4  8787, len 136
SCTP:        INIT_ACK_CHUNK, Tag: 7DD7E424, TSN: 7DD7E424
SCTP: Recv:  Assoc 0: s=10.5.0.4  8787, d=10.1.0.2  8787, len 100
SCTP:        COOKIE_ECHO_CHUNK, len 88
SCTP: Sent:  Assoc NULL: s=10.1.0.2  8787, d=10.5.0.4  8787, len 16
SCTP:        COOKIE_ACK_CHUNK
SCTP: Recv:  Assoc 0: s=10.5.0.4  8787, d=10.1.0.2  8787, len 144
SCTP:        SACK_CHUNK, TSN ack: 7DD7E423, rwnd 18000, num frags 0
SCTP:        DATA_CHUNK, 4/0/100/4F2D8235
SCTP: Sent:  Assoc 0: s=10.1.0.2  8787, d=10.5.0.4  8787, len 28
SCTP:        SACK_CHUNK, TSN ack: 4F2D8235, rwnd 8900, num frags 0
SCTP: Sent:  Assoc 0: s=10.1.0.2  8787, d=10.5.0.4  8787, len 128
SCTP:        DATA_CHUNK, 4/0/100/7DD7E424
SCTP: Recv:  Assoc 0: s=10.5.0.4  8787, d=10.1.0.2  8787, len 28
SCTP:        SACK_CHUNK, TSN ack: 7DD7E424, rwnd 17900, num frags 0
SCTP: Recv:  Assoc 0: s=10.6.0.4  8787, d=10.2.0.2  8787, len 44
SCTP:        HEARTBEAT_CHUNK
SCTP: Sent:  Assoc 0: s=10.2.0.2  8787, d=10.6.0.4  8787, len 44
SCTP:        HEARTBEAT_ACK_CHUNK
SCTP: Recv:  Assoc 0: s=10.5.0.4  8787, d=10.1.0.2  8787, len 128
SCTP:        DATA_CHUNK, 7/0/100/4F2D8236
SCTP: Sent:  Assoc 0: s=10.1.0.2  8787, d=10.5.0.4  8787, len 144
SCTP:        SACK_CHUNK, TSN ack: 4F2D8236, rwnd 9000, num frags 0
SCTP:        DATA_CHUNK, 7/0/100/7DD7E425
SCTP: Recv:  Assoc 0: s=10.5.0.4  8787, d=10.1.0.2  8787, len 28
SCTP:        SACK_CHUNK, TSN ack: 7DD7E424, rwnd 18000, num frags 0
SCTP: Recv:  Assoc 0: s=10.5.0.4  8787, d=10.1.0.2  8787, len 28
SCTP:        SACK_CHUNK, TSN ack: 7DD7E425, rwnd 17900, num frags 0
SCTP: Recv:  Assoc 0: s=10.5.0.4  8787, d=10.1.0.2  8787, len 128
SCTP:        DATA_CHUNK, 4/1/100/4F2D8237
```

The table below describes the significant fields shown in the display.

Table 59: debug ip sctp segments Field Descriptions

Field	Description
s	Source address and port.
d	Destination address and port.
len	Length of chunk, in bytes.
Tag	The identifier for an initialization chunk.
TSN	Transmission sequence number.
rwnd	Receiver window value.
num frags	Number of fragments received.
7 / 0 / 100 / 4F2D8236	(Data chunks) Stream number / datagram sequence number / chunk length, in bytes / chunk transmission sequence number.

Related Commands

Command	Description
clear ip sctp statistics	Empties the buffer that holds SCTP statistics.
debug ip sctp congestion	Shows a list of all current SCTP associations.
debug ip sctp segmentv	Shows every datagram that is sent or received and the chunks that are contained in each. This is the verbose form of the output, and it shows detailed information for each chunk type.
show ip sctp association parameters	Shows the parameters configured for the association defined by the association identifier.
show ip sctp association statistics	Shows the current statistics for the association defined by the association identifier.
show ip sctp errors	Shows error counts logged by SCTP.
show ip sctp instances	Shows all currently defined SCTP instances.
show ip sctp statistics	Shows overall statistics counts for SCTP.
show iua as	Shows information about the current condition of an application server.
show iua asp	Shows information about the current condition of an application server process.

debug ip sctp segmentv

To show verbose diagnostics for every datagram that is sent or received with Stream Control Transmission Protocol (SCTP), use the **debug ip sctp segmentv**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip sctp segmentv

no debug ip sctp segmentv

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(4)T	This command was introduced.

Usage Guidelines

The **debug ip sctp segmentv** command provides the verbose form of the output for datagrams. For the simple form, use the **debug ip sctp segments** command.

Caution The **debug ip sctp segmentv** command generates multiple lines of output for each datagram sent and received. It should be used with extreme caution in a live network.

Examples

The following output shows an example in which an association is established, a few heartbeats are sent, the remote endpoint fails, and the association is restarted:

```
Router# debug ip sctp segmentv
SCTP: Sent:  Assoc 0: s=10.1.0.2  8787, d=10.5.0.4  8787, len 56, ver tag 0
SCTP:        INIT_CHUNK, len 42
SCTP:        Initiate Tag: B131ED6A, Initial TSN: B131ED6A, rwnd 9000
SCTP:        Streams Inbound: 13, Outbound: 13
SCTP:        IP Addr: 10.1.0.2
SCTP:        IP Addr: 10.2.0.2
SCTP:        Supported addr types: 5
SCTP: Recv:  Assoc 0: s=10.5.0.4  8787, d=10.1.0.2  8787, len 56, ver tag 0
SCTP:        INIT_CHUNK, len 42
SCTP:        Initiate Tag: 5516B2F3, Initial TSN: 5516B2F3, rwnd 18000
SCTP:        Streams Inbound: 13, Outbound: 13
SCTP:        IP Addr: 10.5.0.4
SCTP:        IP Addr: 10.6.0.4
SCTP:        Supported addr types: 5
SCTP: Sent:  Assoc NULL: s=10.1.0.2  8787, d=10.5.0.4  8787, len 136, ver tag 5516B2F3
SCTP:        INIT_ACK_CHUNK, len 124
```

```
SCTP:        Initiate Tag: B131ED6A, Initial TSN: B131ED6A, rwnd 9000
SCTP:        Streams Inbound: 13, Outbound: 13
SCTP:        Responder cookie len 88
SCTP:        IP Addr: 10.1.0.2
SCTP:        IP Addr: 10.2.0.2
SCTP: Recv:  Assoc 0: s=10.5.0.4  8787, d=10.1.0.2  8787, len 100, ver tag B131ED6A
SCTP:        COOKIE_ECHO_CHUNK, len 88
SCTP: Sent:  Assoc NULL: s=10.1.0.2  8787, d=10.5.0.4  8787, len 16, ver tag 5516B2F3
SCTP:        COOKIE_ACK_CHUNK
SCTP: Recv:  Assoc 0: s=10.5.0.4  8787, d=10.1.0.2  8787, len 144, ver tag B131ED6A
SCTP:        SACK_CHUNK, len 16
SCTP:        TSN ack: (0xB131ED69)
SCTP:        Rcv win credit: 18000
SCTP:        Num frags: 0
SCTP:        DATA_CHUNK, flags 3, chunkLen 116
SCTP:        DATA_CHUNK, 0/0/100/5516B2F3
SCTP: Sent:  Assoc 0: s=10.1.0.2  8787, d=10.5.0.4  8787, len 28, ver tag 5516B2F3
SCTP:        SACK_CHUNK, len 16
SCTP:        TSN ack: (0x5516B2F3)
SCTP:        Rcv win credit: 8900
SCTP:        Num frags: 0
SCTP: Sent:  Assoc 0: s=10.1.0.2  8787, d=10.5.0.4  8787, len 128, ver tag 5516B2F3
SCTP:        DATA_CHUNK, flags 3, chunkLen 116
SCTP:        DATA_CHUNK, 0/0/100/B131ED6A
SCTP: Recv:  Assoc 0: s=10.6.0.4  8787, d=10.2.0.2  8787, len 44, ver tag B131ED6A
SCTP:        HEARTBEAT_CHUNK
SCTP: Sent:  Assoc 0: s=10.2.0.2  8787, d=10.6.0.4  8787, len 44, ver tag 5516B2F3
SCTP:        HEARTBEAT_ACK_CHUNK
SCTP: Recv:  Assoc 0: s=10.5.0.4  8787, d=10.1.0.2  8787, len 28, ver tag B131ED6A
SCTP:        SACK_CHUNK, len 16
```

The table below describes the significant fields shown in the display.

Table 60: debug ip sctp segmentv Field Descriptions

Field	Description
s	Source address and port.
d	Destination address and port.
len	Length of chunk, in bytes.
ver tag	Verification identifier.
Tag	The identifier for an initialization chunk.
TSN	Transmission sequence number.
rwnd	Receive window value.
Rcv win credit	Receive window value. Same as rwnd.
Num frags	Number of fragments received.
0/0/100/5516B2F3	(Data chunks) Stream number / datagram sequence number / chunk length, in bytes / chunk transmission sequence number.

Related Commands

Command	Description
clear ip sctp statistics	Empties the buffer that holds SCTP statistics.
debug ip sctp congestion	Shows a list of all current SCTP associations.
debug ip sctp segments	Shows short diagnostics for every datagram that is sent or received with SCTP.
show ip sctp association parameters	Shows the parameters configured for the association defined by the association identifier.
show ip sctp association statistics	Shows the current statistics for the association defined by the association identifier.
show ip sctp errors	Shows error counts logged by SCTP.
show ip sctp instances	Shows all currently defined SCTP instances.
show ip sctp statistics	Shows overall statistics counts for SCTP.
show iua as	Shows information about the current condition of an application server.
show iua asp	Shows information about the current condition of an application server process.

debug ip sctp signal

To show signals that are sent from Stream Control Transmission Protocol (SCTP) to the application or upper-layer protocol (ULP), use the **debug ip sctp signal**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip sctp signal

no debug ip sctp signal

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(4)T	This command was introduced.

Usage Guidelines

The **debug ip sctp signal** command can be used to see if the current associations are stable or not. Because it generates output only on state transitions, it is safe to use in a live environment. It still should be used with caution, however, depending on the number of associations being handled by the system and the stability of the network.

The **debug ip sctp state** command is often used at the same time as the **debug ip sctp signal** command. Using the two commands together gives good insight into the stability of associations.

Examples

In the following example, a new association is requested and established. The peer then restarts the association and notes that the association failed and is being reestablished. The local peer then indicates that the association has failed because it has tried to retransmit the specified chunk more than the maximum number of times without success. As a result, the association fails (because of communication loss) and is terminated. The ULP requests that the association be attempted again, and this attempt succeeds. A shutdown is then received from the remote peer, and the local peer enters the shutdown acknowledge sent state, which is followed by the association being terminated. Again, another association attempt is made and succeeds.

```
Router# debug ip sctp signal
Router# debug ip sctp state
<new assoc attempt>
00:20:08: SCTP: Assoc 0:  state CLOSED -> COOKIE_WAIT
00:20:15: SCTP: Assoc 0:  state COOKIE_WAIT -> ESTABLISHED
00:20:15: SCTP: Assoc 0: Sent ASSOC_UP signal for CONFIGD_ASSOC
00:21:03: SCTP: Assoc 0: Restart rcvd from peer
00:21:03: SCTP: Assoc 0: Sent ASSOC_RESTART signal
00:21:04: SCTP: Assoc 0: chunk 62EA7F40 retransmitted more than max times, failing assoc
00:21:04: SCTP: Assoc 0: Sent ASSOC_FAILED signal, reason: SCTP_COMM_LOST
00:21:04: SCTP: Assoc 0: Sent ASSOC_TERMINATE signal
```

```
00:21:04: SCTP: Assoc 0:  state ESTABLISHED -> CLOSED
<new assoc attempt>
00:21:04: SCTP: Assoc 0:  state CLOSED -> COOKIE_WAIT
00:21:04: SCTP: Assoc 0:  state COOKIE_WAIT -> COOKIE_ECHOED
00:21:04: SCTP: Assoc 0:  state COOKIE_ECHOED -> ESTABLISHED
00:21:04: SCTP: Assoc 0: Sent ASSOC_UP signal for CONFIGD_ASSOC
00:21:04: SCTP: Assoc 0: Sent TERMINATE_PENDING signal
00:21:04: SCTP: Assoc 0:  state ESTABLISHED -> SHUTDOWN_ACKSENT
00:21:04: SCTP: Assoc 0: Sent ASSOC_TERMINATE signal
00:21:04: SCTP: Assoc 0:  state SHUTDOWN_ACKSENT -> CLOSED
<new assoc attempt>
00:21:04: SCTP: Assoc 0:  state CLOSED -> COOKIE_WAIT
00:21:04: SCTP: Assoc 0:  state COOKIE_WAIT -> COOKIE_ECHOED
00:21:04: SCTP: Assoc 0:  state COOKIE_ECHOED -> ESTABLISHED
00:21:04: SCTP: Assoc 0: Sent ASSOC_UP signal for CONFIGD_ASSOC
```

Related Commands

Command	Description
clear ip sctp statistics	Empties the buffer that holds SCTP statistics.
debug ip sctp congestion	Shows a list of all current SCTP associations.
debug ip sctp state	Shows SCTP state transitions.
show ip sctp association parameters	Shows the parameters configured for the association defined by the association identifier.
show ip sctp association statistics	Shows the current statistics for the association defined by the association identifier.
show ip sctp errors	Shows error counts logged by SCTP.
show ip sctp instances	Shows all currently defined SCTP instances.
show ip sctp statistics	Shows overall statistics counts for SCTP.
show iua as	Shows information about the current condition of an application server.
show iua asp	Shows information about the current condition of an application server process.

debug ip sctp sndchunks

To show information about chunks that are being sent to remote Stream Control Transmission Protocol (SCTP) peers, use the **debug ip sctp sndchunks**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip sctp sndchunks

no debug ip sctp sndchunks

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(4)T	This command was introduced.

Usage Guidelines

The **debug ip sctp sndchunks** command provides the following information:

- Application send requests from the local SCTP peer
- Chunks being bundled and sent to the remote peer
- Processing of the selective acknowledgments (SACKs) from the remote peer, indicating which chunks were successfully received
- Chunks that are marked for retransmission

Caution The **debug ip sctp sndchunks** command generates large amounts of data if there is any significant amount of traffic flowing. It should be used with extreme caution in live networks.

Examples

The following example shows output for the **debug ip sctp sndchunks** command for a case in which data chunks are being sent, with some of them marked for retransmission:

```
Router# debug ip sctp sndchunks
SCTP: Assoc 0: ApplSend, chunk: 0/10412/100/A23134F8 to 10.5.0.4
SCTP: Assoc 0: ApplSend, chunk: 5/10443/100/A23134F9 to 10.5.0.4
SCTP: Assoc 0: ApplSend, chunk: 5/10448/100/A231355C to 10.5.0.4
SCTP: Assoc 0: Set oldest chunk for dest 10.5.0.4 to TSN A23134F8
SCTP: Assoc 0: Bundling data, added 0/10412/100/A23134F8, outstanding 100
SCTP: Assoc 0: Bundling data, added 5/10443/100/A23134F9, outstanding 200
```

```
SCTP: Assoc 0: Bundling data, added 4/10545/100/A23134FA, outstanding 300
SCTP: Assoc 0: Bundling data, added 10/10371/100/A23134FB, outstanding 400
SCTP: Assoc 0: Bundling data, added 11/10382/100/A23134FC, outstanding 500
SCTP: Assoc 0: Process Sack Chunk, CumTSN=A231350F, numFrags=0
SCTP: Assoc 0: Reset oldest chunk on addr 10.5.0.4 to A2313510
SCTP: Assoc 0: Process Sack Chunk, CumTSN=A2313527, numFrags=0
SCTP: Assoc 0: Reset oldest chunk on addr 10.5.0.4 to A2313528
SCTP: Assoc 0: Process Sack Chunk, CumTSN=A231353F, numFrags=0
SCTP: Assoc 0: Reset oldest chunk on addr 10.5.0.4 to A2313540
SCTP: Assoc 0: Process Sack Chunk, CumTSN=A2313557, numFrags=0
SCTP: Assoc 0: Reset oldest chunk on addr 10.5.0.4 to A2313558
SCTP: Assoc 0: ApplSend, chunk: 10/10385/100/A23135BE to 10.5.0.4
SCTP: Assoc 0: ApplSend, chunk: 8/10230/100/A23135BF to 10.5.0.4
SCTP: Assoc 0: ApplSend, chunk: 5/10459/100/A23135C0 to 10.5.0.4
SCTP: Assoc 0: ApplSend, chunk: 4/10558/100/A23135C1 to 10.5.0.4
SCTP: Assoc 0: Set oldest chunk for dest 10.5.0.4 to TSN A231355D
SCTP: Assoc 0: Bundling data, added 5/10449/100/A231355D, outstanding 100
SCTP: Assoc 0: Bundling data, added 3/10490/100/A231355E, outstanding 200
SCTP: Assoc 0: Process Sack Chunk, CumTSN=A23135A4, numFrags=0
SCTP: Assoc 0: Reset oldest chunk on addr 10.5.0.4 to A23135A5
SCTP: Assoc 0: Process Sack Chunk, CumTSN=A23135BC, numFrags=0
SCTP: Assoc 0: Reset oldest chunk on addr 10.5.0.4 to A23135BD
SCTP: Assoc 0: Process Sack Chunk, CumTSN=A23135C1, numFrags=0
SCTP: Assoc 0: ApplSend, chunk: 5/10460/100/A23135C2 to 10.5.0.4
SCTP: Assoc 0: ApplSend, chunk: 5/10461/100/A23135C3 to 10.5.0.4
SCTP: Assoc 0: ApplSend, chunk: 11/10403/100/A2313626 to 10.5.0.4
SCTP: Assoc 0: Set oldest chunk for dest 10.5.0.4 to TSN A23135C2
SCTP: Assoc 0: Bundling data, added 5/10460/100/A23135C2, outstanding 100
SCTP: Assoc 0: Bundling data, added 5/10461/100/A23135C3, outstanding 200
SCTP: Assoc 0: Bundling data, added 5/10462/100/A23135C4, outstanding 300
SCTP: Assoc 0: Bundling data, added 4/10559/100/A23135C5, outstanding 400
SCTP: Assoc 0: Bundling data, added 4/10560/100/A23135C6, outstanding 500
SCTP: Assoc 0: Bundled 12 chunk(s) in next dgram to 10.5.0.4
SCTP: Assoc 0: Bundling data, added 1/10418/100/A2313622, outstanding 9700
SCTP: Assoc 0: Bundling data, added 3/10502/100/A2313623, outstanding 9800
SCTP: Assoc 0: Bundling data, added 7/10482/100/A2313624, outstanding 9900
SCTP: Assoc 0: Bundling data, added 3/10503/100/A2313625, outstanding 10000
SCTP: Assoc 0: Bundling data, added 11/10403/100/A2313626, outstanding 10100
SCTP: Assoc 0: Bundled 5 chunk(s) in next dgram to 10.5.0.4
SCTP: Assoc 0: Mark chunk A23135C2 for retrans
SCTP: Assoc 0: Mark chunk A23135C3 for retrans
SCTP: Assoc 0: Mark chunk A23135C4 for retrans
SCTP: Assoc 0: Mark chunk A23135C5 for retrans
SCTP: Assoc 0: Mark chunk A23135C6 for retrans
SCTP: Assoc 0: Mark chunk A23135C7 for retrans
SCTP: Assoc 0: Mark chunk A23135C8 for retrans
SCTP: Assoc 0: Mark chunk A23135C9 for retrans
SCTP: Assoc 0: Mark chunk A23135CA for retrans
SCTP: Assoc 0: Bundled 6 chunk(s) in next dgram to 10.6.0.4
SCTP: Assoc 0: Mark chunk A23135C2 for retrans
SCTP: Assoc 0: Mark chunk A23135C3 for retrans
SCTP: Assoc 0: Mark chunk A23135C4 for retrans
```

The table below describes the significant fields shown in the display.

Table 61: debug ip sctp sndchunks Field Descriptions

Field	Description
0 / 10412 / 100 / A23134F8	Stream number / datagram sequence number / chunk length, in bytes / chunk transmission sequence number.
outstanding	Number of bytes outstanding to the specified destination address.
CumTSN	Cumulative transmission sequence number (TSN).

Field	Description
numFrags	Number of fragments sent.

Related Commands

Command	Description
clear ip sctp statistics	Empties the buffer that holds SCTP statistics.
debug ip sctp congestion	Shows a list of all current SCTP associations.
show ip sctp association parameters	Shows the parameters configured for the association defined by the association identifier.
show ip sctp association statistics	Shows the current statistics for the association defined by the association identifier.
show ip sctp errors	Shows error counts logged by SCTP.
show ip sctp instances	Shows all currently defined SCTP instances.
show ip sctp statistics	Shows overall statistics counts for SCTP.
show iua as	Shows information about the current condition of an application server.
show iua asp	Shows information about the current condition of an application server process.

debug ip sctp state

To show state transitions in the Stream Control Transmission Protocol (SCTP), use the **debug ip sctp state**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip sctp state

no debug ip sctp state

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(4)T	This command was introduced.

Usage Guidelines

The **debug ip sctp state** command can be used to see if the current associations are stable or not. Because it generates output only on state transitions, it is safe to use in a live environment. It still should be used with caution, however, depending on the number of associations being handled by the system and the stability of the network.

The **debug ip sctp state** command is often used at the same time as the **debug ip sctp signal** command. Using the two commands together gives good insight into the stability of associations.

Examples

In the following example, a new association is requested and established. The peer then restarts the association and notes that the association failed and is being reestablished. The local peer then indicates that the association has failed because it has tried to retransmit the specified chunk more than the maximum number of times without success. As a result, the association fails (because of communication loss) and is terminated. The upper-layer protocol (ULP) requests that the association be attempted again, and this attempt succeeds. A shutdown is then received from the remote peer, and the local peer enters the shutdown acknowledge sent state, which is followed by the association being terminated. Again, another association attempt is made and succeeds.

```
Router# debug ip sctp signal
Router# debug ip sctp state
<new assoc attempt>
00:20:08: SCTP: Assoc 0:  state CLOSED -> COOKIE_WAIT
00:20:15: SCTP: Assoc 0:  state COOKIE_WAIT -> ESTABLISHED
00:20:15: SCTP: Assoc 0: Sent ASSOC_UP signal for CONFIGD_ASSOC
00:21:03: SCTP: Assoc 0: Restart rcvd from peer
00:21:03: SCTP: Assoc 0: Sent ASSOC_RESTART signal
00:21:04: SCTP: Assoc 0: chunk 62EA7F40 retransmitted more than max times, failing assoc
00:21:04: SCTP: Assoc 0: Sent ASSOC_FAILED signal, reason: SCTP_COMM_LOST
00:21:04: SCTP: Assoc 0: Sent ASSOC_TERMINATE signal
```

```
00:21:04: SCTP: Assoc 0:  state ESTABLISHED -> CLOSED
<new assoc attempt>
00:21:04: SCTP: Assoc 0:  state CLOSED -> COOKIE_WAIT
00:21:04: SCTP: Assoc 0:  state COOKIE_WAIT -> COOKIE_ECHOED
00:21:04: SCTP: Assoc 0:  state COOKIE_ECHOED -> ESTABLISHED
00:21:04: SCTP: Assoc 0: Sent ASSOC_UP signal for CONFIGD_ASSOC
00:21:04: SCTP: Assoc 0: Sent TERMINATE_PENDING signal
00:21:04: SCTP: Assoc 0:  state ESTABLISHED -> SHUTDOWN_ACKSENT
00:21:04: SCTP: Assoc 0: Sent ASSOC_TERMINATE signal
00:21:04: SCTP: Assoc 0:  state SHUTDOWN_ACKSENT -> CLOSED
<new assoc attempt>
00:21:04: SCTP: Assoc 0:  state CLOSED -> COOKIE_WAIT
00:21:04: SCTP: Assoc 0:  state COOKIE_WAIT -> COOKIE_ECHOED
00:21:04: SCTP: Assoc 0:  state COOKIE_ECHOED -> ESTABLISHED
00:21:04: SCTP: Assoc 0: Sent ASSOC_UP signal for CONFIGD_ASSOC
```

The table below describes the significant fields shown in the display.

Table 62: debug ip sctp state Field Descriptions

Field	Description
CLOSED -> COOKIE_WAIT	SCTP endpoint sends initialization chunk and moves to the COOKIE_WAIT state to wait for acknowledgment and a state cookie from the remote endpoint.
COOKIE_WAIT -> COOKIE_ECHOED	SCTP endpoint returns the state cookie to the remote endpoint and enters COOKIE_ECHOED state.
COOKIE_ECHOED -> ESTABLISHED	SCTP endpoint enters ESTABLISHED state after receiving acknowledgment that the state cookie has been received by the remote endpoint.
ESTABLISHED -> SHUTDOWN_ACKSENT	SCTP endpoint enters SHUTDOWN_ACKSENT state after receiving a shutdown message and sending a shutdown acknowledgment to the remote endpoint.
SHUTDOWN_ACKSENT -> CLOSED	SCTP endpoint enters CLOSED state.

Related Commands

Command	Description
clear ip sctp statistics	Empties the buffer that holds SCTP statistics.
debug ip sctp congestion	Shows a list of all current SCTP associations.
debug ip sctp signal	Shows signals that are sent from SCTP to the application or ULP.
show ip sctp association parameters	Shows the parameters configured for the association defined by the association identifier.
show ip sctp association statistics	Shows the current statistics for the association defined by the association identifier.

Command	Description
show ip sctp errors	Shows error counts logged by SCTP.
show ip sctp instances	Shows all currently defined SCTP instances.
show ip sctp statistics	Shows overall statistics counts for SCTP.
show iua as	Shows information about the current condition of an application server.
show iua asp	Shows information about the current condition of an application server process.

debug ip sctp timer

To provide information about Stream Control Transmission Protocol (SCTP) timers that are started, stopped, and triggering, use the **debug ip sctp timer**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip sctp timer

no debug ip sctp timer

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(4)T	This command was introduced.

Usage Guidelines

Many SCTP timers should not be restarted after they have been started once. For these timers, the first call succeeds in starting the timer, and subsequent calls do nothing until the timer either expires or is stopped. For example, the retransmission timer is started when the first chunk is sent, but then is not started again for subsequent chunks when there is outstanding data.

Caution The **debug ip sctp timer** command generates a significant amount of output. It should be used with extreme caution in a live network.

Examples

The following example shows the starting and stopping of various SCTP timers:

```
Router# debug ip sctp timer
SCTP: Assoc 0: Starting CUMSACK timer
SCTP: Timer already started, not restarting
SCTP: Assoc 0: Starting CUMSACK timer
SCTP: Timer already started, not restarting
SCTP: Assoc 0: Timer BUNDLE triggered
SCTP: Assoc 0: Starting RETRANS timer for destaddr 10.5.0.4
SCTP: Assoc 0: Starting RETRANS timer for destaddr 10.5.0.4
SCTP: Timer already started, not restarting
SCTP: Assoc 0: Starting RETRANS timer for destaddr 10.5.0.4
SCTP: Timer already started, not restarting
SCTP: Assoc 0: Starting RETRANS timer for destaddr 10.5.0.4
SCTP: Timer already started, not restarting
SCTP: Assoc 0: Stopping RETRANS timer for destaddr 10.5.0.4
SCTP: Assoc 0: Starting RETRANS timer for destaddr 10.5.0.4
```

```
SCTP: Assoc 0: Stopping RETRANS timer for destaddr 10.5.0.4
SCTP: Assoc 0: Starting CUMSACK timer
SCTP: Timer already started, not restarting
SCTP: Assoc 0: Starting CUMSACK timer
SCTP: Timer already started, not restarting
SCTP: Assoc 0: Starting CUMSACK timer
SCTP: Timer already started, not restarting
SCTP: Assoc 0: Starting CUMSACK timer
SCTP: Timer already started, not restarting
SCTP: Assoc 0: Starting CUMSACK timer
SCTP: Timer already started, not restarting
SCTP: Assoc 0: Stopping CUMSACK timer
SCTP: Assoc 0: Starting CUMSACK timer
SCTP: Assoc 0: Starting CUMSACK timer
SCTP: Timer already started, not restarting
```

The table below describes the significant fields shown in the display.

Table 63: debug ip sctp timer Field Descriptions

Field	Description
CUMSACK	Cumulative selective acknowledgment.
RETRANS	Retransmission.

Related Commands

Command	Description
clear ip sctp statistics	Empties the buffer that holds SCTP statistics.
debug ip sctp congestion	Shows a list of all current SCTP associations.
show ip sctp association parameters	Shows the parameters configured for the association defined by the association identifier.
show ip sctp association statistics	Shows the current statistics for the association defined by the association identifier.
show ip sctp errors	Shows error counts logged by SCTP.
show ip sctp instances	Shows all currently defined SCTP instances.
show ip sctp statistics	Shows overall statistics counts for SCTP.
show iua as	Shows information about the current condition of an application server.
show iua asp	Shows information about the current condition of an application server process.

debug ip sctp warnings

To display diagnostic information about unusual situations in Stream Control Transmission Protocol (SCTP), use the **debug ip sctp warnings**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip sctp warnings

no debug ip sctp warnings

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(4)T	This command was introduced.

Usage Guidelines

In a live system, the debugging messages for performance, state, signal, and warnings are the most useful. They show any association or destination address failures and can be used to monitor the stability of established associations.

The **debug ip sctp warnings** command displays information on any unusual situation that is encountered. These situations may or may not indicate problems, depending on the particulars of the situation.

Examples

The following example shows some events and conditions that are flagged as warnings:

```
Router# debug ip sctp warnings
SCTP: Assoc 0: No cookie in InitAck, discarding
SCTP: Assoc 0: Incoming INIT_ACK: inbound streams reqd 15, allowed 13
SCTP: Assoc 0: Incoming INIT_ACK request: outbound streams req'd 13, allowed 1
SCTP: Assoc 0: Remote verification tag in init ack is zero, discarding
SCTP: Remote verification tag in init is zero, discarding
SCTP: Assoc 0: Rwnd less than min allowed (1500) in incoming INITACK, rcvd 0
SCTP: Assoc 0: Rwnd less than min allowed (1500) in incoming INITACK, rcvd 1499
SCTP: Rwnd in INIT too small (0), discarding
SCTP: Rwnd in INIT too small (1499), discarding
SCTP: Unknown INIT param 16537 (0x4099), length 8
SCTP: Assoc 0: Unknown INITACK param 153 (0x99), length 8
SCTP: Assoc 0: No cookie in InitAck, discarding
SCTP: Assoc 0: No cookie in InitAck, discarding
SCTP: Processing INIT, invalid param len 0, discarding...
SCTP: Assoc 0: Processing INITACK, invalid param len 0, discarding...
```

Related Commands

Command	Description
clear ip sctp statistics	Empties the buffer that holds SCTP statistics.
debug ip sctp congestion	Shows a list of all current SCTP associations.
show ip sctp association parameters	Shows the parameters configured for the association defined by the association identifier.
show ip sctp association statistics	Shows the current statistics for the association defined by the association identifier.
show ip sctp errors	Shows error counts logged by SCTP.
show ip sctp instances	Shows all currently defined SCTP instances.
show ip sctp statistics	Shows overall statistics counts for SCTP.
show iua as	Shows information about the current condition of an application server.
show iua asp	Shows information about the current condition of an application server process.

debug ip sd

To display all session directory (SD) announcements received, use the **debug ip sd**command in privileged EXEC mode. To disable debugging output, use the **no**form of this command.

debug ip sd

no debug ip sd

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

This command shows session directory announcements for multicast IP. Use it to observe multicast activity.

Examples

The following is sample output from the **debug ip sd** command:

```
Router# debug ip sd
SD: Announcement from 172.16.58.81 on Serial0.1, 146 bytes
    s=*cisco: CBONE Audio
    i=cisco internal-only audio conference
    o=dino@dino-ss20.cisco.com
    c=224.0.255.1 16 2891478496 2892688096
    m=audio 31372 1700
SD: Announcement from 172.22.246.68 on Serial0.1, 147 bytes
    s=IMS: U.S. Senate
    i=U.S. Senate at http://town.hall.org/radio/live.html
    o=carl@also.radio.com
    c=224.2.252.231 95 0 0
    m=audio 36572 2642
    a=fmt:gsm
```

The table below describes the significant fields shown in the display.

Table 64: debug ip sd Field Descriptions

Field	Description
SD	Session directory event.
Announcement from	Address sending the SD announcement.
on Serial0.1	Interface receiving the announcement.
146 bytes	Size of the announcement event.
s=	Session name being advertised.
i=	Information providing a descriptive name for the session.

Field	Description
o=	Origin of the session, either an IP address or a name.
c=	Connect description showing address and number of hops.
m=	Media description that includes media type, port number, and ID.

Related Commands

Command	Description
debug ip dvmrp	Displays information on DVMRP packets received and sent.
debug ip igmp	Displays IGMP packets received and sent, and IGMP host-related events.
debug ip mbgp dampening	Logs route flap dampening activity related to MBGP.
debug ip mrouting	Displays changes to the IP multicast routing table.
debug ip pim	Displays PIM packets received and sent, and PIM-related events.

debug ip sdee

To enable debugging messages for Security Device Event Exchange (SDEE) notification events, use the **debug ip sdee** command in privileged EXEC mode. To disable SDEE debugging messages, use the **no** form of this command.

debug ip sdee [alerts] [detail] [messages] [requests] [subscriptions]

no debug ip sdee [alerts] [detail] [messages] [requests] [subscriptions]

Syntax Description

alerts	Displays new alerts that are reported to SDEE from IPS.
detail	Displays detailed SDEE messages.
messages	Displays error and status messages that are reported to SDEE from IPS.
requests	Displays SDEE client requests.
subscriptions	Displays SDEE client subscription requests.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.3(8)T	This command was introduced.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.

Examples

The following is sample SDEE debug output. In this example, you can see which messages correspond to SDEE alerts, requests, and subscriptions.

```
Router# debug ip sdee alerts requests subscriptions
5d00h:SDEE:got request from client at 10.0.0.2
5d00h:SDEE:reported 13 events for client at 10.0.0.2
5d00h:SDEE:GET request for client 10.0.0.2 subscription IDS1720:0
5d00h:SDEE:reported 50 events for client 10.0.0.2 subscription IDS1720:0
5d00h: SDEE alert:sigid 2004 name ICMP Echo Req from 10.0.0.2 time 1021174067
5d00h: SDEE alert:sigid 2004 name ICMP Echo Req from 10.0.0.2 time 1021174071
5d00h: SDEE alert:sigid 2004 name ICMP Echo Req from 10.0.0.2 time 1021174072
5d00h: SDEE alert:sigid 2004 name ICMP Echo Req from 10.0.0.2 time 1021175127
5d00h:SDEE:missed events for IDS1720:0
```

Related Commands

Command	Description
ip ips notify	Specifies the method of event notification.
ip sdee events	Sets the maximum number of SDEE events that can be stored in the event buffer.
ip sdee subscriptions	Sets the maximum number of SDEE subscriptions that can be open simultaneously.

debug ip security

To display IP security option processing, use the **debug ip security** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip security

no debug ip security

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

The **debug ip security** command displays information for both basic and extended IP security options. For interfaces where **ip security** is configured, each IP packet processed for that interface results in debugging output regardless of whether the packet contains IP security options. IP packets processed for other interfaces that also contain IP security information also trigger debugging output. Some additional IP security debugging information is also controlled by the **debug ip packet** command in privileged EXEC mode.

Caution

Because the **debug ip security** command generates a substantial amount of output for every IP packet processed, use it only when traffic on the IP network is low, so other activity on the system is not adversely affected.

Examples

The following is sample output from the **debug ip security** command:

```
Router# debug ip security
IP Security: src 172.24.72.52 dst 172.24.72.53, number of BSO 1
     idb: NULL
     pak: insert (0xFF) 0x0
IP Security: BSO postroute: SECINSERT changed to secret (0x5A) 0x10
IP Security: src 172.24.72.53 dst 172.24.72.52, number of BSO 1
     idb: secret (0x6) 0x10 to secret (0x6) 0x10, no implicit
          def secret (0x6) 0x10
     pak: secret (0x5A) 0x10
IP Security: checking BSO 0x10 against [0x10 0x10]
IP Security: classified BSO as secret (0x5A) 0x10
```

The table below describes significant fields shown in the display.

Table 65: debug ip security Field Descriptions

Field	Description
number of BSO	Indicates the number of basic security options found in the packet.
idb	Provides information on the security configuration for the incoming interface.

Field	Description
pak	Provides information on the security classification of the incoming packet.
src	Indicates the source IP address.
dst	Indicates the destination IP address.

The following line indicates that the packet was locally generated, and it has been classified with the internally significant security level “insert” (0xff) and authority information of 0x0:

```
idb: NULL
pak: insert (0xff) 0x0
```

The following line indicates that the packet was received via an interface with dedicated IP security configured. Specifically, the interface is configured at security level “secret” and with authority information of 0x0. The packet itself was classified at level “secret” (0x5a) and authority information of 0x10.

```
idb: secret (0x6) 0x10 to secret (0x6) 0x10, no implicit
     def secret (0x6) 0x10
pak: secret (0x5A) 0x10
```

debug ip sla error

To enable debugging output of Cisco IOS IP Service Level Agreements (SLAs) operation run-time errors, use the **debug ip sla error** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip sla error [*operation-number*| **ep-api**| **event-publisher**]

no debug ip sla error [*operation-number*| **ep-api**| **event-publisher**]

Syntax Description

operation-number	(Optional) Identification number of the operation for which debugging output is to be enabled.
ep-api	(Optional) Enables IP SLAs Event Publisher application programming interface (API) debug messages.
event-publisher	(Optional) Enables IP SLAs Event Publisher debug messages.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.4(4)T	This command was introduced. This command replaces the **debug ip sla monitor error**command.
12.0(32)SY	This command was integrated into Cisco IOS Release 12.0(32)SY.
12.2(33)SRB	This command was integrated into Cisco IOS Release 12.2(33)SRB. This command replaces the **debug rtr error**command.
12.2(33)SB	This command was integrated into Cisco IOS Release 12.2(33)SB. This command replaces the **debug ip sla monitor error** command.
12.2(33)SXI	This command was integrated into Cisco IOS Release 12.2(33)SXI. This command replaces the **debug ip sla monitor error** command.
12.4(22)T	This command was modified. The **ep-api** and **event-publisher** keywords were added.
12.2(33)SRE	This command was modified. The **ep-api** and **event-publisher** keywords were added.

Usage Guidelines

The **debug ip sla error** *operation-number*command displays run-time errors. When an operation number other than 0 is specified, all run-time errors for that operation are displayed when the operation is active. When the operation number is 0, all run-time errors relating to the IP SLAs scheduler process are displayed. When no operation number is specified, all run-time errors for all active operations configured on the router are displayed.

Note Use the **debug ip sla error**command before using the **debug ip sla trace** command because the **debug ip sla error** command generates a lesser amount of debugging output.

The **debug ip sla error** command is supported in IPv4 networks. This command can also be used to enable debugging output for an IP SLAs operation that supports IPv6 addresses.

Examples

The following is sample output from the **debug ip sla error** command. The output indicates failure because the target is not there or because the responder is not enabled on the target.

```
Router# debug ip sla error
May  5 05:00:35.483: control message failure:1
May  5 05:01:35.003: control message failure:1
May  5 05:02:34.527: control message failure:1
May  5 05:03:34.039: control message failure:1
May  5 05:04:33.563: control message failure:1
May  5 05:05:33.099: control message failure:1
May  5 05:06:32.596: control message failure:1
May  5 05:07:32.119: control message failure:1
May  5 05:08:31.643: control message failure:1
May  5 05:09:31.167: control message failure:1
May  5 05:10:30.683: control message failure:1
```

Related Commands

Command	Description
debug ip sla trace	Traces the execution of an IP SLAs operation.

debug ip sla ethernet-monitor

To enable debugging output for a Cisco IOS IP Service Level Agreements (SLAs) Ethernet operation, use the **debug ip sla ethernet-monitor** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip sla ethernet-monitor [*operation-number*]

no debug ip sla ethernet-monitor [*operation-number*]

Syntax Description

operation-number	(Optional) Number of the Ethernet operation for which the debugging output will be displayed.

Command Default

Debugging activity for a Cisco IOS IP SLAs Ethernet operation does not occur.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(33)SRB	This command was introduced.
12.2(33)SB	This command was integrated into Cisco IOS Release 12.2(33)SB.
12.4(20)T	This command was integrated into Cisco IOS Release 12.4(20)T.
12.2(33)SXI	This command was integrated into Cisco IOS Release 12.2(33)SXI.

Examples

The following is sample output from the **debug ip sla ethernet-monitor** command:

```
Router# debug ip sla ethernet-monitor
00:00:15: IP SLAs Auto Ethernet(0):vlan = 2, domain = DOMAIN_OPERATOR_L3_1, mpid = 6322
                                 from CFM
00:00:15: IP SLAs Auto Ethernet(0):saaHandleEventFromCFM::Received Event from CFM
00:00:15: IP SLAs Auto Ethernet(0):Event::ECFM_SAA_EV_MEP_ADD
00:00:15: IP SLAs Auto Ethernet(0):1 auto-probes found for domain = DOMAIN_OPERATOR_L3_1
and vlan = 2
00:00:15: IP SLAs Auto Ethernet(0):autoProbe probe_id = 1
00:00:15: IP SLAs Auto Ethernet(0):0 Probes already running in auto-probe = 1
00:00:15: IP SLAs Auto Ethernet(1):starting probe with freq = 20 sec
00:00:15: IP SLAs Auto Ethernet(1):starting probe 100001
```

Related Commands

Command	Description
ip sla	Begins configuration for an IP SLAs operation and enters IP SLA configuration mode.
ip sla ethernet-monitor	Begins configuration for an IP SLAs auto Ethernet operation and enters IP SLA Ethernet monitor configuration mode.

debug ip sla monitor error

Note Effective with Cisco IOS Release 12.4(4)T, 12.2(33)SB, and 12.2(33)SXI, the **debug ip sla monitor error**command is replaced by the **debug ip sla error**command. See the **debug ip sla error**command for more information.

To enable debugging output of Cisco IOS IP Service Level Agreements (SLAs) operation run-time errors, use the **debug ip sla monitor error**command in privileged EXEC mode. To disable debugging output, use the **no**form of this command.

debug ip sla monitor error [*operation-number*]

no debug ip sla monitor error [*operation-number*]

Syntax Description

operation-number	(Optional) Identification number of the operation for which debugging output is to be enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(14)T	This command was introduced. This command replaces the **debug rtr error**command.
12.4(4)T	This command was replaced by the **debug ip sla error**command.
12.2(31)SB2	This command was integrated into Cisco IOS Release 12.2(31)SB2.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.
12.2(33)SB	This command was replaced by the **debug ip sla error** command.
12.2(33)SXI	This command was replaced by the **debug ip sla error** command.

Usage Guidelines

The **debug ip sla monitor error**command displays run-time errors. When an operation number other than 0 is specified, all run-time errors for that operation are displayed when the operation is active. When the operation number is 0, all run-time errors relating to the IP SLAs scheduler process are displayed. When no operation number is specified, all run-time errors for all active operations configured on the router are displayed.

Note Use the **debug ip sla monitor error**command before using the **debug ip sla monitor trace** command because the **debug ip sla monitor error** command generates a lesser amount of debugging output.

Examples

The following is sample output from the **debug ip sla monitor error** command. The output indicates failure because the target is not there or because the responder is not enabled on the target. All debugging output for IP SLAs (including the output from the **debug ip sla monitor trace** command) has the format shown in the table below.

```
Router# debug ip sla monitor error
May  5 05:00:35.483: control message failure:1
May  5 05:01:35.003: control message failure:1
May  5 05:02:34.527: control message failure:1
May  5 05:03:34.039: control message failure:1
May  5 05:04:33.563: control message failure:1
May  5 05:05:33.099: control message failure:1
May  5 05:06:32.596: control message failure:1
May  5 05:07:32.119: control message failure:1
May  5 05:08:31.643: control message failure:1
May  5 05:09:31.167: control message failure:1
May  5 05:10:30.683: control message failure:1
```

The table below describes the significant fields shown in the display.

Table 66: debug ip sla monitor error Field Descriptions

Field	Description
IP SLA Monitor 1	Number of the operation generating the message.
Error Return Code	Message identifier indicating the error type (or error itself).
LU0 IP SLA Monitor Probe 1	Name of the process generating the message.
in echoTarget on call luReceive LuApiReturnCode of InvalidHandle - invalid host name or API handle	Supplemental messages that pertain to the message identifier.

Related Commands

Command	Description
debug ip sla monitor trace	Traces the execution of an IP SLAs operation.

debug ip sla monitor mpls-lsp-monitor

Note Effective with Cisco IOS Release 12.2(33)SB, the **debug ip sla monitor mpls-lsp-monitor**command is replaced by the **debug ip sla mpls-lsp-monitor**command. See the **debug ip sla mpls-lsp-monitor**command for more information.

To enable debugging output for the IP Service Level Agreements (SLAs) label switched path (LSP) Health Monitor, use the **debug ip sla monitor mpls-lsp-monitor**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip sla monitor mpls-lsp-monitor [*operation-number*]

no debug ip sla monitor mpls-lsp-monitor [*operation-number*]

Syntax Description

operation-number	(Optional) Number of the LSP Health Monitor operation for which the debugging output will be displayed.

Command Default

Debugging is disabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(31)SB2	This command was introduced.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.
12.2(33)SB	This command was replaced by the **debug ip sla mpls-lsp-monitor** command.

Examples

The following is sample output from the **debug ip sla monitor mpls-lsp-monitor** command:

```
Router# debug ip sla monitor mpls-lsp-monitor
IP SLA Monitor MPLSLM debugging for all entries is on
*Aug 19 19:59: IP SLA Monitor MPLSLM(1):Next hop 10.10.10.8 added in AddQ
*Aug 19 19:59: IP SLA Monitor MPLSLM(1):Next hop 10.10.10.8 added in AddQ
*Aug 19 19:59: IP SLA Monitor MPLSLM(1):Next hop 10.10.10.8 added in AddQ
*Aug 19 19:59: IP SLA Monitor MPLSLM(1):Adding vrf red into tree entry 10.10.10.8
*Aug 19 19:59: IP SLA Monitor MPLSLM(1):Adding Probe 100005
*Aug 19 19:59: IP SLA Monitor MPLSLM(1):Adding ProbeID 100005 to tree entry 10.10.10.8 (1)
*Aug 19 19:59: IP SLA Monitor MPLSLM(1):Adding vrf blue into tree entry 10.10.10.8
*Aug 19 19:59: IP SLA Monitor MPLSLM(1):Duplicate in AddQ 10.10.10.8
```

```
*Aug 19 19:59: IP SLA Monitor MPLSLM(1):Adding vrf green into tree entry 10.10.10.8
*Aug 19 19:59: IP SLA Monitor MPLSLM(1):Duplicate in AddQ 10.10.10.8
*Aug 19 19:59: IP SLA Monitor MPLSLM(1):Added Probe(s) 100005 will be scheduled after 26
secs over schedule period 60
```

Related Commands

Command	Description
auto ip sla mpls-lsp-monitor	Begins configuration for an IP SLAs LSP Health Monitor operation and enters auto IP SLA MPLS configuration mode.

debug ip sla trace

To trace the execution of a Cisco IOS IP Service Level Agreements (SLAs) operation, use the **debug ip sla trace**command in privileged EXEC mode. To disable trace debugging output, use the **no**form of this command.

debug ip sla trace [*operation-number*| **ep-api**| **event-publisher**]

no debug ip sla trace [*operation-number*| **ep-api**| **event-publisher**]

Syntax Description

operation-number	(Optional) Identification number of the operation for which debugging output is to be enabled.
ep-api	(Optional) Enables IP SLAs Event Publisher API debugging output.
event-publisher	(Optional) Enables IP SLAs Event Publisher debugging output.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.4(4)T	This command was introduced. This command replaces the **debug ip sla monitor trace**command.
12.0(32)SY	This command was integrated into Cisco IOS Release 12.0(32)SY.
12.2(33)SRB	This command was integrated into Cisco IOS Release 12.2(33)SRB. This command replaces the **debug rtr trace**command.
12.2(33)SB	This command was integrated into Cisco IOS Release 12.2(33)SB. This command replaces the **debug ip sla monitor trace** command.
12.2(33)SXI	This command was integrated into Cisco IOS Release 12.2(33)SXI. This command replaces the **debug ip sla monitor trace** command.
12.4(22)T	This command was modified. The **ap-api** and **event-publisher** keywords were added.
12.2(33)SRE	This command was modified. The **ep-api** and **event-publisher** keywords were added.

Usage Guidelines

The **debug ip sla trace** *operation-number* command traces the execution of an IP SLAs operation. When an operation number other than 0 is specified, execution for that operation is traced. When the operation number is 0, the IP SLAs scheduler process is traced. When no operation number is specified, all active operations are traced.

The **debug ip sla trace** command also enables the **debug ip sla error** command for the specified operation. However, the **no debug ip sla trace** command does not disable the **debug ip sla error** command. You must manually disable the command by using the **no debug ip sla error** command.

All debugging output (including **debug ip sla error** command output) has the format shown in the **debug ip sla error** command output example.

Note The **debug ip sla trace**command can generate a large number of debug messages. First use the **debug ip sla error** command, and then use the **debug ip sla trace** on a per-operation basis.

Examples

The following is sample output from the **debug ip sla trace** command. In this example, an operation is traced through a single operation attempt: the setup of a connection to the target, and the attempt at an echo to calculate UDP packet response time.

```
Router# debug ip sla trace
May  5 05:25:08.584:rtt hash insert :3.0.0.3 3383
May  5 05:25:08.584:    source=3.0.0.3(3383)  dest-ip=5.0.0.1(9)
May  5 05:25:08.588:sending control msg:
May  5 05:25:08.588: Ver:1 ID:51 Len:52
May  5 05:25:08.592:cmd:command:RTT_CMD_UDP_PORT_ENABLE, ip:5.0.0.1, port:9, duration:5000
May  5 05:25:08.607:receiving reply
May  5 05:25:08.607: Ver:1 ID:51 Len:8
May  5 05:25:08.623:    local delta:8
May  5 05:25:08.627:    delta from responder:1
May  5 05:25:08.627:    received <16> bytes and      responseTime = 3 (ms)
May  5 05:25:08.631:rtt hash remove:3.0.0.3 3383IP SLA Monitor 1:Starting An Echo Operation
 - IP SLA Monitor Probe 1
May  5 05:26:08.104:rtt hash insert :3.0.0.3 2974
May  5 05:26:08.104:    source=3.0.0.3(2974)  dest-ip=5.0.0.1(9)
May  5 05:26:08.108:sending control msg:
May  5 05:26:08.108: Ver:1 ID:52 Len:52
May  5 05:26:08.112:cmd:command:RTT_CMD_UDP_PORT_ENABLE, ip:5.0.0.1, port:9, duration:5000
May  5 05:26:08.127:receiving reply
May  5 05:26:08.127: Ver:1 ID:52 Len:8
May  5 05:26:08.143:    local delta:8
May  5 05:26:08.147:    delta from responder:1
May  5 05:26:08.147:    received <16> bytes and      responseTime = 3 (ms)
May  5 05:26:08.151:rtt hash remove:3.0.0.3 2974IP SLA Monitor 1:Starting An Echo Operation
 - IP SLA Monitor Probe 1
```

Related Commands

Command	Description
debug ip sla error	Enables debugging output of IP SLAs operation run-time errors.

debug ip sla mpls-lsp-monitor

Note: Effective with Cisco IOS Release 15.1(1)S, the **debug ip sla mpls-lsp-monitor** command was replaced by the **debug ip sla trace mpls-lsp-monitor** command. See the **debug ip sla trace mpls-lsp-monitor** command for more information.

To enable debugging output for the IP Service Level Agreements (SLAs) label switched path (LSP) Health Monitor, use the **debug ip sla mpls-lsp-monitor**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip sla mpls-lsp-monitor [*operation-number*]

no debug ip sla mpls-lsp-monitor [*operation-number*]

Syntax Description

operation-number	(Optional) Number of the LSP Health Monitor operation for which the debugging output will be displayed.

Command Default

Debugging is disabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.4(6)T	This command was introduced.
12.0(32)SY	This command was integrated into Cisco IOS Release 12.0(32)SY.
12.2(33)SRB	This command was integrated into Cisco IOS Release 12.2(33)SRB. This command replaces the **debug rtr mpls-lsp-monitor**command.
12.2(33)SB	This command was integrated into Cisco IOS Release 12.2(33)SB. This command replaces the **debug ip sla monitor mpls-lsp-monitor** command.
15.1(1)S	This command was replaced by the **debug ip sla trace mpls-lsp-monitor** command.

Examples

The following is sample output from the **debug ip sla mpls-lsp-monitor** command:

```
Router# debug ip sla mpls-lsp-monitor
IP SLAs MPLSLM debugging for all entries is on
```

```
*Aug 19 19:59: IP SLAs MPLSLM(1):Next hop 10.10.10.8 added in AddQ
*Aug 19 19:59: IP SLAs MPLSLM(1):Next hop 10.10.10.8 added in AddQ
*Aug 19 19:59: IP SLAs MPLSLM(1):Next hop 10.10.10.8 added in AddQ
*Aug 19 19:59: IP SLAs MPLSLM(1):Adding vrf red into tree entry 10.10.10.8
*Aug 19 19:59: IP SLAs MPLSLM(1):Adding Probe 100005
*Aug 19 19:59: IP SLAs MPLSLM(1):Adding ProbeID 100005 to tree entry 10.10.10.8 (1)
*Aug 19 19:59: IP SLAs MPLSLM(1):Adding vrf blue into tree entry 10.10.10.8
*Aug 19 19:59: IP SLAs MPLSLM(1):Duplicate in AddQ 10.10.10.8
*Aug 19 19:59: IP SLAs MPLSLM(1):Adding vrf green into tree entry 10.10.10.8
*Aug 19 19:59: IP SLAs MPLSLM(1):Duplicate in AddQ 10.10.10.8
*Aug 19 19:59: IP SLAs MPLSLM(1):Added Probe(s) 100005 will be scheduled after 26 secs over
 schedule period 60
```

Related Commands

Command	Description
debug ip sla trace mpls-lsp-monitor	Traces the execution of an IP SLAs LSP Health Monitor operation.

debug ip sla trace

To trace the execution of a Cisco IOS IP Service Level Agreements (SLAs) operation, use the **debug ip sla trace**command in privileged EXEC mode. To disable trace debugging output, use the **no**form of this command.

debug ip sla trace [*operation-number*| **ep-api**| **event-publisher**]

no debug ip sla trace [*operation-number*| **ep-api**| **event-publisher**]

Syntax Description

operation-number	(Optional) Identification number of the operation for which debugging output is to be enabled.
ep-api	(Optional) Enables IP SLAs Event Publisher API debugging output.
event-publisher	(Optional) Enables IP SLAs Event Publisher debugging output.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.4(4)T	This command was introduced. This command replaces the **debug ip sla monitor trace**command.
12.0(32)SY	This command was integrated into Cisco IOS Release 12.0(32)SY.
12.2(33)SRB	This command was integrated into Cisco IOS Release 12.2(33)SRB. This command replaces the **debug rtr trace**command.
12.2(33)SB	This command was integrated into Cisco IOS Release 12.2(33)SB. This command replaces the **debug ip sla monitor trace** command.
12.2(33)SXI	This command was integrated into Cisco IOS Release 12.2(33)SXI. This command replaces the **debug ip sla monitor trace** command.
12.4(22)T	This command was modified. The **ap-api** and **event-publisher** keywords were added.
12.2(33)SRE	This command was modified. The **ep-api** and **event-publisher** keywords were added.

Usage Guidelines

The **debug ip sla trace** *operation-number* command traces the execution of an IP SLAs operation. When an operation number other than 0 is specified, execution for that operation is traced. When the operation number is 0, the IP SLAs scheduler process is traced. When no operation number is specified, all active operations are traced.

The **debug ip sla trace** command also enables the **debug ip sla error** command for the specified operation. However, the **no debug ip sla trace** command does not disable the **debug ip sla error** command. You must manually disable the command by using the **no debug ip sla error** command.

All debugging output (including **debug ip sla error** command output) has the format shown in the **debug ip sla error** command output example.

Note The **debug ip sla trace**command can generate a large number of debug messages. First use the **debug ip sla error** command, and then use the **debug ip sla trace** on a per-operation basis.

Examples

The following is sample output from the **debug ip sla trace** command. In this example, an operation is traced through a single operation attempt: the setup of a connection to the target, and the attempt at an echo to calculate UDP packet response time.

```
Router# debug ip sla trace
May  5 05:25:08.584:rtt hash insert :3.0.0.3 3383
May  5 05:25:08.584:    source=3.0.0.3(3383)  dest-ip=5.0.0.1(9)
May  5 05:25:08.588:sending control msg:
May  5 05:25:08.588: Ver:1 ID:51 Len:52
May  5 05:25:08.592:cmd:command:RTT_CMD_UDP_PORT_ENABLE, ip:5.0.0.1, port:9, duration:5000
May  5 05:25:08.607:receiving reply
May  5 05:25:08.607: Ver:1 ID:51 Len:8
May  5 05:25:08.623:    local delta:8
May  5 05:25:08.627:    delta from responder:1
May  5 05:25:08.627:    received <16> bytes and      responseTime = 3 (ms)
May  5 05:25:08.631:rtt hash remove:3.0.0.3 3383IP SLA Monitor 1:Starting An Echo Operation
 - IP SLA Monitor Probe 1
May  5 05:26:08.104:rtt hash insert :3.0.0.3 2974
May  5 05:26:08.104:    source=3.0.0.3(2974)  dest-ip=5.0.0.1(9)
May  5 05:26:08.108:sending control msg:
May  5 05:26:08.108: Ver:1 ID:52 Len:52
May  5 05:26:08.112:cmd:command:RTT_CMD_UDP_PORT_ENABLE, ip:5.0.0.1, port:9, duration:5000
May  5 05:26:08.127:receiving reply
May  5 05:26:08.127: Ver:1 ID:52 Len:8
May  5 05:26:08.143:    local delta:8
May  5 05:26:08.147:    delta from responder:1
May  5 05:26:08.147:    received <16> bytes and      responseTime = 3 (ms)
May  5 05:26:08.151:rtt hash remove:3.0.0.3 2974IP SLA Monitor 1:Starting An Echo Operation
 - IP SLA Monitor Probe 1
```

Related Commands

Command	Description
debug ip sla error	Enables debugging output of IP SLAs operation run-time errors.

debug ip sla trace mpls-lsp-monitor

To trace the execution of an IP Service Level Agreements (SLAs) label switched path (LSP) Health Monitor operation, use the **debug ip sla trace mpls-lsp-monitor**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip sla trace mpls-lsp-monitor [*operation-number*]

no debug ip sla mpls-lsp-monitor

Syntax Description

operation-number	(Optional) Number of the LSP Health Monitor operation for which the debugging output will be displayed. The range is 0 to 2147483647.

Command Default

Trace debugging of IP SLAs LSP Health Monitor operations is disabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
15.1(1)S	This command was introduced. This command replaces the **debug ip sla mpls-lsp-monitor** command.

Usage Guidelines

For Cisco IP SLAs Engine 3.0 in Cisco IOS Release 15.1(1)S, this command replaces the **debug ip sla mpls-lsp-monitor** command.

To determine the IP SLAs engine version, IP SLAs Engine 2.0 or 3.0, running on your Cisco router, use the **show ip application** command in privileged EXEC mode, as shown in the following example:

```
Router# show ip sla application
 IP Service Level Agreements
Version: Round Trip Time MIB 2.2.0, Infrastructure Engine-III
```

The **debug ip sla trace mpls-lsp-monitor** command traces the execution of IP SLAs LSP Health Monitor operations. When an operation number other than 0 is specified, execution for that operation is traced. When the operation number is 0, the IP SLAs scheduler process is traced. When no operation number is specified, all active LSP Health Monitor operations are traced.

This command also enables the **debug ip sla error** command for the specified operation. However, the **no debug ip sla trace mpls-lsp-monitor** command does not disable the **debug ip sla error** command. You must manually disable the command by using the **no debug ip sla error**command.

The **debug ip sla trace mpls-lsp-monitor** command can generate a large number of debug messages. To help reduce the number of debug messages, first use the **debug ip sla error** command and then use the **debug ip sla trace mpls-lsp-monitor** command on a per-operation basis.

Examples

The following is sample output from the **debug ip sla trace mpls-lsp-monitor** command:

```
Router# debug ip sla trace mpls-lsp-monitor
IP SLA Monitor MPLSLM debugging for all entries is on
*Aug 19 19:59: IP SLA Monitor MPLSLM(1):Next hop 10.10.10.8 added in AddQ
*Aug 19 19:59: IP SLA Monitor MPLSLM(1):Next hop 10.10.10.8 added in AddQ
*Aug 19 19:59: IP SLA Monitor MPLSLM(1):Next hop 10.10.10.8 added in AddQ
*Aug 19 19:59: IP SLA Monitor MPLSLM(1):Adding vrf red into tree entry 10.10.10.8
*Aug 19 19:59: IP SLA Monitor MPLSLM(1):Adding Probe 100005
*Aug 19 19:59: IP SLA Monitor MPLSLM(1):Adding ProbeID 100005 to tree entry 10.10.10.8 (1)
*Aug 19 19:59: IP SLA Monitor MPLSLM(1):Adding vrf blue into tree entry 10.10.10.8
*Aug 19 19:59: IP SLA Monitor MPLSLM(1):Duplicate in AddQ 10.10.10.8
*Aug 19 19:59: IP SLA Monitor MPLSLM(1):Adding vrf green into tree entry 10.10.10.8
*Aug 19 19:59: IP SLA Monitor MPLSLM(1):Duplicate in AddQ 10.10.10.8
*Aug 19 19:59: IP SLA Monitor MPLSLM(1):Added Probe(s) 100005 will be scheduled after 26
secs over schedule period 60
```

Related Commands

Command	Description
debug ip sla error	Enables debugging output of Cisco IOS IP SLAs operation run-time errors.
debug ip sla mpls-lsp-monitor	Enables debugging output for Cisco IOS IP SLAs LSP Health Monitor operations in IP SLAs Engine 2.0.
show ip application	Displays global information about Cisco IOS IP SLAs.

debug ip sla trace twamp

To enable debugging output of Cisco IOS IP Service Level Agreements (SLAs) operation for Two-Way Active Measurement Protocol (TWAMP), use the **debug ip sla trace twamp** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip sla trace twamp{**connection** [**source-ip** *ip-address*] | **control**{**reflector** | **server**} | **session** [**source-ip** *ip-address*]}

no debug ip sla trace twamp{**connection** [**source-ip** *ip-address*] | **control**{**reflector** | **server**} | **session** [**source-ip** *ip-address*]}

Syntax Description

connection	Displays communication messages between an IP SLAs TWAMP client and server.
source-ip *ip-address*	(Optional) Debug IP Performance Metrics (IPPM) TWAMP connections for the specified source. Specify the source using the IP address of the client device.
control	Displays communication messages between the IP SLAs TWAMP server and reflector.
reflector	Displays communication messages sent by an IP SLAs TWAMP reflector to the TWAMP server.
server	Displays communication messages sent by an IP SLAs TWAMP server to the TWAMP reflector.
session	Displays communication messages between an IP SLAs TWAMP sender and reflector.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(52)SE	This command was introduced.

Usage Guidelines

Use the **debug ip sla trace twamp** command to display communication messages between the client and server during a TWAMP session.

Note Use the **debug ip sla error twamp connection** command before using the **debug ip sla trace twamp connection** command because the **debug ip sla error twamp connection** command generates less debugging output.

Related Commands

Command	Description
debug ip sla error twamp	Displays exceptions during communication between the IP SLAs TWAMP client and server.

debug ip slb

To display debugging messages for the Cisco IOS Server Load Balancing (SLB) feature, use the **debug ip slb**command in user EXEC or privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip slb {**all**| **asn [msid]**| **conns** [*acl-number*]| **dfp**| **firewallfarm**| **fragments**| **gtp**| **icmp**| **kal-ap**| **natpool**| **probe**| **reals**| **replication**| **route**| **sessions** [**asn**| **gtp**| **ipmobile**| **radius**]| **sticky gtp imsi**| **vservers**}

no debug ip slb {**all**| **asn [msid]**| **conns** [*acl-number*]| **dfp**| **firewallfarm**| **fragments**| **gtp**| **icmp**| **kal-ap**| **natpool**| **probe**| **reals**| **replication**| **route**| **sessions** [**asn**| **gtp**| **ipmobile**| **radius**]| **sticky gtp imsi**| **vservers**}

Syntax Description

all	Displays all debugging messages for Cisco IOS SLB.
asn	Displays debugging messages related to Access Service Network (ASN) load balancing.
msid	(Optional) Displays debugging messages related to the ASN Mobile Station ID (MSID) sticky database.
conns *acl-number*	Displays debugging messages for all connections being handled by IOS SLB, including Wireless Session Protocol (WSP) events and states. The optional *acl-number* argument references an IP access control list (ACL). This argument limits the information displayed based on the client IP address, real server IP address, or virtual server IP address: • For simple ACLs, IOS SLB checks the client IP address. • For extended ACLs, IOS SLB checks the client real and virtual IP addresses. For more information about ACLs, refer to the “Configuring IP Services” chapter of the *Cisco IOS IP Configuration Guide* , Release 12.2.
dfp	Displays debugging messages for Dynamic Feedback Protocol (DFP). • To display debugging messages for the DFP agent subsystem, use the **debug ip dfp agent** command. • To display debugging messages for the general packet radio service (GPRS) DFP weight calculation, use the **debug gprs dfp**command.

firewallfarm	Displays debugging messages related to firewall load balancing.
fragments	Displays debugging messages related to the IOS SLB fragment database.
gtp	Displays all GPRS Tunneling Protocol (GTP)-related packet handler, gateway GPRS support node (GGSN), serving GPRS support node (SGSN), and Network Service Access Point Identifier (NSAPI) debugging messages for IOS SLB.
icmp	Displays all Internet Control Message Protocol debugging messages for IOS SLB.
kal-ap	Displays all KeepAlive Application Protocol (KAL-AP) debugging messages for IOS SLB.
natpool	Displays debugging messages related to the IOS SLB client Network Address Translation (NAT) pool.
probe	Displays debugging messages related to probes.
reals	Displays debugging messages for all real servers defined to IOS SLB.
replication	Displays debugging messages related to IOS SLB stateful backup virtual server.
route	Displays debugging messages for all routing handled by the IOS SLB RADIUS framed-IP sticky database.
sessions [**asn**| **gtp** | **ipmobile** | **radius**	Displays debugging messages for all sessions being handled by IOS SLB. • The optional **asn**keyword enables users to limit the information displayed to only ASN sessions. • The optional **gtp** keyword enables users to limit the information displayed to only GTP sessions. • The optional **ipmobile**keyword enables users to limit the information displayed to only Mobile IP sessions. • The optional **radius** keyword enables users to limit the information displayed to only RADIUS sessions.
sticky gtp imsi	Displays all debugging messages related to the IOS SLB GTP International Mobile Subscriber ID (IMSI) sticky database.

vservers	Displays debugging messages for all virtual servers defined to IOS SLB.

Command Modes

User EXEC or privileged EXEC (#)

Command History

Release	Modification
12.0(7)XE	This command was introduced.
12.1(5)T	This command was integrated into Cisco IOS Release 12.1(5)T.
12.2	This command was integrated into Cisco IOS Release 12.2.
12.1(2)E	The **natpool** and **replication** keywords were added.
12.1(3a)E	The **firewallfarm** keyword was added.
12.1(7)E	The **vservers** keyword was added.
12.1(9)E	The **sessions** keyword was added.
12.1(11b)E	The **route** keyword, the *acl-number* argument, and the **radius** option on the **sessions** keyword were added.
12.2(14)S	This command was integrated into Cisco IOS Release 12.2(14)S.
12.1(13)E3	The **gtp** keyword and the **gtp** option on the **sessions** keyword were added.
12.2(14)ZA2	The **ipmobile** keyword was added.
12.2(18)SXE	The **sticky gtp imsi**keywords were added.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.
12.2(33)SRC	The **kal-ap**keyword was added.
12.2(33)SRC1	The **asn**keyword and the **asn**option on the **sessions** keyword were added.
12.2(33)SRE	The **msid**option on the **asn** keyword was added.

Usage Guidelines

This command displays debugging messages for IOS SLB.

See the following caution before using debug commands:

Caution Because debugging output is assigned high priority in the CPU process, it can render the system unusable. For this reason, use debug commands only to troubleshoot specific problems or during troubleshooting sessions with Cisco technical support staff. Moreover, it is best to use debug commands during periods of lower network flows and fewer users. Debugging during these periods reduces the effect these commands have on other users on the system.

Examples

The following example configures a debugging session to check all IP IOS SLB parameters:

```
Router# debug ip slb all
SLB All debugging is on
Router#
```

The following example stops all debugging:

```
Router# no debug all
All possible debugging has been turned off
Router#
```

The following example configures debugging to check IP IOS SLB replication used with stateful backup and displays the output from the send or transmit virtual server:

```
Router# debug ip slb replication
*Mar  2 08:02:38.019:  SLB Replicate: (send) update vs: VS1 update_count 42
```

The following example shows Cisco IOS SLB DFP debug output:

```
Router# debug ip slb dfp
SLB DFP debugging is on
router#
022048 SLB DFP Queue to main queue - type 2 for Agent 161.44.2.3458229
022048 SLB DFP              select_rc = -1   readset = 0
022048 SLB DFP       Sleeping...
022049 SLB DFP              readset = 0
022049 SLB DFP              select_rc = -1   readset = 0
022049 SLB DFP Processing Q event for Agent 161.44.2.3458229 - OPEN
022049 SLB DFP Queue to conn_proc_q - type 2 for Agent 161.44.2.3458229
022049 SLB DFP              readset = 0
022049 SLB DFP Set SLB_DFP_SIDE_QUEUE
022049 SLB DFP Processing Conn Q event for Agent 161.44.2.3458229 - OPEN
022049 SLB DFP Open to Agent 161.44.2.3458229 succeeded, socket = 0
022049 SLB DFP Agent 161.44.2.3458229 start connect
022049 SLB DFP Connect to Agent 161.44.2.3458229 successful - socket 0
022049 SLB DFP Queue to main queue - type 6 for Agent 161.44.2.3458229
022049 SLB DFP Processing Conn Q unknown MAJOR 80
022049 SLB DFP Reset SLB_DFP_SIDE_QUEUE
022049 SLB DFP              select_rc = -1   readset = 0
022049 SLB DFP       Sleeping...
022050 SLB DFP              readset = 1
022050 SLB DFP              select_rc = 1   readset = 1
022050 SLB DFP Agent 161.44.2.3458229 fd = 0 readset = 1
022050 SLB DFP Message length 44 from Agent 161.44.2.3458229
022050 SLB DFP Agent 161.44.2.3458229 setting Host 17.17.17.17, Bind ID 1 Weight 1
022050 SLB DFP Agent 161.44.2.3458229 setting Host 34.34.34.34, Bind ID 2 Weight 2
022050 SLB DFP Agent 161.44.2.3458229 setting Host 51.51.51.51, Bind ID 3 Weight 3
022050 SLB DFP Processing Q event for Agent 161.44.2.3458229 - WAKEUP
022050 SLB DFP              readset = 1
022050 SLB DFP              select_rc = 1   readset = 1
022050 SLB DFP Agent 161.44.2.3458229 fd = 0 readset = 1
022050 SLB DFP Message length 64 from Agent 161.44.2.3458229
022050 SLB DFP Agent 161.44.2.3458229 setting Host 17.17.17.17, Bind ID 1 Weight 1
022050 SLB DFP Agent 161.44.2.3458229 setting Host 68.68.68.68, Bind ID 4 Weight 4
022050 SLB DFP Agent 161.44.2.3458229 setting Host 85.85.85.85, Bind ID 5 Weight 5
022050 SLB DFP Agent 161.44.2.3458229 setting Host 17.17.17.17, Bind ID 111 Weight 111
022050 SLB DFP              readset = 1
```

```
022115 SLB DFP Queue to main queue - type 5 for Agent 161.44.2.3458229
022115 SLB DFP              select_rc = -1   readset = 0
022115 SLB DFP       Sleeping...
022116 SLB DFP              readset = 1
022116 SLB DFP              select_rc = -1   readset = 0
022116 SLB DFP Processing Q event for Agent 161.44.2.3458229 - DELETE
022116 SLB DFP Queue to conn_proc_q - type 5 for Agent 161.44.2.3458229
022116 SLB DFP              readset = 1
022116 SLB DFP Set SLB_DFP_SIDE_QUEUE
022116 SLB DFP Processing Conn Q event for Agent 161.44.2.3458229 - DELETE
022116 SLB DFP Connection to Agent 161.44.2.3458229 closed
022116 SLB DFP Agent 161.44.2.3458229 deleted
022116 SLB DFP Processing Conn Q unknown MAJOR 80
022116 SLB DFP Reset SLB_DFP_SIDE_QUEUE
022116 SLB DFP Set SLB_DFP_SIDE_QUEUE
022116 SLB DFP Reset SLB_DFP_SIDE_QUEUE
```

debug ip snat

To display information about IP packets translated by the IP stateful network address translation (SNAT) feature, use the **debug ip snat** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip snat [detailed]

no debug ip snat [detailed]

Syntax Description

detailed	(Optional) Displays debug information in a detailed format.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(13)T	This command was introduced.

Usage Guidelines

The **SNAT** feature allows two or more network address translators to function as a translation group. One member of the translation group handles traffic requiring translation of IP address information. It informs the backup translator of active flows as they occur. The backup translator can then use information from the active translator to prepare duplicate translation table entries enabling the backup translator to become the active translator in the event of a critical failure. Traffic continues to flow without interruption because the same network address translations are used and the state of those translations has been previously defined.

Caution Because the **debug ip snat** command generates a significant amount of output, use it only when traffic on the IP network is low, so other activity on the system is not adversely affected.

Examples

The following is sample output from the **debug ip snat** command:

```
Router# debug ip snat detailed
2w6d:SNAT:Establish TCP peers for PRIMARY
2w6d:SNAT (Send):Enqueuing SYNC Message for Router-Id 100
2w6d:SNAT(write2net):192.168.123.2 <---> 192.168.123.3  send message
2w6d:SNAT(write2net):ver 2, id 100, opcode 1, len 68
2w6d:SNAT (Send):Enqueuing DUMP-REQUEST Message for Router-Id 100
2w6d:SNAT(write2net):192.168.123.2 <---> 192.168.123.3  send message
2w6d:SNAT(write2net):ver 2, id 100, opcode 6, len 68
```

```
2w6d:SNAT (readfromnet):Enqueuing SYNC Message msg to readQ
2w6d:SNAT (Receive):Processed SYNC Message from Router-Id:0 for Router-Id:200's entry/entries
2w6d:SNAT (readfromnet):Enqueuing DUMP-REQUEST Message msg to readQ
try/entries
2w6d:SNAT(sense):Send SYNC message
2w6d:SNAT (Send):Enqueuing SYNC Message for Router-Id 100
2w6d:SNAT(write2net):192.168.123.2 <---> 192.168.123.3  send message
2w6d:SNAT(write2net):ver 2, id 100, opcode 1, len 68
2w6d:SNAT (readfromnet):Enqueuing SYNC Message msg to readQ
2w6d:SNAT (Receive):Processed SYNC Message from Router-Id:200 for Router-Id:200's
entry/entries
```

The table below describes the significant fields shown in the display.

Table 67: debug ip snat Field Descriptions

Field	Description
SNAT:	Indicates that the packet is being translated by the SNAT feature.
DUMP-REQUEST Message	Requests for entries after the SNAT router is active.

debug ip socket

To display all state change information for all sockets, use the **debug ip socket** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip socket

no debug ip socket

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

Use this command to collect information on the socket interface. To get more complete information on a socket/TCP port pair, use this command in conjunction with the **debug ip tcp transactions** command.

Because the socket debugging information is state-change oriented, you will not see the debugging message on a per-packet basis. However, if the connections normally have very short lives (few packet exchanges during the life cycle of a connection), then socket debugging could become expensive because of the state changes involved during connection setup and teardown.

Examples

The following is sample output from the **debug ip socket** output from a server process:

```
Router# debug ip socket
Added socket 0x60B86228 to process 40
SOCKET: set TCP property TCP_PID, socket 0x60B86228, TCB 0x60B85E38
Accepted new socket fd 1, TCB 0x60B85E38
Added socket 0x60B86798 to process 40
SOCKET: set TCP property TCP_PID, socket 0x60B86798, TCB 0x60B877C0
SOCKET: set TCP property TCP_BIT_NOTIFY, socket 0x60B86798, TCB 0x60B877C0
SOCKET: created new socket to TCP, fd 2, TCB 0x60B877C0
SOCKET: bound socket fd 2 to TCB 0x60B877C0
SOCKET: set TCP property TCP_WINDOW_SIZE, socket 0x60B86798, TCB 0x60B877C0
SOCKET: listen on socket fd 2, TCB 0x60B877C0
SOCKET: closing socket 0x60B86228, TCB 0x60B85E38
SOCKET: socket event process: socket 0x60B86228, TCB new state --> FINWAIT1
socket state: SS_ISCONNECTED SS_CANTSENDMORE SS_ISDISCONNECTING
SOCKET: Removed socket 0x60B86228 from process 40 socket list
```

The following is sample output from the **debug ip socket** command from a client process:

```
Router# debug ip socket
Added socket 0x60B70220 to process 2
SOCKET: set TCP property TCP_PID, socket 0x60B70220, TCB 0x60B6CFDC
SOCKET: set TCP property TCP_BIT_NOTIFY, socket 0x60B70220, TCB 0x60B6CFDC
SOCKET: created new socket to TCP, fd 0, TCB 0x60B6CFDC
SOCKET: socket event process: socket 0x60B70220, TCB new state --> SYNSENT
socket state: SS_ISCONNECTING
SOCKET: socket event process: socket 0x60B70220, TCB new state --> ESTAB
socket state: SS_ISCONNECTING
SOCKET: closing socket 0x60B70220, TCB 0x60B6CFDC
SOCKET: socket event process: socket 0x60B70220, TCB new state --> FINWAIT1
socket state: SS_ISCONNECTED SS_CANTSENDMORE SS_ISDISCONNECTING
SOCKET: Removed socket 0x60B70220 from process 2 socket list
```

The table below describes the significant fields shown in the display.

Table 68: debug ip socket Field Descriptions

Field	Description
Added socket 0x60B86228 process 40	New socket is opened for process 40.
SOCKET	Indicates that this is a SOCKET transaction.
set TCP property TCP_PID	Sets the process ID to the TCP associated with the socket.
socket 0x60B86228, TCB 0x60B85E38	Address for the socket/TCP pair.
set TCP property TCP_BIT_NOTIFY	Sets the method for how the socket wants to be notified for an event.
created new socket to TCP, fd 2	Opened a new socket referenced by file descriptor 2 to TCP.
bound socket fd 2 to TCB	Bound the socket referenced by file descriptor 2 to TCP.
listen on socket fd 2	Indicates which file descriptor the application is listening to.
closing socket	Indicates that the socket is being closed.
socket event process	Processed a state change event occurred in the transport layer.
TCB new state --> FINWAIT1	TCP state machine changed to FINWAIT1. (See the **debug ip tcp transaction** command for more information on TCP state machines.)

Field	Description
socket state: SS_ISCONNECTED SS_CANTSENDMORE SS_ISDISCONNECTING	New SOCKET state flags after the transport event processing. This socket is still connected, but disconnecting is in progress, and it will not send more data to peer. Possible SOCKET state flags follow: • SS_NOFDREF No file descriptor reference for this socket. • SS_ISCONNECTING Socket connecting is in progress. • SS_ISBOUND Socket is bound to TCP. • SS_ISCONNECTED Socket is connected to peer. • SS_ISDISCONNECTING Socket disconnecting is in progress. • SS_CANTSENDMORE Can't send more data to peer. • SS_CANTRCVMORE Can't receive more data from peer. • SS_ISDISCONNECTED Socket is disconnected. Connection is fully closed.
Removed socket 0x60B86228 from process 40 socket list	Connection is closed, and the socket is removed from the process socket list.

Related Commands

Command	Description
debug ip tcp transactions	Displays information on significant TCP transactions such as state changes, retransmissions, and duplicate packets.

debug ip ssh

To display debugging messages for Secure Shell (SSH), use the **debug ip ssh** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip ssh [**detail**| **packet**]

no debug ip ssh

Syntax Description

detail	(Optional) Specifies SSH protocol, channel requests and information state changes.
packet	(Optional) Specifies information regarding the SSH packet.

Command Default

Debugging for SSH is not enabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.0(5)S	This command was introduced.
12.1(1)T	This command was integrated into Cisco IOS Release 12.1T.
12.4(20)T	The detail and packet keywords were added.
Cisco IOS XE Release 2.4	This command was implemented on the Cisco ASR 1000 series routers.

Usage Guidelines

Use the **debug ip ssh** command to ensure normal operation of the SSH server.

Examples

The following example shows the SSH debugging output:

```
Router# debug ip ssh
00:53:46: SSH0: starting SSH control process
00:53:46: SSH0: Exchanging versions - SSH-1.5-Cisco-1.25
00:53:46: SSH0: client version is - SSH-1.5-1.2.25
00:53:46: SSH0: SSH_SMSG_PUBLIC_KEY message sent
00:53:46: SSH0: SSH_CMSG_SESSION_KEY message received
00:53:47: SSH0: keys exchanged and encryption on
00:53:47: SSH0: authentication request for userid guest
00:53:47: SSH0: authentication successful for jcisco
00:53:47: SSH0: starting exec shell
```

The following example shows the SSH detail output:

```
Router# debug ip ssh detail
00:04:22: SSH0: starting SSH control process
00:04:22: SSH0: sent protocol version id SSH-1.99-Cisco-1.25
00:04:22: SSH0: protocol version id is - SSH-1.99-Cisco-1.25
00:04:22: SSH2 0: SSH2_MSG_KEXINIT sent
00:04:22: SSH2 0: SSH2_MSG_KEXINIT received
00:04:22: SSH2:kex: client->server enc:aes128-cbc mac:hmac-sha1
00:04:22: SSH2:kex: server->client enc:aes128-cbc mac:hmac-sha1
00:04:22: SSH2 0: expecting SSH2_MSG_KEXDH_INIT
00:04:22: SSH2 0: SSH2_MSG_KEXDH_INIT received
00:04:22: SSH2: kex_derive_keys complete
00:04:22: SSH2 0: SSH2_MSG_NEWKEYS sent
00:04:22: SSH2 0: waiting for SSH2_MSG_NEWKEYS
00:04:22: SSH2 0: SSH2_MSG_NEWKEYS received
00:04:24: SSH2 0: authentication successful for lab
00:04:24: SSH2 0: channel open request
00:04:24: SSH2 0: pty-req request
00:04:24: SSH2 0: setting TTY - requested: height 24, width 80; set: height 24, width 80
00:04:24: SSH2 0: shell request
00:04:24: SSH2 0: shell message received
00:04:24: SSH2 0: starting shell for vty
00:04:38: SSH0: Session terminated normally
```

The following example shows the SSH packet output:

```
Router# debug ip ssh packet
00:05:43: SSH2 0: send:packet of  length 280 (length also includes padlen of 4)
00:05:43: SSH2 0: ssh_receive: 64 bytes received
00:05:43: SSH2 0: input: total packet length of 280 bytes
00:05:43: SSH2 0: partial packet length(block size)8 bytes,needed 272 bytes, maclen 0
00:05:43: SSH2 0: ssh_receive: 64 bytes received
00:05:43: SSH2 0: partial packet length(block size)8 bytes,needed 272 bytes, maclen 0
00:05:43: SSH2 0: ssh_receive: 64 bytes received
00:05:43: SSH2 0: partial packet length(block size)8 bytes,needed 272 bytes, maclen 0
00:05:43: SSH2 0: ssh_receive: 64 bytes received
00:05:43: SSH2 0: partial packet length(block size)8 bytes,needed 272 bytes, maclen 0
00:05:43: SSH2 0: ssh_receive: 24 bytes received
00:05:43: SSH2 0: partial packet length(block size)8 bytes,needed 272 bytes, maclen 0
00:05:43: SSH2 0: input: padlength 4 bytes
00:05:43: SSH2 0: ssh_receive: 64 bytes received
00:05:43: SSH2 0: input: total packet length of 144 bytes
00:05:43: SSH2 0: partial packet length(block size)8 bytes,needed 136 bytes, maclen 0
00:05:43: SSH2 0: ssh_receive: 64 bytes received
00:05:43: SSH2 0: partial packet length(block size)8 bytes,needed 136 bytes, maclen 0
00:05:43: SSH2 0: ssh_receive: 16 bytes received
00:05:43: SSH2 0: partial packet length(block size)8 bytes,needed 136 bytes, maclen 0
00:05:43: SSH2 0: input: padlength 6 bytes
00:05:43: SSH2 0: signature length 143
00:05:43: SSH2 0: send:packet of  length 448 (length also includes padlen of 7)
00:05:43: SSH2 0: send:packet of  length 16 (length also includes padlen of 10)
00:05:43: SSH2 0: newkeys: mode 1
00:05:43: SSH2 0: ssh_receive: 16 bytes received
00:05:43: SSH2 0: input: total packet length of 16 bytes
00:05:43: SSH2 0: partial packet length(block size)8 bytes,needed 8 bytes, maclen 0
00:05:43: SSH2 0: input: padlength 10 bytes
00:05:43: SSH2 0: newkeys: mode 0
00:05:43: SSH2 0: ssh_receive: 52 bytes received
00:05:43: SSH2 0: input: total packet length of 32 bytes
00:05:43: SSH2 0: partial packet length(block size)16 bytes,needed 16 bytes, maclen 20
00:05:43: SSH2 0: MAC compared for #3 :ok
```

debug ip subscriber

To enable Intelligent Services Gateway (ISG) IP subscriber session debugging, use the **debug ip subscriber** command in privileged EXEC mode. To disable debugging, use the **no** form of this command.

debug ip subscriber {**all**| **error**| **event**| **fsm**| **packet**}

no debug ip subscriber {**all**| **error**| **event**| **fsm**| **packet**}

Syntax Description

all	Displays all debugging messages related to IP subscriber sessions.
error	Displays debugging messages about IP subscriber session errors.
event	Displays debugging messages about IP subscriber session events.
fsm	Displays debugging messages related to session state changes for IP subscriber sessions.
packet	Displays debugging messages related to IP subscriber session packets.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(31)SB2	This command was introduced.
12.2(33)SRC	This command was integrated into Cisco IOS Release 12.2(33)SRC.
Cisco IOS XE Release 2.2	This command was integrated into Cisco IOS XE Release 2.2.

Examples

The following example show sample output for the **debug ip subscriber** command:

```
Router# debug ip subscriber packet
Packet debugs:
1d07h: IPSUB_DP: [Et0/0:I:CEF:0000.0000.0002] Rx driver forwarded packet via les, return
code = 0
1d07h: IPSUB_DP: [Et0/0:I:PROC:0000.0000.0002] Packet classified, results = 0x18
1d07h: IPSUB_DP: [ms1:I:PROC:0000.0000.0002] Rx driver forwarded the packet
1d07h: IPSUB_DP: [ms1:I:PROC:0000.0000.0002] Packet classified, results = 0x42
1d07h: IPSUB_DP: [ms1:O:PROC:RED:50.0.0.3] Packet classified, results = 0x14
Router#
1d07h: IPSUB_DP: [ms1:O:PROC:RED:50.0.0.3] Subscriber features executed, return code = 0
```

```
1d07h: IPSUB_DP: [ms1:O:PROC:RED:50.0.0.3] Tx driver forwarding the packet
1d07h: IPSUB_DP: [Et0/0:O:PROC:RED:50.0.0.3] Packet classified, results = 0x14
```

Related Commands

Command	Description
show ip subscriber	Displays information about ISG IP subscriber sessions.

debug ip subscriber redundancy

To enable Intelligent Service Gateway (ISG) IP subscriber session debugging on a Cisco 7600 router, use the **debug ip subscriber** command in privileged EXEC mode. To disable debugging, use the **no** form of this command.

debug ip subscriber redundancy

no debug ip subscriber redundancy

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(33)SRC	This command was introduced.

Examples

The following example shows that the **debug ip subscriber redundancy**command is turned on:

```
Router# debug ip subscriber redundancy
 IP subscriber redundancy debugging is on.
```

Related Commands

Command	Description
clear ip subscriber interface	Disconnects and removes all ISG IP subscriber sessions associated with a specific interface on a Cisco 7600 router.
clear ip subscriber slot	Disconnects and removes all ISG IP subscriber sessions associated with a specific hardware slot on a Cisco 7600 router.
show ip subscriber interface	Displays information about an ISG IP subscriber interface on a Cisco 7600 router.
show ip subscriber redundancy	Displays information about ISG IP subscriber sessions on a Cisco 7600 router.
show debugging	Displays information about the types of debugging that are enabled for your router.

debug ip tcp congestion

To display information about TCP congestion events, use the **debug ip tcp congestion** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip tcp congestion

no debug ip tcp congestion

Syntax Description

This command has no arguments or keywords.

Command Default

Information from the New Reno congestion control algorithm is displayed.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
15.1(2)T	This command was introduced.

Usage Guidelines

The **debug ip tcp congestion** command can be used to debug a performance problem on a TCP/IP network that you have isolated above the data-link layer. It also displays information related to variation in TCP's send window, congestion window, and congestion threshold window.

Examples

The following is sample output from the **debug ip tcp congestion** command:

```
Router# debug ip tcp congestion

*May 20 22:49:49.091: Setting New Reno as congestion control algorithm
*May 22 05:21:47.281: Advance cwnd by 12
*May 22 05:21:47.281: TCP85FD0C10: sndcwnd: 1472
*May 22 05:21:47.285: Advance cwnd by 3
*May 22 05:21:47.285: TCP85FD0C10: sndcwnd: 1475
*May 22 05:21:47.285: Advance cwnd by 3
*May 22 05:21:47.285: TCP85FD0C10: sndcwnd: 1478
*May 22 05:21:47.285: Advance cwnd by 9
*May 22 05:21:47.285: TCP85FD0C10: sndcwnd: 1487
.
.
.
*May 20 22:50:32.559: [New Reno] sndcwnd: 8388480 ssthresh: 65535 snd_mark: 232322
*May 20 22:50:32.559: 10.168.10.10:42416 <---> 10.168.30.11:49100 congestion window changes
*May 20 22:50:32.559: cwnd from 8388480 to 2514841, ssthresh from 65535 to 2514841
```

For IOS TCP, New Reno is the default congestion control algorithm. However, an application can also use Binary Increase Congestion Control (BIC) as the congestion algorithm. The following is sample output from the **debug ip tcp congestion** command using the BIC congestion algorithm:

```
Router# debug ip tcp congestion

*May 22 05:21:42.281: Setting BIC as congestion control algorithm
```

```
*May 22 05:21:47.281: Advance cwnd by 12
*May 22 05:21:47.281: TCP85FD0C10: sndcwnd: 1472
*May 22 05:21:47.285: Advance cwnd by 3
*May 22 05:21:47.285: TCP85FD0C10: sndcwnd: 1475
*May 22 05:21:47.285: Advance cwnd by 3
*May 22 05:21:47.285: TCP85FD0C10: sndcwnd: 1478
*May 22 05:21:47.285: Advance cwnd by 9
*May 22 05:21:47.285: TCP85FD0C10: sndcwnd: 1487
.
.
.
.
.
.
*May 20 22:50:32.559: [BIC] sndcwnd: 8388480 ssthresh: 65535 bic_last_max_cwnd: 0 last_cwnd:
 8388480
*May 20 22:50:32.559: 10.168.10.10:42416 <---> 10.168.30.11:49100 congestion window changes
*May 20 22:50:32.559: cwnd from 8388480 to 2514841, ssthresh from 65535 to 2514841
*May 20 22:50:32.559: bic_last_max_cwnd changes from 0 to 8388480
```

The table below describes the significant fields shown in the display.

Table 69: debug ip tcp congestion Field Descriptions

Field	Description
Setting New Reno as congestion control algorithm	TCP is using New Reno as the congestion control algorithm.
TCP85FD0C10	TCP's control block identifier.
Advance cwnd	Increase in TCP’s congestion window.
sndcwnd	TCP's send congestion window.
[New Reno]	Values reflected are those of TCP’s New Reno congestion control.
ssthresh:	TCP’s slow start threshold.
snd_mark	New value of one of New Reno’s parameters.
10.168.10.10:42416:	Local address and port number for the TCP connection.
10.168.30.11.49100:	Foreign address and port number for the TCP connection.
congestion window changes	Change in TCP’s send congestion window.

Related Commands

Command	Description
ip tcp window-size	Alters the TCP window size.

debug ip tcp driver

To display information on TCP driver events; for example, connections opening or closing, or packets being dropped because of full queues, use the **debug ip tcp driver** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip tcp driver

no debug ip tcp driver

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

The TCP driver is the process that the router software uses to send packet data over a TCP connection. Remote source-route bridging (RSRB), serial tunneling (STUN), and X.25 switching currently use the TCP driver.

Using the **debug ip tcp driver** command together with the **debug ip tcp driver-pak**command provides the most verbose debugging output concerning TCP driver activity.

Examples

The following is sample output from the **debug ip tcp driver** command:

```
Router# debug ip tcp driver
TCPDRV359CD8: Active open 172.21.80.26:0 --> 172.21.80.25:1996 OK, lport 36628
TCPDRV359CD8: enable tcp timeouts
TCPDRV359CD8: 172.21.80.26:36628 --> 172.21.80.25:1996 Abort
TCPDRV359CD8: 172.21.80.26:36628 --> 172.21.80.25:1996 DoClose tcp abort
```

The table below describes the significant fields shown in the display.

Table 70: debug ip tcp driver Field Descriptions

Field	Description
TCPDRV359CD8:	Unique identifier for this instance of TCP driver activity.
Active open 172.21.80.26	Indication that the router at IP address 172.21.80.26 has initiated a connection to another router.
:0	TCP port number the initiator of the connection uses to indicate that any port number can be used to set up a connection.
--> 172.21.80.25	IP address of the remote router to which the connection has been initiated.

Field	Description
:1996	TCP port number that the initiator of the connection is requesting that the remote router use for the connection. (1996 is a private TCP port number reserved in this implementation for RSRB.)
OK,	Indication that the connection has been established. If the connection has not been established, this field and the following field do not appear in this line of output.
lport 36628	TCP port number that has actually been assigned for the initiator to use for this connection.

The following line indicates that the TCP driver user (RSRB, in this case) will allow TCP to drop the connection if excessive retransmissions occur:

```
TCPDRV359CD8: enable tcp timeouts
```

The following line indicates that the TCP driver user (in this case, RSRB) at IP address 172.21.80.26 (and using TCP port number 36628) is requesting that the connection to IP address 172.21.80.25 using TCP port number 1996 be aborted:

```
TCPDRV359CD8: 172.21.80.26:36628 --> 172.21.80.25:1996 Abort
```

The following line indicates that this connection was in fact closed because of an abnormal termination:

```
TCPDRV359CD8: 172.21.80.26:36628 --> 172.21.80.25:1996 DoClose tcp abort
```

debug ip tcp driver-pak

To display information on every operation that the TCP driver performs, use the **debug ip tcp driver-pak** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip tcp driver-pak

no debug ip tcp driver-pak

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

This command turns on a verbose debugging by logging at least one debugging message for every packet sent or received on the TCP driver connection.

The TCP driver is the process that the router software uses to send packet data over a TCP connection. Remote source-rate bridging (RSRB), serial tunneling (STUN), and X.25 switching currently use the TCP driver.

To observe the context within which certain **debug ip tcp driver-pak** messages occur, turn on this command in conjunction with the **debug ip tcp driver** command.

Caution

Because the **debug ip tcp driver-pak** command generates so many messages, use it only on lightly loaded systems. This command not only places a substantial load on the system processor, it also may change the symptoms of any unexpected behavior that occurs.

Examples

The following is sample output from the **debug ip tcp driver-pak** command:

```
Router# debug ip tcp driver-pak
TCPDRV359CD8: send 2E8CD8 (len 26) queued
TCPDRV359CD8: output pak 2E8CD8 (len 26) (26)
TCPDRV359CD8: readf 42 bytes (Thresh 16)
TCPDRV359CD8: readf 26 bytes (Thresh 16)
TCPDRV359CD8: readf 10 bytes (Thresh 10)
TCPDRV359CD8: send 327E40 (len 4502) queued
TCPDRV359CD8: output pak 327E40 (len 4502) (4502)
```

The table below describes the significant fields shown in the display.

Table 71: debug ip tcp driver-pak Field Descriptions

Field	Description
TCPDRV359CD8	Unique identifier for this instance of TCP driver activity.
send	Indicates that this event involves the TCP driver sending data.

Field	Description
2E8CD8	Address in memory of the data the TCP driver is sending.
(len 26)	Length of the data (in bytes).
queued	Indicates that the TCP driver user process (in this case, RSRB) has transferred the data to the TCP driver to send.

The following line indicates that the TCP driver has sent the data that it had received from the TCP driver user, as shown in the previous line of output. The last field in the line (26) indicates that the 26 bytes of data were sent out as a single unit.

```
TCPDRV359CD8: output pak 2E8CD8 (len 26) (26)
```

The following line indicates that the TCP driver has received 42 bytes of data from the remote IP address. The TCP driver user (in this case, remote source-route bridging) has established an input threshold of 16 bytes for this connection. (The input threshold instructs the TCP driver to transfer data to the TCP driver user only when at least 16 bytes are present.)

```
TCPDRV359CD8: readf 42 bytes (Thresh 16)
```

debug ip tcp ecn

To turn on debugging of the TCP Explicit Congestion Notification (ECN) capability, use the **debug ip tcp ecn**command in privileged EXEC mode. To turn off the debugging, use the **no** form of this command.

debug ip tcp ecn

no debug ip tcp ecn

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.3(7)T	This command was introduced.
12.2(31)SB2	This command was integrated into Cisco IOS Release 12.2(31)SB2.
Cisco IOS XE Release 2.1	This command was integrated into Cisco IOS XE Release 2.1.

Examples

The following example shows the messages that verify that the end hosts are connected and configured for ECN:

```
Router# debug ip tcp ecn
!
TCP ECN debugging is on
!
Router# telnet 10.1.25.31

Trying 10.1.25.31 ...
!
01:43:19: 10.1.25.35:11000 <---> 10.1.25.31:23   out ECN-setup SYN
01:43:21: 10.1.25.35:11000 <---> 10.1.25.31:23   congestion window changes
01:43:21: cwnd from 1460 to 1460, ssthresh from 65535 to 2920
01:43:21: 10.1.25.35:11000 <---> 10.1.25.31:23   in non-ECN-setup SYN-ACK
```

Before a TCP connection can use ECN, a host sends an ECN-setup SYN (synchronization) packet to a remote end that contains an ECE and CWR bit set in the header. This indicates to the remote end that the sending TCP is ECN-capable, rather than an indication of congestion. The remote end sends an ECN-setup SYN-ACK (acknowledgment) packet to the sending host.

In the example above, the “out ECN-setup SYN” text means that a SYN packet with the ECE and CWR bit set was sent to the remote end. The “in non-ECN-setup SYN-ACK” text means that the remote end did not favorably acknowledge the ECN request and that therefore the session is ECN capable.

The following debug output shows that ECN capabilities are enabled at both ends. In response to the ECN-setup SYN, the other end favorably replied with an ECN-setup SYN-ACK message. This connection is now ECN capable for the rest of the session.

```
Router# telnet 10.10.10.10
```

```
Trying 10.10.10.10 ... Open
Password required, but none set
!
1d20h: 10.1.25.34:11003 <---> 10.1.25.35:23   out ECN-setup SYN
1d20h: 10.1.25.34:11003 <---> 10.1.25.35:23   in ECN-setup SYN-ACK
```

Use the **show tcp tcb** command to display the end-host connections.

Related Commands

Command	Description
ip tcp ecn	Enables TCP ECN.
show tcp tcb	Displays the status of local and remote end hosts.

debug ip tcp ha

To display TCP high availability (HA) events or debugging information for TCP stack interactions between the active Route Processor (RP) and the standby RP, use the **debug ip tcp ha** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip tcp ha {**events**| **transactions**} **[detail]**

no debug ip tcp ha {**events**| **transactions**} **[detail]**

Syntax Description

events	Displays TCP HA failures.
transactions	Displays failed TCP stack interactions between the active RP and standby RP.
detail	(Optional) Displays detailed debugging information about successful TCP HA operations and useful informational messages or about successful TCP stack interactions between the active and standby RP.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(28)SB	This command was introduced.
15.0(1)S	This command was integrated into Cisco IOS Release 15.0(1)S.
Cisco IOS XE 3.1S	This command was integrated into Cisco IOS XE Release 3.1S.

Usage Guidelines

The **debug ip tcp ha** command is used to display TCP stateful switchover (SSO) events or debugging information for TCP stack interactions between the active RP and the standby RP. This is command is useful for troubleshooting SSO-aware TCP connections.

Use the **debug ip tcp ha** command with the **transactions** keyword to display failed TCP stack interactions between the active RP and standby RP. This form of the command displays failed TCP HA messages, RF redundancy-related client-application transactions, IPC client-application transactions, and In-Service Software Upgrade (ISSU) transactions.

Use the **debug ip tcp ha** command with the **transactions** and **detail** keywords to display successful TCP stack interactions between the active and standby RP. This form of the command displays successful TCP HA messages, RF redundancy-related client-application transactions, IPC client-application transactions, and ISSU transactions.

Use the **debug ip tcp ha** command with the **events** keyword to display TCP HA failures. This form of the command displays TCP HA failed encode or decode messages, system resources failures (such as memory allocation failures in the context of TCP HA), failed state changes, and failures that occur when SSO is enabled or disabled.

Use the **debug ip tcp ha** command with the **events** and **detail** keywords to display successful TCP HA operations and useful informational messages. This form of the command displays successful TCP encode or decode messages, state changes, and operations that occur when SSO is enabled or disabled.

Examples

The following is sample output from the **debug ip tcp ha** command with the **transactions** and **detail** keywords. The following output shows packet flow from the active to the standby RP for an established TCP SSO connection:

```
*Feb 19 23:28:23.324: TCPHA: Sending pkt msg, conn_id = 39, seq no = 2727115707
*Feb 19 23:28:23.324: TCPHA: Sending pkt msg, conn_id = 396, seq no = 2959469308
*Feb 19 23:28:23.324: TCPHA: Sending pkt msg, conn_id = 41, seq no = 1270243395
*Feb 19 23:28:23.932: TCPHA: Sending pkt msg, conn_id = 42, seq no = 974255741
*Feb 19 23:28:23.932: TCPHA: Sending pkt msg, conn_id = 475, seq no = 3059612402
*Feb 19 23:28:24.544: TCPHA: Sending dummy pkt to standby; cid=109, size=19

*Feb 19 23:28:42.976: TCPHA: Recd IPC msg len 24, type 3
*Feb 19 23:28:42.976: TCPHA: Recd IPC msg len 24, type 3
*Feb 19 23:28:43.172: TCPHA: Recd IPC msg len 79, type 2
*Feb 19 23:28:43.172: TCPHA: Recd IPC msg len 79, type
```

debug ip tcp intercept

To display TCP intercept statistics, use the **debug ip tcp intercept**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip tcp intercept

no debug ip tcp intercept

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Examples

The following is sample output from the **debug ip tcp intercept** command:

```
Router# debug ip tcp intercept
```

A connection attempt arrives:

```
INTERCEPT: new connection (172.19.160.17:61774) => (10.1.1.30:23)
INTERCEPT: 172.19.160.17:61774 <- ACK+SYN (10.1.1.30:61774)
```

A second connection attempt arrives:

```
INTERCEPT: new connection (172.19.160.17:62030) => (10.1.1.30:23)
INTERCEPT: 172.19.160.17:62030 <- ACK+SYN (10.1.1.30:62030)
```

The router resends to both apparent clients:

```
INTERCEPT: retransmit 2 (172.19.160.17:61774) <- (10.1.1.30:23) SYNRCVD
INTERCEPT: retransmit 2 (172.19.160.17:62030) <- (10.1.1.30:23) SYNRCVD
```

A third connection attempt arrives:

```
INTERCEPT: new connection (171.69.232.23:1048) => (10.1.1.30:23)
INTERCEPT: 171.69.232.23:1048 <- ACK+SYN (10.1.1.30:1048)
```

The router sends more retransmissions trying to establish connections with the apparent clients:

```
INTERCEPT: retransmit 4 (172.19.160.17:61774) <- (10.1.1.30:23) SYNRCVD
INTERCEPT: retransmit 4 (172.19.160.17:62030) <- (10.1.1.30:23) SYNRCVD
INTERCEPT: retransmit 2 (171.69.232.23:1048) <- (10.1.1.30:23) SYNRCVD
```

The router establishes the connection with the third client and resends to the server:

```
INTERCEPT: 1st half of connection is established (171.69.232.23:1048) => (10.1.1.30:23)
INTERCEPT: (171.69.232.23:1048) SYN -> 10.1.1.30:23
INTERCEPT: retransmit 2 (171.69.232.23:1048) -> (10.1.1.30:23) SYNSENT
```

The server responds; the connection is established:

```
INTERCEPT: 2nd half of connection established (171.69.232.23:1048) => (10.1.1.30:23)
INTERCEPT: (171.69.232.23:1048) ACK -> 10.1.1.30:23
```

The router resends to the first two apparent clients, times out, and sends resets:

```
INTERCEPT: retransmit 8 (172.19.160.17:61774) <- (10.1.1.30:23) SYNRCVD
INTERCEPT: retransmit 8 (172.19.160.17:62030) <- (10.1.1.30:23) SYNRCVD
INTERCEPT: retransmit 16 (172.19.160.17:61774) <- (10.1.1.30:23) SYNRCVD
INTERCEPT: retransmit 16 (172.19.160.17:62030) <- (10.1.1.30:23) SYNRCVD
INTERCEPT: retransmitting too long (172.19.160.17:61774) => (10.1.1.30:23) SYNRCVD
```

```
INTERCEPT: 172.19.160.17:61774 <- RST (10.1.1.30:23)
INTERCEPT: retransmitting too long (172.19.160.17:62030) => (10.1.1.30:23) SYNRCVD
INTERCEPT: 172.19.160.17:62030 <- RST (10.1.1.30:23)
```

debug ip tcp packet

To enable debug messages for received and sent TCP packets, use the **debug ip tcp packet** command in privileged EXEC mode. To disable TCP packet debug messages, use the **no** form of this command.

debug ip tcp packet [**line-number**| **address ip-address**| {**aux**| **console**| **tty**| **vty**} **line-number**| **in**| **out**| **port port-number**| **slot/port**| **slot/subslot/port**]

no debug ip tcp packet [**line-number**| **address ip-address**| {**aux**| **console**| **tty**| **vty**} **line-number**| **in**| **out**| **port port-number**| **slot/port**| **slot/subslot/port**]

Syntax Description

line-number	(Optional) Line number. Valid range is 0 to 710.
address *ip-address*	(Optional) Specifies the source or destination IP address.
aux *line-number*	(Optional) Specifies the auxiliary line.
console *line-number*	(Optional) Specifies the primary terminal line.
in	(Optional) Specifies the incoming segments.
out	(Optional) Specifies the outgoing segments.
port *port-number*	(Optional) Specifies the source or destination port number.
tty *line-number*	(Optional) Specifies the terminal controller.
vty *line-number*	(Optional) Specifies the virtual terminal.
slot / *port*	(Optional) Specifies the slot and port for modems. The slash mark is required.
slot / *subslot* / *port*	(Optional) Specifies the slot, subslot, and port for modems. The slash mark is required.

Command Default

If no optional arguments or keywords are entered, this command displays all TCP packet debug messages.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
11.1	This command was introduced.

Examples

The following is sample output from the **debug ip tcp packet**command:

```
Router# debug ip tcp packet
tcp0: I LISTEN 172.16.0.0:49620 172.16.0.1:80 seq 2116160325
OPTS 4 SYN WIN 1024
tcp0: O SYNRCVD 172.16.0.34:49620 172.16.0.1:80 seq 3992162775
OPTS 4 ACK 2116160325 SYN WIN 4128
tcp0: I SYNRCVD 172.16.0.34:49620 172.16.0.1:80 seq 2116160326
RST WIN 0
```

Related Commands

Command	Description
debug ip packet detail	Displays general IP debugging information and IP security option security transactions.
debug ip tcp driver	Displays information on TCP driver events; for example, connections opening or closing, or packets being dropped because of full queues.
debug ip tcp transactions	Displays information on significant TCP transactions such as state changes, retransmissions, and duplicate packets.

debug ip tcp transactions

To display information on significant TCP transactions such as state changes, retransmissions, and duplicate packets, use the **debug ip tcp transactions**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip tcp transactions

no debug ip tcp transactions

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
11.0	This command was introduced.
12.3(7)T	The command output was enhanced to account for the following conditions: TCP entering Fast Recovery mode, duplicate acknowledgments being received during Fast Recovery mode, and partial acknowledgments being received.
12.2(31)SB2	This command was integrated into Cisco IOS Release 12.2(31)SB2.

Usage Guidelines

This command is particularly useful for debugging a performance problem on a TCP/IP network that you have isolated above the data-link layer.

The **debug ip tcp transactions** command displays output for packets that the router sends and receives, but does not display output for packets that it forwards.

Examples

The following is sample output from the **debug ip tcp transactions**command:

```
Router# debug ip tcp transactions
TCP: sending SYN, seq 168108, ack 88655553
TCP0: Connection to 10.9.0.13:22530, advertising MSS 966
TCP0: state was LISTEN -> SYNRCVD [23 -> 10.9.0.13(22530)]
TCP0: state was SYNSENT -> SYNRCVD [23 -> 10.9.0.13(22530)]
TCP0: Connection to 10.9.0.13:22530, received MSS 956
TCP0: restart retransmission in 5996
TCP0: state was SYNRCVD -> ESTAB [23 -> 10.9.0.13(22530)]
TCP2: restart retransmission in 10689
TCP2: restart retransmission in 10641
TCP2: restart retransmission in 10633
TCP2: restart retransmission in 13384 -> 10.0.0.13(16151)]
TCP0: restart retransmission in 5996 [23 -> 10.0.0.13(16151)]
```

The following line from the **debug ip tcp transactions** command output shows that TCP has entered Fast Recovery mode:

```
fast re-transmit - sndcwnd - 512, snd_last - 33884268765
```

The following lines from the **debug ip tcp transactions** command output show that a duplicate acknowledgment is received when in Fast Recovery mode (first line) and a partial acknowledgment has been received (second line):

```
TCP0:ignoring second congestion in same window sndcwn - 512, snd_1st - 33884268765
TCP0:partial ACK received sndcwnd:338842495
```

The table below describes the significant fields shown in the display.

Table 72: debug ip tcp transactions Field Descriptions

Field	Description
TCP	Indicates that this is a TCP transaction.
sending SYN	Indicates that a synchronize packet is being sent.
seq 168108	Indicates the sequence number of the data being sent.
ack 88655553	Indicates the sequence number of the data being acknowledged.
TCP0	Indicates the TTY number (0, in this case) with which this TCP connection is associated.
Connection to 10.9.0.13:22530	Indicates the remote address with which a connection has been established.
advertising MSS 966	Indicates the maximum segment size that this side of the TCP connection is offering to the other side.

Field	Description
state was LISTEN -> SYNRCVD	Indicates that the TCP state machine changed state from LISTEN to SYNRCVD. Possible TCP states that can follow are: • CLOSED--Connection closed. • CLOSEWAIT--Received a FIN segment. • CLOSING--Received a FIN/ACK segment. • ESTAB--Connection established. • FINWAIT 1--Sent a FIN segment to start closing the connection. • FINWAIT 2--Waiting for a FIN segment. • LASTACK--Sent a FIN segment in response to a received FIN segment. • LISTEN--Listening for a connection request. • SYNRCVD--Received a SYN segment and responded. • SYNSENT--Sent a SYN segment to start connection negotiation. • TIMEWAIT--Waiting for the network to clear segments for this connection before the network no longer recognizes the connection as valid. This must occur before a new connection can be set up.
[23 -> 10.9.0.13(22530)]	The elements within these brackets are as follows: • The first field (23) indicates the local TCP port. • The second field (10.9.0.13) indicates the destination IP address. • The third field (22530) indicates the destination TCP port.
restart retransmission in 5996	Indicates the number of milliseconds until the next retransmission takes place.
sndcwnd - 512	Indicates the size of the send congestion window.
snd_last - 33884268765	Indicates the size of the last window.

debug ip traffic-export events

To enable debugging messages for exported IP packet events, use the **debug ip traffic-export**command in privileged EXEC mode. To disable debugging messages, use the **no** form of this command.

debug ip traffic-export events

no debug ip traffic-export events

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(4)T	This command was introduced.
12.2(25)S	This command was integrated into Cisco IOS Release 12.2(25)S.

Examples

The following is sample output from the **debug ip traffic-export events**command:

```
Router# debug ip traffic-export events
RITE:exported input packet # 547
RITE:exported input packet # 548
RITE:exported input packet # 549
RITE:exported input packet # 550
RITE:exported input packet # 551
RITE:exported input packet # 552
RITE:exported input packet # 553
RITE:exported input packet # 554
RITE:exported input packet # 555
RITE:exported input packet # 556
RITE:exported input packet # 557
RITE:exported input packet # 558
RITE:exported input packet # 559
RITE:exported input packet # 560
RITE:exported input packet # 561
RITE:exported input packet # 562
```

Related Commands

Command	Description
ip traffic-export profile	Creates or edits an IP traffic export profile and enables the profile on an ingress interface.

debug ip trigger-authentication

To display information related to automated double authentication, use the **debug ip trigger-authentication** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip trigger-authentication [verbose]

no debug ip trigger-authentication [verbose]

Syntax Description

verbose	(Optional) Specifies that the complete debugging output be displayed, including information about packets that are blocked before authentication is complete.

Command Modes

Privileged EXEC

Usage Guidelines

Use this command when troubleshooting automated double authentication.

This command displays information about the remote host table. Whenever entries are added, updated, or removed, a new debugging message is displayed.

What is the remote host table? Whenever a remote user needs to be user-authenticated in the second stage of automated double authentication, the local device sends a User Datagram Protocol (UDP) packet to the host of the remote user. Whenever such a UDP packet is sent, the host IP address of the user is added to a table. If additional UDP packets are sent to the same remote host, a new table entry is not created; instead, the existing entry is updated with a new time stamp. This remote host table contains a cumulative list of host entries; entries are deleted after a timeout period or after you manually clear the table by using the **clear ip trigger-authentication** command.

If you include the **verbose** keyword, the debugging output also includes information about packet activity.

Examples

The following is sample output from the **debug ip trigger-authentication** command. In this example, the local device at 172.21.127.186 sends a UDP packet to the remote host at 172.21.127.114. The UDP packet is sent to request the remote user's username and password (or PIN). (The output says "New entry added.")

After a timeout period, the local device has not received a valid response from the remote host, so the local device sends another UDP packet. (The output says "Time stamp updated.")

Then the remote user is authenticated, and after a length of time (the timeout period) the entry is removed from the remote host table. (The output says "remove obsolete entry.")

```
myfirewall# debug ip trigger-authentication
TRIGGER_AUTH: UDP sent from 172.21.127.186 to 172.21.127.114, qdata=7C2504
              New entry added, timestamp=2940514234
TRIGGER_AUTH: UDP sent from 172.21.127.186 to 172.21.127.114, qdata=7C2504
              Time stamp updated, timestamp=2940514307
TRIGGER_AUTH: remove obsolete entry, remote host=172.21.127.114
```

The following is sample output from the **debug ip trigger-authentication verbose** command. In this example, messages about packet activity are included because of the use of the **verbose** keyword.

You can see many packets that are being blocked at the interface because the user has not yet been double authenticated. These packets will be permitted through the interface only after the user has been double authenticated. (You can see packets being blocked when the output says “packet enqueued” and then “packet ignored.”)

```
TRIGGER_AUTH: packet enqueued, qdata=69FEEC
              remote host=172.21.127.113, local host=172.21.127.186 (if: 0.0.0.0)
TRIGGER_AUTH: UDP sent from 172.21.127.186 to 172.21.127.113, qdata=69FEEC
              Time stamp updated
TRIGGER_AUTH: packet enqueued, qdata=69FEEC
              remote host=172.21.127.113, local host=172.21.127.186 (if: 0.0.0.0)
TRIGGER_AUTH: packet ignored, qdata=69FEEC
TRIGGER_AUTH: packet enqueued, qdata=69FEEC
              remote host=172.21.127.113, local host=172.21.127.186 (if: 0.0.0.0)
TRIGGER_AUTH: packet ignored, qdata=69FEEC
TRIGGER_AUTH: packet enqueued, qdata=69FEEC
              remote host=172.21.127.113, local host=172.21.127.186 (if: 0.0.0.0)
TRIGGER_AUTH: UDP sent from 172.21.127.186 to 172.21.127.113, qdata=69FEEC
              Time stamp updated
TRIGGER_AUTH: packet enqueued, qdata=69FEEC
              remote host=172.21.127.113, local host=172.21.127.186 (if: 0.0.0.0)
TRIGGER_AUTH: packet ignored, qdata=69FEEC
TRIGGER_AUTH: packet enqueued, qdata=69FEEC
              remote host=172.21.127.113, local host=172.21.127.186 (if: 0.0.0.0)
TRIGGER_AUTH: packet ignored, qdata=69FEEC
```

debug ip trm

To enable debug information of the Trend Registration Module (TRM), use the **debug ip trm** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip trm [**detailed**| **timers**]

no debug ip trm [**detailed**| **timers**]

Syntax Description

detailed	(Optional) The system prints detailed information about the TRM. If not specified, the system displays basic status information.
timers	(Optional) The system prints information about timer events on the TRM. If not specified, the system displays basic status information.

Command Default

This command is not enabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.4(15)XZ	This command was introduced.

Usage Guidelines

Use the **debug ip trm** to enable debug information of the TRM, which handles the registration between the system and the Trend Router Provisioning Server (TRPS).

Examples

The following is sample output from the **debug ip trm**command:

```
Router# debug ip trm
TRM: Exceeded retry timeouts. Setting server inactive
```

The following is sample output from the **debug ip trm detailed** command:

```
Router# debug ip trm detailed
TRM: Sending Reg Req to TRPS. Requesting AV Key = No
Modify Trend Global Parameter map
```

The following is sample output from the **debug ip trm timers** command:

```
Router# debug ip trm timers
TRM: Wait timer for active server. Sent Reg request
```

debug ip urd

To display debugging messages for URL Rendezvous Directory (URD) channel subscription report processing, use the **debug ip urd command in privileged EXEC**mode. To disable debugging output, use the **no** form of this command.

debug ip urd [*hostname*| *ip-address*]

no debug ip urd

Syntax Description

hostname	(Optional) The domain Name System (DNS) name.
ip-address	(Optional) The IP address.

Command Default

If no host name or IP address is specified, all URD reports are debugged.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.1(3)T	This command was introduced.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Examples

The following is sample output from the **debug ip urd** command:

```
Router# debug ip urd
13:36:25 pdt:URD:Data intercepted from 171.71.225.103
13:36:25 pdt:URD:Enqueued string:
'/cgi-bin/error.pl?group=232.16.16.16&port=32620&source=171.69.214.1&li'
13:36:25 pdt:URD:Matched token:group
13:36:25 pdt:URD:Parsed value:232.16.16.16
13:36:25 pdt:URD:Creating IGMP source state for group 232.16.16.16
```

debug ip urlfilter

To enable debug information of URL filter subsystems, use the **debug ip urlfilter** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip urlfilter {**function-trace**| **detailed**| **events**}

no debug ip urlfilter {**function-trace**| **detailed**| **events**}

Syntax Description

function-trace	The system displays a sequence of important functions that are called when configuring URL filtering.
detailed	The system displays detailed information about various activities that occur during URL filtering.
events	The system displays various events such as queue event, timer event, and socket event.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(11)YU	This command was introduced.
12.2(15)T	This command was integrated into Cisco IOS Release 12.2(15)T.
12.4(15)XZ	This command was implemented on the Cisco 881 and Cisco 888 platforms.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.

Examples

The following is sample output from the **debug ip urlfilter** command when SmartFilter URL filtering configured:

```
Router# debug ip urlfilter detailed
urlfilter:
  Urlfilter Detailed Debugs debugging is on
Router# show ip urlfilter config

N2H2 URL Filtering is ENABLED
Primary N2H2 server configurations
=========================================
N2H2 server IP address:192.168.1.103
N2H2 server port:4005
N2H2 retransmission time out:6 (in seconds)
```

```
N2H2 number of retransmission:2
Secondary N2H2 servers configurations
============================================
Other configurations
=====================
Allow Mode:OFF
System Alert:ENABLED
Audit Trail:ENABLED
Log message on N2H2 server:DISABLED
Maximum number of cache entries:5
Maximum number of packet buffers:20
Maximum outstanding requests:1000
fw1_4#
1d15h:URLF:got a socket read event...
1d15h:URLF:socket recv failed.
1d15h:URLF:Closing the socket for server (192.168.1.103:4005)
1d15h:%URLF-3-SERVER_DOWN:Connection to the URL filter server 192.168.1.103 is down
1d15h:URLF:Opening a socket for server (192.168.1.103:4005)
1d15h:URLF:socket fd 0
1d15h:%URLF-5-SERVER_UP:Connection to an URL filter server(192.168.1.103) is made, the
router is returning from ALLOW MODE
1d15h:URLF:got cache idle timer event...
1d16h:URLF:got cache absolute timer event...
1d16h:URLF:got cache idle timer event...
1d16h:URLF:creating uis 0x63A95DB4, pending request 1
1d16h:URLF:domain name not found in the exclusive list
1d16h:URLF:got an cbac queue event...
1d16h:URLF:socket send successful...172.17.192.130:8080) -> 192.168.1.103:1052 seq 3344720064
 wnd 24820
1d16h:URLF:holding pak 0x634A8A08 (172.17.192.130:8080) -> 192.168.1.103:1052 seq 3344721524
 wnd 24820
1d16h:URLF:holding pak 0x634A98CC (172.17.192.130:8080) -> 192.168.1.103:1052 seq 3344722984
 wnd 24820
1d16h:URLF:got a socket read event...
1d16h:URLF:socket recv (header) successful.
1d16h:URLF:socket recv (data) successful.
1d16h:URLF:n2h2 lookup code = 1
1d16h:URLF:Site/URL Blocked:sis 0x63675DC4, uis 0x63A95DB4
1d16h:%URLF-4-URL_BLOCKED:Access denied URL 'http://www.example.com/', client
192.168.1.103:1052 server 172.17.192.130:8080
1d16h:URLF:(192.168.1.103:1052) RST -> 172.17.192.130:8080 seq 3361738063 wnd 0
1d16h:URLF:(172.17.192.130:8080) FIN -> 192.168.1.103:1052 seq 3344720064 wnd 0
1d16h:URLF:deleting uis 0x63A95DB4, pending requests 0
1d16h:URLF:got cache idle timer event...
1d16h:URLF:creating uis 0x63A95DB4, pending request 1
1d16h:URLF:domain name not found in the exclusive list
1d16h:URLF:got an cbac queue event...
1d16h:URLF:socket send successful...
1d16h:URLF:holding pak 0x634A812C (172.17.192.130:8080) -> 192.168.1.103:1101 seq 3589711120
 wnd 24820
1d16h:URLF:holding pak 0x634A2E7C (172.17.192.130:8080) -> 192.168.1.103:1101 seq 3589712580
 wnd 24820
1d16h:URLF:holding pak 0x634A3464 (172.17.192.130:8080) -> 192.168.1.103:1101 seq 3589714040
 wnd 24820
1d16h:URLF:got a socket read event...
1d16h:URLF:socket recv (header) successful.
1d16h:URLF:socket recv (data) successful.
1d16h:URLF:n2h2 lookup code = 0
1d16h:%URLF-6-URL_ALLOWED:Access allowed for URL 'http://www.example1.com/', client
192.168.1.103:1101 server 172.17.192.130:8080
1d16h:URLF:Site/URL allowed:sis 0x6367D0C4, uis 0x63A95DB4
1d16h:URLF:releasing pak 0x634A812C:(172.17.192.130:8080) -> 192.168.1.103:1101 seq 3589711120
 wnd 24820
1d16h:URLF:releasing pak 0x634A2E7C:(172.17.192.130:8080) -> 192.168.1.103:1101 seq 3589712580
 wnd 24820
1d16h:URLF:releasing pak 0x634A3464:(172.17.192.130:8080) -> 192.168.1.103:1101 seq 3589714040
 wnd 24820
1d16h:URLF:deleting uis 0x63A95DB4, pending requests 0
1d16h:URLF:got cache idle timer event...
1d16h:URLF:creating uis 0x63A9777C, pending request 1
1d16h:URLF:domain name not found in the exclusive list
1d16h:URLF:got an cbac queue event...
1d16h:URLF:socket send successful...
```

```
1d16h:URLF:got a socket read event...
1d16h:URLF:socket recv (header) successful.
1d16h:URLF:socket recv (data) successful.
1d16h:URLF:n2h2 lookup code = 1
1d16h:URLF:Site/URL Blocked:sis 0x63677ED4, uis 0x63A9777C
1d16h:%URLF-4-URL_BLOCKED:Access denied URL 'http://www.example2.com/', client
192.168.1.103:1123 server 172.17.192.130:8080
1d16h:URLF:(192.168.1.103:1123) RST -> 172.17.192.130:8080 seq 3536466275 wnd 0
1d16h:URLF:(172.17.192.130:8080) FIN -> 192.168.1.103:1123 seq 3618929551 wnd 0
1d16h:URLF:deleting uis 0x63A9777C, pending requests 0
1d16h:URLF:got cache idle timer event...
```

debug ip verify mib

To view debug output that displays the operation of Unicast Reverse Path Forwarding (RPF) MIB objects and the helper software, use the **debug ip verify mib** command in privileged EXEC mode. To disable debugging for Unicast RPF, use the **no** form of this command.

debug ip verify mib

no debug ip verify mib

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging activity for the operation of Unicast RPF MIB objects and helper software does not occur.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(31)SB2	This command was introduced.
12.2(33)SRC	This command was integrated into Cisco IOS Release 12.2(33)SRC.
12.4(20)T	This command was integrated into Cisco IOS Release 12.4(20)T.
12.2(33)SXI2	This command was integrated into Cisco IOS Release 12.2(33)SXI2.

Usage Guidelines

Debug information for the Unicast RPF MIB is collected only when logging is enabled. Unicast RPF messages are stored in the logging buffer, and they are not displayed on the console unless you use the **debug ip verify mib** command.

Examples

The following example shows sample output of the **debug ip verify mib**command:

```
Router> enable
Router# debug ip verify mib
01:29:45: cipUrpfScalar_get, searchType 161
01:29:45: ipurpfmib_get_scalars
01:29:45: cipUrpfScalar_get, searchType 161
01:29:45: cipUrpfScalar_get, searchType 161
01:29:45: ipurpfmib_get_scalars
01:29:45: cipUrpfScalar_get, searchType 161
01:29:45: cipUrpfScalar_get, searchType 161
01:29:45: ipurpfmib_get_scalars
01:29:45: cipUrpfScalar_get, searchType
161ipurpfmib_get_urpf_entryipurpfmib_get_urpf_entryipurpfmib_get_urpf_entryipurpfmib_get_
urpf_entry
01:29:45: cipUrpfIfMonEntry_get, searchType 161
01:29:45: ipurpfmib_get_urpf_ifmon_entry entry: ST 161, if 1, ip 1
01:29:45: cipUrpfIfMonEntry_get, searchType 161
```

```
01:29:45: ipurpfmib_get_urpf_ifmon_entry entry: ST 161, if 1, ip 1
01:29:45: cipUrpfIfMonEntry_get, searchType 161
01:29:45: ipurpfmib_get_urpf_ifmon_entry entry: ST 161, if 1, ip 1
01:29:45: cipUrpfIfMonEntry_get, searchType 161
01:29:45: ipurpfmib_get_urpf_ifmon_entry entry: ST 161, if 1, ip 1
01:29:45: cipUrpfIfMonEntry_get, searchType 161
01:29:45: ipurpfmib_get_urpf_ifmon_entry entry: ST 161, if 1, ip 1
01:29:45: cipUrpfIfMonEntry_get, searchType 161
01:29:45: ipurpfmib_get_urpf_ifmon_entry entry: ST 161, if 1, ip 1
01:29:45: cipUrpfIfMonEntry_get, searchType 161
01:29:45: ipurpfmib_get_urpf_ifmon_entry entry: ST 161, if 1, ip 1
01:29:45: cipUrpfIfMonEntry_get, searchType 161
01:29:45: ipurpfmib_get_urpf_ifmon_entry entry: ST 161, if 1, ip 1
01:29:45: cipUrpfIfMonEntry_get, searchType 161
01:29:45: ipurpfmib_get_urpf_ifmon_entry entry: ST 161, if 1, ip 1
01:29:45: cipUrpfIfMonEntry_get, searchType 161
01:29:45: ipurpfmib_get_urpf_ifmon_entry entry: ST 161, if 1, ip 1
```

Related Commands

Command	Description
show ip interface	Displays the usability status of interfaces configured for IP.

debug ip virtual-reassembly

To enable debugging of the virtual fragment reassembly (VFR) subsystem, use the **debug ip virtual-reassembly** command in privileged EXEC mode. To disable VFR debugging, use the **no** form of this command.

debug ip virtual-reassembly [**list** {**access-list**| **extended-access-list**}]

no debug ip virtual-reassembly [**list** {**access-list**| **extended-access-list**}]

Syntax Description

list	(Optional) Enables VFR conditional debugging.
access-list	Filters the generated list of VFR conditional debugging messages. The valid range is from 1 to 199.
extended-access-list	Filters the generated list of extended VFR conditional debugging messages. The valid range is from 1300 to 2699.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(8)T	This command was introduced.
15.0(1)M	The list keyword was introduced.

Examples

The following sample output from the **debug ip virtual-reassembly** command allows you to monitor datagram fragmentation and reassembly status--such as whether a datagram is incomplete and when fragments (from the datagram) are created (after a datagram is determined to be complete).

```
Router# debug ip virtual-reassembly
00:17:35: IP_VFR: fragment (sa:13.0.0.2, da:17.0.0.2, id:11745, offset:0, len:104) in fast
 path...
00:17:35: IP_VFR: created frag state for sa:13.0.0.2, da:17.0.0.2, id:11745...
00:17:35: IP_VFR: pak incomplete cpak-offset:0, cpak-len:104, flag: 1
00:17:35: IP_VFR: dgrm incomplete, returning...
00:17:35: IP_VFR: fragment (sa:13.0.0.2, da:17.0.0.2, id:11745, offset:104, len:104) in
fast path...
00:17:35: IP_VFR: cpak-offset:0, cpak-len:104, npak-offset:104
00:17:35: IP_VFR: pak incomplete cpak-offset:104, cpak-len:104, flag: 1
00:17:35: IP_VFR: dgrm incomplete, returning...
00:17:35: IP_VFR: fragment (sa:13.0.0.2, da:17.0.0.2, id:11745, offset:208, len:104) in
fast path...
00:17:35: IP_VFR: cpak-offset:0, cpak-len:104, npak-offset:104
00:17:35: IP_VFR: cpak-offset:104, cpak-len:104, npak-offset:208
00:17:35: IP_VFR: pak incomplete cpak-offset:208, cpak-len:104, flag: 1
00:17:35: IP_VFR: dgrm incomplete, returning...
```

```
00:17:35: IP_VFR: fragment (sa:13.0.0.2, da:17.0.0.2, id:11745, offset:312, len:104) in
fast path...
00:17:35: IP_VFR: cpak-offset:0, cpak-len:104, npak-offset:104
00:17:35: IP_VFR: cpak-offset:104, cpak-len:104, npak-offset:208
00:17:35: IP_VFR: cpak-offset:208, cpak-len:104, npak-offset:312
00:17:35: IP_VFR: pak incomplete cpak-offset:312, cpak-len:104, flag: 1
00:17:35: IP_VFR: dgrm incomplete, returning...
00:17:35: IP_VFR: fragment (sa:13.0.0.2, da:17.0.0.2, id:11745, offset:416, len:92) in fast
 path...
00:17:35: IP_VFR: cpak-offset:0, cpak-len:104, npak-offset:104
00:17:35: IP_VFR: cpak-offset:104, cpak-len:104, npak-offset:208
00:17:35: IP_VFR: cpak-offset:208, cpak-len:104, npak-offset:312
00:17:35: IP_VFR: cpak-offset:312, cpak-len:104, npak-offset:416
00:17:35: IP_VFR: dgrm complete, switching the frags.
00:17:35: IP_VFR: switching fragment (sa:13.0.0.2, da:17.0.0.2, id:11745, offset:0, len:104)
00:17:35: IP_VFR: switching fragment (sa:13.0.0.2, da:17.0.0.2, id:11745, offset:104,
len:104)
00:17:35: IP_VFR: switching fragment (sa:13.0.0.2, da:17.0.0.2, id:11745, offset:208,
len:104)
00:17:35: IP_VFR: switching fragment (sa:13.0.0.2, da:17.0.0.2, id:11745, offset:312,
len:104)
00:17:35: IP_VFR: switching fragment (sa:13.0.0.2, da:17.0.0.2, id:11745, offset:416, len:92)
00:17:35: IP_VFR: all fragments have been switched.
00:17:35: IP_VFR: pak_subblock_free - pak 0x64A3DC30
00:17:35: IP_VFR: pak_subblock_free - pak 0x6430F010
00:17:35: IP_VFR: pak_subblock_free - pak 0x6430F678
00:17:35: IP_VFR: pak_subblock_free - pak 0x643119B4
00:17:35: IP_VFR: deleted frag state for sa:13.0.0.2, da:17.0.0.2, id:11745
00:17:35: IP_VFR: pak_subblock_free - pak 0x64A3D5C8
```

Related Commands

Command	Description
ip virtual-reassembly	Enables VFR on an interface.

debug ip wccp

To display information about IPv4 Web Cache Communication Protocol (WCCP) services, use the **debug ip wccp** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip wccp {**default**| **vrf** *vrf-name* {**events**| **packets [control]**}| **events**| **packets** [**bypass**| **control**| **redirect**]| **platform**| **subblocks**}

no debug ip wccp {**default**| **vrf** *vrf-name* {**events**| **packets [control]**}| **events**| **packets** [**bypass**| **control**| **redirect**]| **platform**| **subblocks**}

Syntax Description

default	Displays information about default WCCP services.
vrf *vrf-name*	Specifies a virtual routing and forwarding (VRF) instance to associate with a service group.
events	Displays information about significant WCCP events.
packets	Displays information about every WCCP packet received or sent by the router.
control	(Optional) Displays information about WCCP control packets.
bypass	(Optional) Displays information about WCCP bypass packets.
redirect	(Optional) Displays information about WCCP redirect packets.
platform	Displays information about the WCCP platform application programming interface (API).
subblocks	Displays information about WCCP subblocks.

Command Default

Debug information is not displayed.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
15.0(1)M	This command was introduced. This command replaces the **debug ip wccp packets** and **debug ip wccp events** commands.

Release	Modification
12.2(33)SRE	This command was integrated into Cisco IOS Release 12.2(33)SRE.
Cisco IOS XE Release 3.1S	This command was integrated into Cisco IOS XE Release 3.1S.

Usage Guidelines

When the **vrf** keyword is not used, the command displays debug information about all WCCP services on the router. The **default** keyword is used to specify default WCCP services.

Examples

The following is sample output from the **debug ip wccp events** command when a Cisco Cache Engine is added to the list of available Web caches:

```
Router# debug ip wccp events
WCCP-EVNT: Built I_See_You msg body w/1 usable web caches, change # 0000000A
WCCP-EVNT: Web Cache 192.168.25.3 added
WCCP-EVNT: Built I_See_You msg body w/2 usable web caches, change # 0000000B
WCCP-EVNT: Built I_See_You msg body w/2 usable web caches, change # 0000000C
```

The following is sample output from the **debug ip wccp packets** command. The router is sending keepalive packets to the Cisco Cache Engines at 192.168.25.4 and 192.168.25.3. Each keepalive packet has an identification number associated with it. When the Cisco Cache Engine receives a keepalive packet from the router, it sends a reply with the identification number back to the router.

```
Router# debug ip wccp packets
WCCP-PKT: Received valid Here_I_Am packet from 192.168.25.4 w/rcvd_id 00003532
WCCP-PKT: Sending I_See_You packet to 192.168.25.4 w/ rcvd_id 00003534
WCCP-PKT: Received valid Here_I_Am packet from 192.168.25.3 w/rcvd_id 00003533
WCCP-PKT: Sending I_See_You packet to 192.168.25.3 w/ rcvd_id 00003535
WCCP-PKT: Received valid Here_I_Am packet from 192.168.25.4 w/rcvd_id 00003534
WCCP-PKT: Sending I_See_You packet to 192.168.25.4 w/ rcvd_id 00003536
WCCP-PKT: Received valid Here_I_Am packet from 192.168.25.3 w/rcvd_id 00003535
WCCP-PKT: Sending I_See_You packet to 192.168.25.3 w/ rcvd_id 00003537
WCCP-PKT: Received valid Here_I_Am packet from 192.168.25.4 w/rcvd_id 00003536
WCCP-PKT: Sending I_See_You packet to 192.168.25.4 w/ rcvd_id 00003538
WCCP-PKT: Received valid Here_I_Am packet from 192.168.25.3 w/rcvd_id 00003537
WCCP-PKT: Sending I_See_You packet to 192.168.25.3 w/ rcvd_id 00003539
```

Related Commands

Command	Description
clear ip wccp	Clears the counter for packets redirected using WCCP.
ip wccp	Enables support of the specified WCCP service for participation in a service group.
ip wccp redirect	Enables packet redirection on an outbound or inbound interface using WCCP.
show ip interface	Lists a summary of the IP information and status of an interface.

debug ipc

To display debugging messages about interprocess communication (IPC) activity, use the **debug ipc** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ipc {**all**| **ports**| **seats**| **sessions**| **zones**}

no debug ipc {**all**| **ports**| **seats**| **sessions**| **zones**}

Syntax Description

all	Displays all debugging IPC messages. A confirmation message will appear because enabling this keyword can severely impact performance.
ports	Displays debugging messages related to the creation and deletion of IPC ports.
seats	Displays debugging messages related to the creation and deletion of IPC nodes (seats).
sessions	Displays debugging messages related to the creation and deletion of IPC sessions.
zones	Displays debugging messages related to the creation and deletion of IPC zones.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2	This command was introduced.
12.3(11)T	The **sessions** and **zones** keywords were added.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

Use the **debug ipc** command to troubleshoot IPC issues discovered when the **show ipc** command is run. The debugging output varies depending on the types of IPC packets that are selected by the different keywords.

Caution Use the **debug ipc all** command with caution because it enables the **debug ipc packets** command and the volume of output can severely impact system performance. A confirmation message is displayed. We recommend that you use one of the other keywords to focus on a specific IPC activity and to limit the volume of output.

Examples

The following example shows the confirmation message that appears when the **debug ipc all** command is entered:

```
Router# debug ipc all
This may severely impact system performance. Continue? [confirm]
```

The following example shows how to enable the display of debugging messages about IPC sessions. The debugging output varies depending on the type of IPC activity that is specified. Each entry includes some text explanation--the example below shows that the IPC control session was opened to port 0x1030000, closed, and then cleared--followed by a series of header or data fields.

```
Router# debug ipc sessions
Session level events debugging is on
*Sep 14 13:13:35.435: IPC: Control Session opened to port 0x1030000
*Sep 14 13:13:35.439: -Traceback= 40779898 4077649C 40776A00 40777040 4077554C
*Sep 14 13:13:35.439: IPC: Session 0 to port 0x1030000 closed
*Sep 14 13:13:35.439: -Traceback= 4077A9D4 40776370 4077132C 40771A58 4062EC7C 4028EC8C
40649710 4057F87C
*Sep 14 13:13:35.439: IPC: Session handle of session 0 to port 0x1030000 cleared
*Sep 14 13:13:35.439: -Traceback= 407798EC 4077A9E0 40776370 4077132C 40771A58 4062EC7C
4028EC8C 40649710 4057F87C
```

Related Commands

Command	Description
debug ipc packets	Displays debugging messages about IPC packets.
show ipc	Displays IPC information.

debug ipc acks

To display debugging messages about interprocess communication (IPC) acknowledgments (ACKs), use the **debug ipc acks** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ipc acks [**rx**| **tx**] [**dest** *destination-port-id*] [**source** *source-seat-id*] [**session** *session-id*] [**header dump**]

no debug ipc acks [**rx**| **tx**] [**dest** *destination-port-id*] [**source** *source-seat-id*] [**session** *session-id*] [**header dump**]

Syntax Description

rx	(Optional) Displays debugging messages related to the retrieval of IPC ACK messages.
tx	(Optional) Displays debugging messages related to the transmission of IPC ACK messages.
dest	(Optional) Displays debugging messages related to a destination port of IPC ACK messages. If not specified, information about all destinations is displayed. • Use the *destination-port-id* argument to specify a hexadecimal number that represents a destination port ID. The range is from 0 to FFFFFFFF.
source	(Optional) Displays debugging information about messages from an IPC node. If not specified, information about all nodes is displayed. • Use the *source-seat-id* argument to specify a hexadecimal number that represents a source seat ID. The range is from 0 to FFFFFFFF.
session	(Optional) Displays debugging messages related to an IPC session. If not specified, information about all sessions is displayed. • Use the *session-id* argument to specify a session ID. The range is from 0 to 65535.
header dump	(Optional) Displays only the packet header information.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(11)T	This command was introduced.

Usage Guidelines

Use the **debug ipc acks** command to troubleshoot IPC ACK issues. To enable debugging for other IPC activities, use the **debug ipc** command.

Examples

The following example shows how to enable the display of packet headers only when debugging IPC ACK messages. The debugging output varies depending on the type of IPC activity that is specified. Each entry includes some text explanation--the example below shows that the server received an ACK HDR--followed by a series of header or data fields.

```
Router# debug ipc acks header dump
Aug 19 03:52:36.136:IPC:Server received ACK HDR:442A64E0 src:100000A, dst:406116E8,
index:-1, seq:22045, sz:0, type:65535, flags:2 hi:1F371, lo:0
```

Related Commands

Command	Description
debug ipc	Displays IPC debugging information.

debug ipc errors

To display debugging messages about interprocess communication (IPC) errors and warnings, use the **debug ipc errors**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ipc errors [driver] [sequence] [timeout]

no debug ipc errors [driver] [sequence] [timeout]

Syntax Description

driver	(Optional) Displays debugging messages related to IPC errors at the driver (transport) medium.
sequence	(Optional) Displays information related to IPC messages that have sequence-related issues, such as duplicate or unexpected messages.
timeout	(Optional) Displays only information related to IPC messages that have timed out.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2	This command was introduced.
12.3(11)T	The **driver**, **sequence**, and **timeout** keywords were added.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

Use the **debug ipc errors** command to troubleshoot IPC error issues. To enable debugging for other IPC activities, use the **debug ipc** command. The debugging output varies depending on the type of IPC activity that is specified.

Examples

The following example shows how to enable the display of error debugging information about IPC messages that have timed out. The debugging output varies depending on the type of IPC activity that is specified. Each entry includes some text explanation--the example below shows that the message number 4428D3D0 timed out waiting for an acknowledgment (Ack)--followed by a series of header or data fields.

```
Router# debug ipc errors timeout
Message Timeouts debugging is on
*Sep 14 14:42:17.103: IPC: Message 4428D3D0 timed out waiting for Ack
*Sep 14 14:42:17.103: IPC:  MSG: ptr: 0x4428D3D0, flags: 0x88, retries: 6, seq: 0x1030002,
refcount: 2,
retry: 00:00:00, rpc_result = 0x0, data_buffer = 0x4442AB10, header = 0x4442AED4,
```

```
data = 0x4442AEF4
HDR: src: 0x10000, dst: 0x103000A, index: 0, seq: 2, sz: 512, type: 0, flags: 0x400
hi: 0x1EC, lo: 0x4442AEF4
DATA: 00 00 00 05 00 00 00 00 00 00 00 3A 00 00 00 00 00 00 00 00
```

Related Commands

Command	Description
debug ipc	Displays IPC debugging information.

debug ipc events

To display debugging messages about interprocess communication (IPC) events, use the **debug ipc events** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ipc events [flushes] [retries]

no debug ipc events [flushes] [retries]

Syntax Description

flushes	(Optional) Displays only information related to IPC messages that are flushed.
retries	(Optional) Displays only information related to IPC messages that are re-sent.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2	This command was introduced.
12.3(11)T	The **flushes** and **retries** keywords were added.\
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

Use the **debug ipc events** command to troubleshoot IPC events issues. To enable debugging for other IPC activities, use the **debug ipc** command.

Examples

The following example shows how to enable the display of debugging messages about IPC events:

```
Router# debug ipc events
Special Events debugging is on
```

The following example shows how to enable the display of event debugging information about IPC messages that are re-sent. The debugging output varies depending on the type of IPC activity that is specified. Each entry includes some text explanation--the example below shows that there was a retry attempt for a specific message--followed by a series of header or data fields.

```
Router# debug ipc events retries
Message Retries debugging is on
*Sep 14 14:46:44.151: IPC: Retry attempt for MSG: ptr: 0x442AFE74, flags: 0x88,
retries:4, seq: 0x1030003,
refcount: 2, retry: 00:00:00, rpc_result = 0x0, data_buffer = 0x445EBA44,
header =0x445EBE08, data = 0x445EBE28
HDR: src: 0x10000, dst: 0x103000A, index: 0, seq: 3, sz: 512, type: 0, flags: 0x400
```

```
hi:0x201, lo: 0x445EBE28
DATA: 00 00 00 05 00 00 00 00 00 00 00 3A 00 00 00 00 00 00 03 D2
```

Related Commands

Command	Description
debug ipc	Displays IPC debugging information.

debug ipc fragments

To display debugging messages about interprocess communication (IPC) fragments, use the **debug ipc fragments**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ipc fragments [**rx**| **tx**] [**dest** *destination-port-id*] [**source** *source-seat-id*] [**session** *session-id*] [**type** *application-type*] [**flags** *header-flag*] [**sequence** *sequence*] [**msgidhi** *msg-id-high*] [**msgidlo** *msg-id-low*] [**data offset** *offset-from-header* **value** *value-to-match* **dump** *bytes*] [**size** *size*] [**header dump**]

no debug ipc fragments [**rx**| **tx**] [**dest** *destination-port-id*] [**source** *source-seat-id*] [**session** *session-id*] [**type** *application-type*] [**flags** *header-flag*] [**sequence** *sequence*] [**msgidhi** *msg-id-high*] [**msgidlo** *msg-id-low*] [**data offset** *offset-from-header* **value** *value-to-match* **dump** *bytes*] [**size** *size*] [**header dump**]

Syntax Description

rx	(Optional) Displays debugging messages related to the retrieval of IPC fragments.
tx	(Optional) Displays debugging messages related to the transmission of IPC fragments.
dest	(Optional) Displays debugging messages related to a destination port of IPC fragments. If not specified, information about all destinations is displayed. • Use the *destination-port-id* argument to specify a hexadecimal number that represents a destination port ID. The range is from 0 to FFFFFFFF.
source	(Optional) Displays debugging information about messages from an IPC node. If not specified, information about all nodes is displayed. • Use the *source-seat-id* argument to specify a hexadecimal number that represents a source seat ID. The range is from 0 to FFFFFFFF.
session	(Optional) Displays debugging messages related to an IPC session. If not specified, information about all sessions is displayed. • Use the *session-id* argument to specify a session ID. The range is from 0 to 65535.

type	(Optional) Displays debugging messages related to a type of IPC fragments. If not specified, information about all application types is displayed. • Use the *application-type* argument to specify a hexadecimal number that represents an application. The range is from 0 to FFFF.
flags	(Optional) Displays debugging messages related to an IPC fragment's header flag. If not specified, information about all header flags is displayed. • Use the *header-flag* argument to specify a hexadecimal number that represents a header flag value. The range is from 0 to FFFF.
sequence	(Optional) Displays debugging messages related to a sequence number of an IPC fragment. If not specified, information about all sequence numbers is displayed. • Use the *sequence* argument to specify a sequence number. The range is from 0 to 65535.
msgidhi	(Optional) Displays debugging messages related to the higher byte of the unique ID of an IPC fragment. • Use the *msg-id-high* argument to specify a hexadecimal number that represents a higher byte of the unique ID. The range is from 0 to FFFFFFFF.
msgidlo	(Optional) Displays debugging messages related to the lower byte of the unique ID of an IPC fragment. • Use the *msg-id-low* argument to specify a hexadecimal number that represents a lower byte of the unique ID. The range is from 0 to FFFFFFFF.

data	(Optional) Displays debugging messages related to the IPC fragment payload. If not specified, information about all of the IPC fragment's payload is displayed.
	• **offset** --(Optional) Displays offset IPC data. If this keyword is configured, the **value** keyword must also be configured.
	  • Use the *offset-from-header* argument to specify the offset value from the start of the IPC data. The range is from 0 to 65535.
	  • Use the **value** keyword to configure the value expected at the offset of the IPC data.
	  • Use the *value-to-match* argument to specify the hexadecimal number that represents the value expected at the offset of the IPC data. The range is from 0 to FF.
	• **dump** --(Optional) Configures the number of data bytes to display.
	  • Use the *bytes* argument to specify the number of data bytes. The range is from 0 to 65535.

size	(Optional) Displays IPC fragment debugging messages of a specific size. If not specified, information about messages of any size is displayed. • Use the *size* argument to specify the message size in rows. The range is from 0 to 65535.
header dump	(Optional) Displays only the packet header information.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(11)T	This command was introduced.

Usage Guidelines

Use the **debug ipc fragments**command to troubleshoot IPC fragment issues. To enable debugging for other IPC activities, use the **debug ipc** command.

Examples

The following example shows how to enable the display of debugging information about IPC fragments. The debugging output varies depending on the type of IPC activity that is specified. Each entry includes some text explanation--the example below shows that the server received a fragment message--followed by a series of header or data fields.

```
Router# debug ipc fragments
IPC Fragments debugging is on
01:43:55: IPC: Server received fragment MSG: ptr: 0x503A4348, flags: 0x100, retries: 0,
seq: 0x0,
refcount: 1, retry: never, rpc_result = 0x0, data_buffer = 0x433809E8, header = 0x8626748,
data = 0x8626768
HDR: src: 0x10000, dst: 0x2210015, index: 0, seq: 1, sz: 1468, type: 0, flags: 0x10
hi:0x9AA, lo: 0x7D0
DATA: 00 00 00 01 00 00 00 00 00 00 00 AA 00 00 00 00 00 00 17 E4
```

Related Commands

Command	Description
debug ipc	Displays IPC debugging information.

debug ipc nacks

To display debugging messages about interprocess communication (IPC) negative acknowledgments (NACKs), use the **debug ipc nacks** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ipc nacks [**rx**| **tx**] [**dest** *destination-port-id*] [**source** *source-seat-id*] [**session** *session-id*] [**header dump**]

no debug ipc nacks [**rx**| **tx**] [**dest** *destination-port-id*] [**source** *source-seat-id*] [**session** *session-id*] [**header dump**]

Syntax Description

rx	(Optional) Displays debugging messages related to the retrieval of IPC NACK messages.
tx	(Optional) Displays debugging messages related to the transmission of IPC NACK messages.
dest	(Optional) Displays debugging messages related to a destination port of IPC NACK messages. If not specified, information about all destinations is displayed. • Use the *destination-port-id* argument to specify a hexadecimal number that represents a destination port ID. The range is from 0 to FFFFFFFF.
source	(Optional) Displays debugging information about messages from an IPC node. If not specified, information about all nodes is displayed. • Use the *source-seat-id* argument to specify a hexadecimal number that represents a source seat ID. The range is from 0 to FFFFFFFF.
session	(Optional) Displays debugging messages related to an IPC session. If not specified, information about all sessions is displayed. • Use the *session-id* argument to specify a session ID. The range is from 0 to 65535.
header dump	(Optional) Displays only the packet header information.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(11)T	This command was introduced.

Usage Guidelines

Use the **debug ipc nacks** command to troubleshoot IPC NACK issues. To enable debugging for other IPC activities, use the **debug ipc** command.

Examples

The following example shows how to enable the display of packet headers only when debugging IPC NACK messages. The debugging output varies depending on the type of IPC activity that is specified. Each entry includes some text explanation--the example below shows that the server sent a NACK message and received a NACK header--followed by a series of header or data fields.

```
Router# debug ipc nacks header dump
IPC Nacks debugging is on
01:46:11: IPC: Server sent NACK MSG: ptr: 0x432A7428, flags: 0x100, retries: 0, seq: 0x0,
refcount: 1, retry: never, rpc_result = 0x0, data_buffer = 0x431E4B50, header = 0x855F508,
data = 0x855F528
HDR: src: 0x2210015, dst: 0x10000, index: 1, seq: 3, sz: 0, type: 0, flags: 0x100
hi: 0x4A9, lo: 0x85AA3E8
01:46:11: SP: IPC: Server received NACK HDR: E46A448 src: 2210015, dst: 10000, index: 1,
seq: 3, sz: 0, type: 0, flags: 100 hi: 4A9, lo: 85AA3E8
```

Related Commands

Command	Description
debug ipc	Displays IPC debugging information.

debug ipc packets

To display debugging messages about interprocess communication (IPC) packets, use the **debug ipc packets** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ipc packets [**rx**| **tx**] [**dest** *destination-port-id*] [**source** *source-seat-id*] [**session** *session-id*] [**type** *application-type*] [**flags** *header-flag*] [**sequence** *sequence*] [**msgidhi** *msg-id-high*] [**msgidlo** *msg-id-low*] [**data offset** *offset-from-header* **value** *value-to-match* **dump** *bytes*] [**size** *size*] [**header dump**]

no debug ipc packets [**rx**| **tx**] [**dest** *destination-port-id*] [**source** *source-seat-id*] [**session** *session-id*] [**type** *application-type*] [**flags** *header-flag*] [**sequence** *sequence*] [**msgidhi** *msg-id-high*] [**msgidlo** *msg-id-low*] [**data offset** *offset-from-header* **value** *value-to-match* **dump** *bytes*] [**size** *size*] [**header dump**]

Syntax Description

rx	(Optional) Displays debugging messages related to the retrieval of IPC packets.
tx	(Optional) Displays debugging messages related to the transmission of IPC packets.
dest	(Optional) Displays debugging messages related to a destination port of IPC packets. If not specified, information about all destinations is displayed. • Use the *destination-port-id* argument to specify a hexadecimal number that represents a destination port ID. The range is from 0 to FFFFFFFF.
source	(Optional) Displays debugging information about messages from an IPC node. If not specified, information about all nodes is displayed. • Use the *source-seat-id* argument to specify a hexadecimal number that represents a source seat ID. The range is from 0 to FFFFFFFF.
session	(Optional) Displays debugging messages related to an IPC session. If not specified, information about all sessions is displayed. • Use the *session-id* argument to specify a session ID. The range is from 0 to 65535.

type	(Optional) Displays debugging messages related to a type of IPC packet. If not specified, information about all application types is displayed. • Use the *application-type* argument to specify a hexadecimal number that represents an application. The range is from 0 to FFFF.
flags	(Optional) Displays debugging messages related to an IPC packet header flag. If not specified, information about all header flags is displayed. • Use the *header-flag* argument to specify a hexadecimal number that represents a header flag value. The range is from 0 to FFFF.
sequence	(Optional) Displays debugging messages related to a sequence number of an IPC packet. If not specified, information about all sequence numbers is displayed. • Use the *sequence* argument to specify a sequence number. The range is from 0 to 65535.
msgidhi	(Optional) Displays debugging messages related to the higher byte of the unique ID of an IPC packet. • Use the *msg-id-high* argument to specify a hexadecimal number that represents a higher byte of the unique ID. The range is from 0 to FFFFFFFF.
msgidlo	(Optional) Displays debugging messages related to the lower byte of the unique ID of an IPC packet. • Use the *msg-id-low* argument to specify a hexadecimal number that represents a lower byte of the unique ID. The range is from 0 to FFFFFFFF.

data	(Optional) Displays debugging messages related to the IPC packet payload. If not specified, information about all of the IPC packet's payload is displayed.
	• **offset** --(Optional) Displays offset IPC data. If this keyword is configured, the **value** keyword must also be configured.
	  • Use the *offset-from-header* argument to specify the offset value from the start of the IPC data. The range is from 0 to 65535.
	  • Use the **value** keyword to configure the value expected at the offset of the IPC data.
	  • Use the *value-to-match* argument to specify the hexadecimal number that represents the value expected at the offset of the IPC data. The range is from 0 to FF.
	• **dump** --(Optional) Configures the number of data bytes to display.
	  • Use the *bytes* argument to specify the number of data bytes. The range is from 0 to 65535.

size	(Optional) Displays IPC packet debugging messages of a specific size. If not specified, information about messages of any size is displayed. • Use the *size* argument to specify the message size in rows. The range is from 0 to 65535.
header dump	(Optional) Displays only the packet header information.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(11)T	This command was introduced.

Usage Guidelines

Use the **debug ipc packets** command to troubleshoot IPC packet issues. To enable debugging for other IPC activities, use the **debug ipc** command.

Caution Use the **debug ipc packets** command with caution because the volume of output can severely impact system performance. A confirmation message is displayed. We recommend that you use one of the optional keywords to focus on a specific IPC activity and to limit the volume of output.

Examples

The following example shows how to enable the display of IPC packet debugging messages and includes some sample output. The debugging output varies depending on the type of IPC activity that is specified. Each entry includes some text explanation--the example below shows that the IPC server received a message--followed by a series of header or data fields.

```
Router# debug ipc packets
This may severely impact system performance. Continue?[confirm] Y
Aug 19 030612.297 IPC Server received MSG ptr 0x441BE75C, flags 0x80, retries 0,
seq 0x0, refcount 1, retry never, rpc_result = 0x0, data_buffer = 0x443152A8,
header = 0x4431566C, data = 0x4431568C
HDR src 0x1060000, dst 0x1000C, index 2, seq 0, sz 28, type 770,
flags 0x40 hi 0x1F25B, lo 0x442F0BC0
DATA 00 00 00 06 00 00 00 02 00 00 00 06 00 E7 00 02 00 00 00 00
```

The following example shows how to enable the display of IPC messages received with a destination port of 0x1000C in session 1 with a message size of 500 rows.

```
Router# debug ipc packets rx dest 1000C session 1 size 500
```

Related Commands

Command	Description
debug ipc	Displays IPC debugging information.

debug ipc rpc

To display debugging messages about interprocess communication (IPC) remote-procedure call (RPC) packets, use the **debug ipc rpc**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ipc rpc [**rx**| **tx**] [**query**| **response**] [**dest** *destination-port-id*] [**source** *source-seat-id*] [**session** *session-id*] [**type** *application-type*] [**flags** *header-flag*] [**sequence** *sequence*] [**msgidhi** *msg-id-high*] [**msgidlo** *msg-id-low*] [**data offset** *offset-from-header* **value** *value-to-match* **dump** *bytes*] [**size** *size*] [**header dump**]

no debug ipc rpc [**rx**| **tx**] [**query**| **response**] [**dest** *destination-port-id*] [**source** *source-seat-id*] [**session** *session-id*] [**type** *application-type*] [**flags** *header-flag*] [**sequence** *sequence*] [**msgidhi** *msg-id-high*] [**msgidlo** *msg-id-low*] [**data offset** *offset-from-header* **value** *value-to-match* **dump** *bytes*] [**size** *size*] [**header dump**]

Syntax Description

rx	(Optional) Displays debugging messages related to the retrieval of IPC RPC packets.
tx	(Optional) Displays debugging messages related to the transmission of IPC RPC packets.
query	(Optional) Displays debugging messages related to IPC RPC queries.
response	(Optional) Displays debugging messages related to IPC RPC responses.
dest	(Optional) Displays debugging messages related to a destination port of IPC RPC packets. If not specified, information about all destinations is displayed. • Use the *destination-port-id* argument to specify a hexadecimal number that represents a destination port ID. The range is from 0 to FFFFFFFF.
source	(Optional) Displays debugging information about messages from an IPC node. If not specified, information about all nodes is displayed. • Use the *source-seat-id* argument to specify a hexadecimal number that represents a source seat ID. The range is from 0 to FFFFFFFF.
session	(Optional) Displays debugging messages related to an IPC session. If not specified, information about all sessions is displayed. • Use the *session-id* argument to specify a session ID. The range is from 0 to 65535.

type	(Optional) Displays debugging messages related to a type of IPC RPC message. If not specified, information about all application types is displayed. • Use the *application-type* argument to specify a hexadecimal number that represents an application. The range is from 0 to FFFF.
flags	(Optional) Displays debugging messages related to an IPC RPC message header flag. If not specified, information about all header flags is displayed. • Use the *header-flag* argument to specify a hexadecimal number that represents a header flag value. The range is from 0 to FFFF.
sequence	(Optional) Displays debugging messages related to a sequence number of an IPC RPC message. If not specified, information about all sequence numbers is displayed. • Use the *sequence* argument to specify a sequence number. The range is from 0 to 65535.
msgidhi	(Optional) Displays debugging messages related to the higher byte of the unique ID of an IPC RPC message. • Use the *msg-id-high* argument to specify a hexadecimal number that represents a higher byte of the unique ID. The range is from 0 to FFFFFFFF.
msgidlo	(Optional) Displays debugging messages related to the lower byte of the unique ID of an IPC RPC message. • Use the *msg-id-low* argument to specify a hexadecimal number that represents a lower byte of the unique ID. The range is from 0 to FFFFFFFF.

data	(Optional) Displays debugging messages related to the IPC RPC payload. If not specified, information about all of the IPC RPC's payload is displayed.
	• **offset** --(Optional) Displays offset IPC data. If this keyword is configured, the **value** keyword must also be configured.
	  • Use the *offset-from-header* argument to specify the offset value from the start of the IPC data. The range is from 0 to 65535.
	  • Use the **value** keyword to configure the value expected at the offset of the IPC data.
	  • Use the *value-to-match* argument to specify the hexadecimal number that represents the value expected at the offset of the IPC data. The range is from 0 to FF.
	• **dump** --(Optional) Configures the number of data bytes to display.
	  • Use the *bytes* argument to specify the number of data bytes. The range is from 0 to 65535.

size	(Optional) Displays IPC RPC debugging messages of a specific size. If not specified, information about messages of any size is displayed. • Use the *size* argument to specify the message size in rows. The range is from 0 to 65535.
header dump	(Optional) Displays only the packet header information.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(11)T	This command was introduced.

Usage Guidelines

Use the **debug ipc rpc**command to troubleshoot IPC RPC packet issues. To enable debugging for other IPC activities, use the **debug ipc** command. The debugging output varies depending on the type of IPC activity that is specified.

Examples

The following example shows how to enable the display of packet headers only when debugging IPC RPC response messages. The debugging output varies depending on the type of IPC activity that is specified. Each entry includes some text explanation--the example below shows that the server received an RPC response--followed by a series of header or data fields.

```
Router# debug ipc rpc response header dump source 2210003
RPC debugging is on
01:53:43: SP: IPC: Server received RPC Reply HDR: E450048 src: 2210003, dst: 10000,
index:0, seq: 1716, sz: 4, type: 2914, flags: 208 hi: A07, lo: E264DE8
```

Related Commands

Command	Description
debug ipc	Displays IPC debugging information.

debug iphc ipc

To display the IP header compression (IPHC) interprocessor communication (IPC) messages that are passed between the route processor (RP) and line cards (LCs), use the **debug iphc ipc**command in privileged EXEC mode. To disable the display of these messages, use the **no** form of this command.

debug iphc ipc [**events**| **statistics**]

no debug iphc ipc [**events**| **statistics**]

Syntax Description

events	(Optional) Displays IPHC IPC command and control events.
statistics	(Optional) Displays IPHC IPC counter updates.

Command Default

IPHC IPC messages are not displayed.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(32)SY	This command was introduced.
12.4(10)	This command was integrated into Cisco IOS Release 12.4(10).

Usage Guidelines

If you issue the **debug iphc ipc** command without keywords, all the IPC messages that are passed between the RP and the LC are displayed. On routers with many interfaces and distributed systems, the number of IPC messages becomes unwieldy, because of all the counter updates. To display only the events that indicate interface state changes, issue the **debug iphc ipc events** command.

Examples

The following example enables the display of all IPHC IPC messages:

```
Router# debug iphc ipc
IPHC IPC statistics debugging is on
IPHC IPC event debugging is on
The following example disables IPHC IPC statistics debugging:
Router# no debug iphc ipc statistics
IPHC IPC statistics debugging is off
```

The following example enables the display of IPHC IPC event messages:

```
Router# debug iphc ipc events
IPHC IPC event debugging is on
```

The command output shows the event messages as the interface changes from enabled to administratively down:

```
%OSPF-5-ADJCHG: Process 1, Nbr 10.10.10.10 on Multilink8 from FULL to DOWN
%LINK-5-CHANGED: Interface Multilink8, changed state to administratively down.
IPHC IPC 2: Set Negotiated mesg (Mu PPP 128 2 0)
IPHC Mu8: Distributed FS disabled
IPHC IPC 2: Send Set Configured mesg (Mu PPP 128 2 0)
IPHC IPC Mu8: i/f state change complete (Up/Down: 0/1)
```

The following example enables the display of IPHC IPC counter updates:

```
Router# debug iphc ipc statistics
IPHC IPC statistics debugging is on
```

The command output shows the interface counter updates:

```
IPHC IPHC 2: recv Stats msg, count:4
IPHC IPC Mu8: stats update from LC
IPHC IPC Mu6: stats update from LC
IPHC IPC Se2/0/0/3:0: stats update from LC
IPHC IPC Se2/0/0/1:0: stats update from LC
```

Related Commands

Command	Description
show interfaces	Displays statistics for all interfaces.
show ipc	Displays IPC statistics.

debug ipv6 cef drop

To display debug messages for Cisco Express Forwarding for IPv6 (CEFv6) and distributed CEFv6 (dCEFv6) dropped packets, use the **debug ipv6 cef drop**command in privileged EXEC mode. To disable debug messages for CEFv6 and dCEFv6 dropped packets, use the **no** form of this command.

debug ipv6 cef drop [rpf]

no debug ipv6 cef drop

Syntax Description

rpf	(Optional) Displays packets dropped by the IPv6 CEF Unicast Reverse-Path Forwarding (Unicast RPF) feature.

Command Default

Debugging for CEFv6 and dCEFv6 dropped packets is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(22)S	This command was introduced.
12.2(13)T	This command was integrated into Cisco IOS Release 12.2(13)T.
12.2(14)S	This command was integrated into Cisco IOS Release 12.2(14)S.
12.2(25)S	The **rpf** keyword was added.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.

Usage Guidelines

The **debug ipv6 cef drop** command is similar to the **debug ip cef drops** command, except that it is IPv6-specific.

Note

By default, the network server sends the output from **debug** commands and system error messages to the console. To redirect debug output, use the **logging** command options in global configuration mode. Destinations include the console, virtual terminals, internal buffer, and UNIX hosts running a syslog server. For complete information on **debug** commands and redirecting debug output, refer to the Release 12.3 *Cisco IOS Debug Command Reference* .

Examples

The following is sample output from the **debug ipv6 cef drop**command:

```
Router# debug ipv6 cef drop
*Aug 30 08:20:51.169: IPv6-CEF: received packet on Serial6/0/2
*Aug 30 08:20:51.169: IPv6-CEF: found no adjacency for 2001:0DB8::1 reason 2
*Aug 30 08:20:51.169: IPv6-CEF: packet not switched: code 0x1
```

The table below describes the significant fields shown in the display.

Table 73: debug ipv6 cef drop Field Descriptions

Field	Description
IPv6-CEF: received packet on Serial6/0/2	Cisco Express Forwarding has received a packet addressed to the router via serial interface 6/0/2.
IPv6-CEF: found no adjacency for 2001:0DB8::1	Cisco Express Forwarding has found no adjacency for the IPv6 address prefix of 2001:0DB8::1.
IPv6-CEF: packet not switched	Cisco Express Forwarding has dropped the packet.

Related Commands

Command	Description
debug ipv6 cef events	Displays debug messages for CEFv6 and dCEFv6 general events.
debug ipv6 cef table	Displays debug messages for CEFv6 and dCEFv6 table modification events.

debug ipv6 cef events

To display debug messages for Cisco Express Forwarding for IPv6 (CEFv6) and distributed CEFv6 (dCEFv6) general events, use the **debug ipv6 cef events**command in privileged EXEC mode. To disable debug messages for CEFv6 and dCEFv6 general events, use the **no** form of this command.

debug ipv6 cef events

no debug ipv6 cef events

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging for CEFv6 and dCEFv6 general events is not enabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.0(22)S	This command was introduced.
12.2(13)T	This command was integrated into Cisco IOS Release 12.2(13)T.
12.2(14)S	This command was integrated into Cisco IOS Release 12.2(14)S.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.

Usage Guidelines

The **debug ipv6 cef events**command is similar to the **debug ip cef events**command, except that it is IPv6-specific.

Note By default, the network server sends the output from debug commands and system error messages to the console. To redirect debug output, use the logging command options in global configuration mode. Destinations include the console, virtual terminals, internal buffer, and UNIX hosts running a syslog server. For complete information on **debug** commands and redirecting debug output, refer to the Release 12 *Cisco IOS Debug Command Reference* .

Examples

The following is sample output from the **debug ipv6 cef events**command:

```
Router# debug ipv6 cef events
IPv6 CEF packet events debugging is on
Router#
*Aug 30 08:22:57.809: %LINK-3-UPDOWN: Interface Serial6/0/2, changed state to up
*Aug 30 08:22:58.809: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial6/0/2, changed
 state to up
*Aug 30 08:23:00.821: CEFv6-IDB: Serial6/0/2 address 2001:0DB8::248 add download succeeded
```

The table below describes the significant fields shown in the display.

Table 74: debug ipv6 cef events Field Descriptions

Field	Description
Interface Serial6/0/2, changed state to up	Indicates that the interface hardware on serial interface 6/0/2 is currently active.
Line protocol on Interface Serial6/0/2, changed state to up	Indicates that the software processes that handle the line protocol consider the line usable for serial interface 6/0/2.
Serial6/0/2 address 2001:0DB8::248 add download succeeded	The IPv6 address 2001:0DB8::248 was downloaded successfully.

Related Commands

Command	Description
debug ipv6 cef table	Displays debug messages for CEFv6 and dCEFv6 table modification events.

debug ipv6 cef hash

To display debug messages for Cisco Express Forwarding for IPv6 (CEFv6) and distributed CEFv6 (dCEFv6) load-sharing hash algorithm events, use the **debug ipv6 cef hash**command in privileged EXEC mode. To disable debug messages for CEFv6 and dCEFv6 load-sharing hash algorithm events, use the **no** form of this command.

debug ipv6 cef hash

no debug ipv6 cef hash

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging for CEFv6 and dCEFv6 load-sharing hash algorithm events is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(22)S	This command was introduced.
12.2(13)T	This command was integrated into Cisco IOS Release 12.2(13)T.
12.2(14)S	This command was integrated into Cisco IOS Release 12.2(14)S.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.

Usage Guidelines

The **debug ipv6 cef hash**command is similar to the **debug ip cef hash**command, except that it is IPv6-specific.

Use this command when changing the load-sharing algorithm to display IPv6 hash table details.

Note By default, the network server sends the output from debug commands and system error messages to the console. To redirect debug output, use the logging command options in global configuration mode. Destinations include the console, virtual terminals, internal buffer, and UNIX hosts running a syslog server.

Related Commands

Command	Description
debug ipv6 cef events	Displays debug messages for CEFv6 and dCEFv6 general events.
debug ipv6 cef table	Displays debug messages for CEFv6 and dCEFv6 table modification events.

debug ipv6 cef receive

To display debug messages for Cisco Express Forwarding for IPv6 (CEFv6) and distributed CEFv6 (dCEFv6) packets that are process-switched on the router, use the **debug ipv6 cef receive**command in privileged EXEC mode. To disable debug messages for CEFv6 and dCEFv6 packets that are process-switched on the router, use the **no** form of this command.

debug ipv6 cef receive

no debug ipv6 cef receive

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging for CEFv6 and dCEFv6 packets that are process-switched on the router is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(22)S	This command was introduced.
12.2(13)T	This command was integrated into Cisco IOS Release 12.2(13)T.
12.2(14)S	This command was integrated into Cisco IOS Release 12.2(14)S.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.

Usage Guidelines

The **debug ipv6 cef receive**command is similar to the **debug ip cef receive**command, except that it is IPv6-specific.

Note

By default, the network server sends the output from debug commands and system error messages to the console. To redirect debug output, use the logging command options in global configuration mode. Destinations include the console, virtual terminals, internal buffer, and UNIX hosts running a syslog server. For complete information on debug commands and redirecting debug output, refer to the Release 12 *Cisco IOS Debug Command Reference* .

Examples

The following is sample output from the **debug ipv6 cef receive**command when another router in the network pings 2001:0DB8::2 which is a local address on this box:

```
Router# debug ipv6 cef receive
IPv6 CEF packet receives debugging is on
router#
*Aug 30 08:25:14.869: IPv6CEF-receive: Receive packet for 2001:0DB8::2
*Aug 30 08:25:14.897: IPv6CEF-receive: Receive packet for 2001:0DB8::2
*Aug 30 08:25:14.925: IPv6CEF-receive: Receive packet for 2001:0DB8::2
*Aug 30 08:25:14.953: IPv6CEF-receive: Receive packet for 2001:0DB8::2
*Aug 30 08:25:14.981: IPv6CEF-receive: Receive packet for 2001:0DB8::2
```

The table below describes the significant fields shown in the display.

Table 75: debug ipv6 cef receive Field Descriptions

Field	Description
IPv6CEF-receive: Receive packet for 2001:0DB8::2	Cisco Express Forwarding has received a packet addressed to the router.

Related Commands

Command	Description
debug ipv6 cef events	Displays debug messages for CEFv6 and dCEFv6 general events.
debug ipv6 cef table	Displays debug messages for CEFv6 and dCEFv6 table modification events.

debug ipv6 cef table

To display debug messages for Cisco Express Forwarding for IPv6 (CEFv6) and distributed CEFv6 (dCEFv6) table modification events, use the **debug ipv6 cef table**command in privileged EXEC mode. To disable debug messages for CEFv6 and dCEFv6 table modification events, use the **no** form of this command.

debug ipv6 cef table [background]

no debug ipv6 cef table [background]

Syntax Description

background	(Optional) Sets CEFv6 and dCEFv6 table background updates.

Command Default

Debugging for CEFv6 and dCEFv6 table modification events is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(22)S	This command was introduced.
12.2(13)T	This command was integrated into Cisco IOS Release 12.2(13)T.
12.2(14)S	This command was integrated into Cisco IOS Release 12.2(14)S.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.

Usage Guidelines

The **debug ipv6 cef table**command is similar to the **debug ip cef table**command, except that it is IPv6-specific.

This command is used to record CEFv6 and dCEFv6 table events related to the Forwarding Information Base (FIB) tables. Types of events include the following:

- Routing updates that populate the FIB tables
- Flushing of the FIB tables
- Adding or removing of entries to the FIB tables
- Table reloading process

Note By default, the network server sends the output from debug commands and system error messages to the console. To redirect debug output, use the logging command options in global configuration mode. Destinations include the console, virtual terminals, internal buffer, and UNIX hosts running a syslog server. For complete information on debug commands and redirecting debug output, refer to the *Cisco IOS Debug Command Reference* .

Examples

The following is sample output from the **debug ipv6 cef table**command when a static route is added:

```
Router# debug ipv6 cef table
IPv6 CEF table debugging is on
router(config)# ipv6 route 5555::/64 serial 2/0 3000::2
router(config)#
*Feb 24 08:46:09.187: IPv6CEF-Table: Event add, 5555::/64
*Feb 24 08:46:09.187: IPv6 CEF table: Created path_list 01184570
*Feb 24 08:46:09.187: IPv6 CEF table: Adding path 01181A80 to path_list 01184570 old path
count=0
*Feb 24 08:46:09.187: IPv6 CEF table: No matching list for path list 01184570
*Feb 24 08:46:09.187: IPv6 CEF table: Adding fib entry 0117EE80 to path_list 01184570 old
refcount=0
*Feb 24 08:46:09.187: IPv6 CEF table: Added path_list 01184570 to hash 50
*Feb 24 08:46:09.187: IPv6 CEF: Linking path 01181A80 to adjacency 01138E28
*Feb 24 08:46:09.187: IPv6 CEF table: Created 0 loadinfos for path_list 01184570
*Feb 24 08:46:09.187: IPv6CEF-Table: Validated 5555::/64
```

The following is sample output when the static route is removed:

```
router(config)# no ipv6 route 5555::/64 serial 2/0 3000::2
router(config)#
*Feb 24 08:46:43.871: IPv6CEF-Table: Event delete, 5555::/64
*Feb 24 08:46:43.871: IPv6CEF-Table: Invalidated 5555::/64
*Feb 24 08:46:43.871: IPv6CEF-Table: Deleted 5555::/64
*Feb 24 08:46:43.871: IPv6 CEF table: Removing fib entry 0117EE80 from path_list 01184570
old refcount=1
*Feb 24 08:46:43.871: IPv6 CEF table: Removed path_list 01184570 from hash 50
*Feb 24 08:46:43.871: IPv6 CEF table: Freeing path_list 01184570 refcount=0
*Feb 24 08:46:43.871: IPv6 CEF table: Freeing all 1 paths in path_list 01184570
*Feb 24 08:46:43.871: IPv6 CEF: deleting path 01181A80
```

Related Commands

Command	Description
debug ipv6 cef events	Displays debug messages for CEFv6 and dCEFv6 general events.

debug ipv6 dhcp

To enable debugging for Dynamic Host Configuration Protocol (DHCP) for IPv6, use the **debug ipv6 dhcp** command in privileged EXEC mode. To disable debugging for DHCP for IPv6, use the **no** form of this command.

debug ipv6 dhcp [detail]

no debug ipv6 dhcp [detail]

Syntax Description

detail	(Optional) Displays detailed information about DHCP for IPv6 message decoding.

Command Default

Debugging for the DHCP for IPv6 is disabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(4)T	This command was introduced.
12.4(24)T	This command was integrated into Cisco IOS Release 12.4(24)T.
Cisco IOS XE Release 2.1	This command was integrated into Cisco IOS XE Release 2.1.
12.2(33)SRE	This command was modified. It was integrated into Cisco IOS Release 12.2(33)SRE.

Usage Guidelines

The **debug ipv6 dhcp detail** command is used to show debug information related to the server address assignment.

Examples

The following example enables debugging for DHCP for IPv6:

```
Router# debug ipv6 dhcp detail
IPv6 DHCP debugging is on (detailed)
```

Related Commands

Command	Description
debug ipv6 dhcp database	Enables debugging for the DHCP for IPv6 binding database agent.

Command	Description
debug ipv6 dhcp relay	Enables the DHCP for IPv6 relay agent debugging.

debug ipv6 dhcp database

To enable debugging for the Dynamic Host Configuration Protocol (DHCP) for IPv6 binding database agent, use the **debug ipv6 dhcp database** command in privileged EXEC mode. To disable the display of debug messages for the DHCP for IPv6 binding database agent, use the **no** form of this command.

debug ipv6 dhcp database

no debug ipv6 dhcp database

Syntax Description

This command has no keywords or arguments.

Command Default

Debugging for the DHCP for IPv6 binding database agent is disabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(4)T	This command was introduced.
Cisco IOS XE Release 2.1	This command was integrated into Cisco IOS XE Release 2.1.

Usage Guidelines

The **debug ipv6 dhcp database** command enables debugging for DHCP for IPv6 database processing.

Examples

The following example enables debugging for the DHCP for IPv6 binding database agent:

```
Router# debug ipv6 dhcp database
```

Related Commands

Command	Description
debug ipv6 dhcp	Enables debugging for DHCP for IPv6.

debug ipv6 dhcp redundancy

To enable Dynamic Host Configuration Protocol for IPv6 (DHCPv6) server redundancy debugging, use the **debug ipv6 dhcp redundancy** command in privileged EXEC mode. To disable DHCPv6 server redundancy debugging, use the **no** form of this command.

debug ipv6 dhcp redundancy [detail]

no debug ipv6 dhcp redundancy [detail]

Syntax Description

detail	(Optional) Displays detailed DHCPv6 High Availability (HA) packet information.

Command Default

DHCPv6 server redundancy debugging is disabled by default.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
15.2(1)S	This command was introduced.
Cisco IOS XE Release 3.5S	This command was integrated into Cisco IOS XE Release 3.5S.

Usage Guidelines

To debug DHCPv6 server redundancy, use the **debug ipv6 dhcp redundancy** command in privileged EXEC mode. To view detailed DHCPv6 HA packet information, use the optional **detail** keyword.

Examples

The following example shows how to enable DHCPv6 redundancy debugging:

```
Router# debug ipv6 dhcp redundancy
```

Related Commands

Command	Description
debug ipv6 dhcp relay	Enables DHCPv6 relay agent debugging.

debug ipv6 dhcp relay

To enable DHCP for IPv6 relay agent debugging, use the **debug ipv6 dhcp relay**command in user EXEC or privileged EXEC mode. To disable DHCP for IPv6 relay agent debugging, use the **no** form of this command.

debug ipv6 dhcp relay [bulk-lease]

no debug ipv6 dhcp relay [bulk-lease]

Syntax Description

bulk-lease	(Optional) Enables bulk lease query debugging flows.

Command Modes

User EXEC (>) Privileged EXEC (#)

Command History

Release	Modification
12.3(11)T	This command was introduced.
Cisco IOS XE Release 2.1	This command was integrated into Cisco IOS XE Release 2.1.
15.1(1)S	This command was modified. The **bulk-lease** keyword was added.

Usage Guidelines

The DHCP functions for IPv6 client, server, and relay agent are mutually exclusive on an interface. When one of these functions is enabled and a user tries to configure a different function on the same interface, one of the following messages is displayed: Interface is in DHCP client mode, Interface is in DHCP server mode, or Interface is in DHCP relay mode.

Examples

The following example enables DHCP for IPv6 relay agent debugging:

```
Router# debug ipv6 dhcp relay
```

Related Commands

Command	Description
debug ipv6 dhcp	Enables DHCP debugging for IPv6.

debug ipv6 eigrp

To display information about the Enhanced Interior Gateway Routing Protocol (EIGRP) for IPv6 protocol, use the **debug ipv6 eigrp** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ipv6 eigrp [*as-number*] [**neighbor** *ipv6-address*| **notification**| **summary**]

no debug ipv6 eigrp

Syntax Description

as-number	(Optional) Autonomous system number.
neighbor *ipv6-address*	(Optional) IPv6 address of the neighboring router.
notification	(Optional) Displays EIGRP for IPv6 events and notifications in the console of the router.
summary	(Optional) Displays a summary of EIGRP for IPv6 routing information.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.4(6)T	This command was introduced.
12.2(33)SRB	This command was integrated into Cisco IOS Release 12.2(33)SRB.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.
Cisco IOS XE Release 2.1	This command was introduced on Cisco ASR 1000 Series Routers.

Usage Guidelines

Because the **debug ipv6 eigrp** command generates a substantial amount of output, use it only when traffic on the network is light.

Examples

The following example enables debugging output:

```
Router# debug ipv6 eigrp
```

debug ipv6 icmp

To display debugging messages for IPv6 Internet Control Message Protocol (ICMP) transactions (excluding IPv6 ICMP neighbor discovery transactions), use the **debug ipv6 icmp**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ipv6 icmp

no debug ipv6 icmp

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging for IPv6 ICMP is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(2)T	This command was introduced.
12.0(21)ST	This command was integrated into Cisco IOS Release 12.0(21)ST.
12.0(22)S	This command was integrated into Cisco IOS Release 12.0(22)S.
12.2(14)S	This command was integrated into Cisco IOS Release 12.2(14)S.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(25)SG	This command was integrated into Cisco IOS Release 12.2(25)SG.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.
12.2(33)SB	This command's output was modified on the Cisco 10000 series router for the PRE3 and PRE4.
15.1(1)S	This command was integrated into Cisco IOS 15.1(1)S.

Usage Guidelines

The **debug ipv6 icmp**command is similar to the **debug ip icmp**command, except that it is IPv6-specific. When you run this command, you can view echo reply messages that are generated in response to echo requests.

Note By default, the network server sends the output from **debug** commands and system error messages to the console. To redirect debugging output, use the logging command options in global configuration mode. Destinations include the console, virtual terminals, internal buffer, and UNIX hosts running a syslog server.

This command helps you determine whether the router is sending or receiving IPv6 ICMP messages. Use it, for example, when you are troubleshooting an end-to-end connection problem.

Note For more information about the fields in **debug ipv6 icmp** output, refer to RFC 2463, *Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6)* .

Cisco 10000 Series Router Usage Guidelines

In Cisco IOS Release 12.2(33)SB, output from the debug ipv6 icmp command displays information similar to the following:

```
ICMPv6: Received echo reply from 2010:1:1:1:1:1:1:2
```

In Cisco IOS Release 12.2(31)SB, the debug ipv6 icmp command output displays information similar to the following:

```
ICMPv6: Received ICMPv6 packet from 2010:1:1:1:1:1:1:2, type 129
```

Examples

The following is sample output from the **debug ipv6 icmp**command:

```
Router# debug ipv6 icmp
13:28:40:ICMPv6:Received ICMPv6 packet from 2000:0:0:3::2, type 136
13:28:45:ICMPv6:Received ICMPv6 packet from FE80::203:A0FF:FED6:1400, type 135
13:28:50:ICMPv6:Received ICMPv6 packet from FE80::203:A0FF:FED6:1400, type 136
13:28:55:ICMPv6:Received ICMPv6 packet from FE80::203:A0FF:FED6:1400, type 135
```

The table below describes significant fields shown in the first line of the display.

Table 76: debug ipv6 icmp Field Descriptions

Field	Description
13:28:40:	Indicates the time (hours:minutes:seconds) at which the ICMP neighbor discovery event occurred.
n w*n* d: (not shown in sample output)	Indicates time (weeks, days) since last reboot of the event occurring. For example, 1w4d: indicates the time (since the last reboot) of the event occurring was 1 week and 4 days ago.
ICMPv6:	Indication that this message describes an ICMP version 6 packet.
Received ICMPv6 packet from 2000:0:0:3::2	IPv6 address from which the ICMP version 6 packet is received.

Field	Description
type 136	The number variable indicates one of the following IPv6 ICMP message types: • 1--Destination unreachable. The router cannot forward a packet that was sent or received. • 2--Packet too big. The router attempts to send a packet that exceeds the maximum transmission unit (MTU) of a link between itself and the packet destination. • 3--Time exceeded. Either the hop limit in transit or the fragment reassembly time is exceeded. • 4--Parameter problem. The router attempts to send an IPv6 packet that contains invalid parameters. An example is a packet containing a next header type unsupported by the router that is forwarding the packet. • 128--Echo request. The router received an echo reply. • 129--Echo reply. The router sent an echo reply. • 133--Router solicitation messages. Hosts send these messages to prompt routers on the local link to send router advertisement messages. • 134--Router advertisement messages. Routers periodically send these messages to advertise their link-layer addresses, prefixes for the link, and other link-specific information. These messages are also sent in response to router solicitation messages. • 135--Neighbor solicitation messages. Nodes send these messages to request the link-layer address of a station on the same link. • 136--Neighbor advertisement messages. Nodes send these messages, containing their link-local addresses, in response to neighbor solicitation messages. • 137--Redirect messages. Routers send these messages to hosts when a host attempts to use a less-than-optimal first hop address when forwarding packets. These messages contain a better first hop address that should be used instead.

Following are examples of the IPv6 ICMP messages types that can be displayed by the **debug ipv6 icmp** command:

- ICMP echo request and ICMP echo reply messages. In the following example, an ICMP echo request is sent to address 2052::50 and an ICMP echo reply is received from address 2052::50.

```
1w4d:ICMPv6:Sending echo request to 2052::50
1w4d:ICMPv6:Received echo reply from 2052::50
```

- ICMP packet too big messages. In the following example, a router tried to forward a packet to destination address 2052::50 via the next hop address 2052::52. The size of the packet was greater than 1280 bytes, which is the MTU of destination address 2052::50. As a result, the router receives an ICMP packet too big message from the next hop address 2052::52.

```
1w4d:Received ICMP too big from 2052::52 about 2052::50, MTU=1300
```

- ICMP parameter problem messages. In the following example, an ICMP parameter problem message is received from address 2052::52.

```
1w4d:Received ICMP parameter problem from 2052::52
```

- ICMP time exceeded messages. In the following example, an ICMP time exceeded message is received from address 2052::52.

```
1w4d:Received ICMP time exceeded from 2052::52
```

- ICMP unreachable messages. In the following example, an ICMP unreachable message with code 1 is received from address 2052::52. Additionally, an ICMP unreachable message with code 1 is sent to address 2060::20 about address 2062::20.

```
1w4d:Received ICMP unreachable code 1 from 2052::52
1w4d:Sending ICMP unreachable code 1 to 2060::20 about 2062::20
```

The table below lists the codes for ICMP unreachable messages.

Table 77: ICMP Unreachable Messages--Code Descriptions

Code	Description
0	The router has no route to the packet destination.
1	Although the router has a route to the packet destination, communication is administratively prohibited.
3	The address is unreachable.
4	The port is unreachable.

Related Commands

Command	Description
debug ipv6 nd	Displays debugging messages for IPv6 ICMP neighbor discovery transactions.

debug ipv6 inspect through debug local-ack state

debug ipv6 inspect

To display messages about Cisco IOS firewall events, use the **debug ipv6 inspect** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ipv6 inspect {**function-trace**| **object-creation**| **object-deletion**| **events**| **timers**| **protocol**| **detailed**}

no debug ipv6 inspect detailed

Syntax Description

function-trace	Displays messages about software functions called by the Cisco IOS firewall.
object-creation	Displays messages about software objects being created by the Cisco IOS firewall. Object creation corresponds to the beginning of Cisco IOS firewall-inspected sessions.
object-deletion	Displays messages about software objects being deleted by the Cisco IOS firewall. Object deletion corresponds to the closing of Cisco IOS firewall-inspected sessions.
events	Displays messages about Cisco IOS firewall software events, including information about Cisco IOS firewall packet processing.
timers	Displays messages about Cisco IOS firewall timer events such as when a Cisco IOS firewall idle timeout is reached.
protocol	Displays messages about Cisco IOS firewall-inspected protocol events, including details about the protocol's packets.
detailed	Use this form of the command in conjunction with other Cisco IOS firewall debugging commands. This causes detailed information to be displayed for all the other enabled Cisco IOS firewall debugging.

Command Default

None

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(2)T	This command was introduced
12.0(22)S	This command was integrated into Cisco IOS Release 12.0(22)S.
12.2(14)S	This command was integrated into Cisco IOS Release 12.2(14)S.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Command History

12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.

Examples

The following example enables the display of messages about Cisco IOS firewall events:

```
debug ipv6 inspect
```

Related Commands

Command	Description
ipv6 inspect audit-trail	Turns on CBAC audit trail messages, which are displayed on the console after each Cisco IOS firewall session closes.
ipv6 inspect name	Defines a set of ipv6 inspection rules.
show ipv6 inspect	Displays CBAC configuration and session information.

debug ipv6 mfib

To enable debugging output on the IPv6 Multicast Forwarding Information Base (MFIB), use the **debug ipv6 mfib** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ipv6 mfib [**vrf** *vrf-name*] [*group-name*| *group-address*] [**adjacency**| **db**| **fs**| **init**| **interface**| **mrib** [**detail**]| **nat**| **pak**| **platform**| **ppr**| **ps**| **signal**| **table**]

no debug ipv6 mfib

Syntax Description

vrf *vrf-name*	(Optional) Specifies a virtual routing and forwarding (VRF) configuration.
group-name | *group-address*	(Optional) IPv6 address, name, or interface of the multicast group as defined in the Domain Name System (DNS) hosts table.
adjacency	(Optional) Enables debugging output for adjacency management activity.
db	(Optional) Enables debugging output for route database management activity.
fs	(Optional) Enables debugging output for fast switching activity.
init	(Optional) Enables debugging output for initialization or deinitialization activity.
interface	(Optional) Enables debugging output for IPv6 MFIB interfaces.
mrib	(Optional) Enables debugging output for communication with the MRIB.
detail	(Optional) Enables detailed debugging output regarding the MRIB.
nat	(Optional) Enables debugging output for Network Address Translation (NAT) events associated with all tables.
pak	(Optional) Enables debugging output for packet forwarding activity.
platform	(Optional) Enables debugging output related to the hardware platform use of application program interfaces (APIs).

ppr	(Optional) Enables debugging output for packet preservation events.
ps	(Optional) Enables debugging output for process-level-only packet forwarding activity.
signal	(Optional) Enables debugging output for activity regarding MFIB data-driven signaling to routing protocols.
table	(Optional) Enables debugging output for IPv6 MFIB table activity.

Command Modes

Privileged EXEC

Release	Modification
12.3(2)T	This command was introduced.
12.2(18)S	This command was integrated into Cisco IOS Release 12.2(18)S.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.
Cisco IOS XE Release 2.1	This command was introduced on Cisco ASR 1000 series routers.
12.2(33)SRE	The **detail** keyword was added.
15.1(1)T	The **detail** keyword was added.
15.1(4)M	The **vrf** *vrf-name* keyword and argument were added.

Usage Guidelines

If no keywords are used, all IPbv6 MFIB activity debugging output is displayed.

Examples

The following example enables debugging output for adjacency management activity on the IPv6 MFIB:

```
Router# debug ipv6 mfib adjacency
```

debug ipv6 mld

To enable debugging on Multicast Listener Discovery (MLD) protocol activity, use the **debug ipv6 mld**command in privileged EXEC mode. To restore the default value, use the **no** form of this command.

debug ipv6 mld [*group-name*| *group-address*| *interface-type*]

no debug ipv6 mld [*group-name*| *group-address*| *interface-type*]

Cisco IOS Release 12.0(26)S

debug ipv6 mld [**group** *group-name*| *group-address*| **interface** *interface-type*]

no debug ipv6 mld [**group** *group-name*| *group-address*| **interface** *interface-type*]

Syntax Description

group-name | *group-address*or **group** *group-name* | *group-address*	(Optional) IPv6 address or name of the multicast group.
interface-type or **interface** *interface-type*	(Optional) Interface type. For more information, use the question mark (**?**) online help function.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(2)T	This command was introduced.
12.2(18)S	This command was integrated into Cisco IOS Release 12.2(18)S.
12.0(26)S	This command was integrated into Cisco IOS Release 12.0(26)S.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(25)SG	This command was integrated into Cisco IOS Release 12.2(25)SG.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.
Cisco IOS XE Release 2.1	This command was introduced on Cisco ASR 1000 Series Routers.

Usage Guidelines

This command helps discover whether the MLD protocol activities are working correctly. In general, if MLD is not working, the router process never discovers that there is a host on the network that is configured to receive multicast packets.

The messages displayed by the **debug ipv6 mld**command show query and report activity received from other routers and hosts. Use this command in conjunction with **debug ipv6 pim** to display additional multicast activity, to learn more information about the multicast routing process, or to learn why packets are forwarded out of particular interfaces.

Examples

The following example enables debugging on MLD protocol activity:

```
Router# debug ipv6 mld
```

Related Commands

Command	Description
debug ipv6 pim	Enables debugging on PIM protocol activity.

debug ipv6 mld explicit

To display information related to the explicit tracking of hosts, use the **debug ipv6 mld explicit**command in privileged EXEC mode. To disable debugging, use the **no** form of this command.

debug ipv6 mld explicit [*group-name*| *group-address*]

no debug ipv6 mld explicit [*group-name*| *group-address*]

Syntax Description

group-name | *group-address*	(Optional) IPv6 address or name of the multicast group.

Command Default

Debugging for the explicit tracking of hosts is not enabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.3(7)T	This command was introduced.
12.2(25)S	This command was integrated into Cisco IOS Release 12.2(25)S.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(25)SG	This command was integrated into Cisco IOS Release 12.2(25)SG.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.
Cisco IOS XE Release 2.1	This command was introduced on Cisco ASR 1000 Series Routers.

Usage Guidelines

When the optional *group-name* or *group-address* argument is not used, all debugging information is displayed.

Examples

The following example shows how to enable information to be displayed about the explicit tracking of hosts. The command output is self-explanatory:

```
Router# debug ipv6 mld explicit
00:00:56:MLD:ET host FE80::A8BB:CCFF:FE00:800 report for FF05::6 (0 srcs) on Ethernet1/0
00:00:56:MLD:ET host FE80::A8BB:CCFF:FE00:800 switch to exclude for FF05::6 on Ethernet1/0
00:00:56:MLD:ET MRIB modify for (*,FF05::6) on Ethernet1/0 new 100, mdf 100
```

debug ipv6 mld ssm-map

To display debug messages for Source Specific Multicast (SSM) mapping related to Multicast Listener Discovery (MLD), use the **debug ipv6 mld ssm-map**command in privileged EXEC mode. To disable debug messages for SSM mapping, use the **no** form of this command.

debug ipv6 mld ssm-map [*source-address*]

no debug ipv6 mld ssm-map [*source-address*]

Syntax Description

source-address	(Optional) Source address associated with an MLD membership for a group identified by the access list.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(18)SXE	This command was introduced.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

Consult Cisco technical support before using this command.

Examples

The following example allows debugging information for SSM mapping to be displayed:

```
Router# debug ipv6 mld ssm-map
```

Related Commands

Command	Description
ipv6 mld ssm-map enable	Enables the SSM mapping feature for groups in the configured SSM range
ipv6 mld ssm-map query dns	Enables DNS-based SSM mapping.
ipv6 mld ssm-map static	Configures static SSM mappings.
show ipv6 mld ssm-map	Displays SSM mapping information.

debug ipv6 mobile

To enable the display of debugging information for Mobile IPv6, use the **debug ipv6 mobile**command in privileged EXEC mode.

debug ipv6 mobile {**binding-cache**| **forwarding**| **home-agent**| **registration**}

Syntax Description

binding-cache	Events associated with the binding cache.
forwarding	Events associated with forwarding (tunneling) packets for which the router is acting as home agent.
home-agent	Events associated with the home agent, Dynamic Home Address Agent Discovery (DHAAD), Mobile prefix discovery (MPD), and generic home agent (HA) debugging and binding acknowledgments.
registration	Events associated with binding updates that are registrations.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(14)T	This command was introduced.

Usage Guidelines

The **debug ipv6 mobile** command enables the display of selected debugging information. You may use multiple command lines to enable concurrent debugging of multiple classes of information.

Examples

In the following example, debugging information is displayed for binding updates processing:

```
Router# debug ipv6 mobile registration
```

Related Commands

Command	Description
binding	Configures binding options for the Mobile IPv6 home agent feature in home-agent configuration mode.
ipv6 mobile home-agent (global configuration)	Enters home agent configuration mode.

Command	Description
ipv6 mobile home-agent (interface configuration)	Initializes and start the IPv6 Mobile home agent on a specific interface.
ipv6 mobile home-agent preference	Configures the home agent preference value on the interface.

debug ipv6 mobile mag

To debug the Mobile Access Gateway (MAG) application programming interface (API), information, or events, use the **debug ipv6 mobile mag** command in privileged EXEC mode. To disable display of the debugging output, use the **no** form of this command.

debug ipv6 mobile mag {**api**| **events**| **info**}

no debug ipv6 mobile mag {**api**| **events**| **info**}

Syntax Description

api	Enables API-specific debug events.
events	Enables all events occurring within the Local Mobility Anchor (LMA) and the MAG.
info	Provides debug information within the Proxy Mobile IPv6 (PMIPV6) module.

Command Default

Debugging is not enabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS XE Release 3.4S	This command was introduced.
15.2(4)M	This command was integrated into Cisco IOS Release 15.2(4)M.

Usage Guidelines

Use the **debug ipv6 mobile mag events** command to enable events occurring within the LMA and MAG. The following table lists the common causes for Proxy Binding Update (PBU) rejections:

PBU Reject Status	Description
PMIPV6_BA_ACCEPTED	The PBU is accepted.
GRE_KEY_OPTION_NOT_REQUIRED	The PBU is processed successfully but the GRE encapsulation and GRE keys are not required.
PMIPV6_BA_UNSPEC_FAIL	The PBU is rejected for an unspecified reason.
PMIPV6_BA_ADMIN_FAIL	The PBU is rejected due to administrative reasons.

PBU Reject Status	Description
PMIPV6_BA_RESOURCE_FAIL	The PBU is rejected due to insufficient resources.
PMIPV6_BA_HM_REG_FAIL	The PBU is rejected because it has an unsupported home registration.
PMIPV6_BA_HM_SUBNET_FAIL	The PBU is rejected because the current subnet is not the home subnet.
PMIPV6_BA_BAD_SEQ_FAIL	The PBU is rejected because the sequence number is out of the specified range.
PMIPV6_BA_CHANGE_FAIL	The PBU is rejected because the registration type has changed.
PMIPV6_BA_AUTH_FAIL	The PBU is rejected because the authorization has failed.
PROXY_REG_NOT_ENABLED	The PBU is rejected because the registration of the proxy is not enabled for the mobile node.
NOT_LMA_FOR_THIS_MOBILE_NODE	The PBU is rejected because the current Local Mobility Anchor (LMA) is not the appropriate LMA for the mobile node.
MAG_NOT_AUTHORIZED_FOR_ PROXY_REG	The PBU is rejected because the Mobile Access Gateway (MAG) is not authorized to send PBUs.
NOT_AUTHORIZED_FOR_HNP	The PBU is rejected because it is not authorized for the Home Network Prefix (HNP).
TIMESTAMP_MISMATCH	The PBU is rejected because it has an invalid timestamp value.
TIMESTAMP_LOWER_THAN_ PREV_ACCEPTED	This PBU is rejected because the timestamp value is lower than the previously accepted value.
MISSING_HNP_OPTION	The PBU is rejected because it is the Home Network Prefix (HNP) option.
BCE_PBU_PREFIX_SET_DO _NOT_MATCH	The PBU is rejected because the Home Network Prefixes (HNPs) that are received in the PBU do not match with the Binding Cache Entry (BCE).
MISSING_MN_IDENTIFIER_OPTION	The PBU is rejected because the mobile node identifier option is missing.
MISSING_HANDOFF_INDICATOR_ OPTION	The PBU is rejected because the Handoff Indicator is missing.

Examples

The following is sample output from the **debug ipv6 mobile mag api** command displays the APIs that are called during the call setup flow:

```
Device# debug ipv6 mobile mag api

07:52:08.051: MIP_PDL_API: pmipv6_pdl_get_att API Called
07:52:08.051: [PMIPV6_BINDING_API]: pmipv6_get_binding API called
07:52:08.051: [PMIPV6_BINDING_API]: pmipv6_get_binding API called
07:52:08.051: [PMIPV6_MAG_API]: mag_bul_do_state_transition API called
07:52:08.051: [PMIPV6_MAG_API]: pmipv6_mag_bul_null_state_hndlr API called
07:52:08.051: [PMIPV6_MAG_API]: pmipv6_mag_bul_null_state_exit API called
07:52:08.051: [PMIPV6_MAG_API]: pmipv6_mag_bul_init_state_entry API called
07:52:08.051: [PMIPV6_BINDING_API]: pmipv6_add_binding_entry API called
07:52:08.051: MIP_PDL_API: pmipv6_pdl_get_timestamp API Called
07:52:08.053: [PMIPV6_MAG_API]: pmipv6_mag_should_handle_pkt called
07:52:08.053: [PMIPV6_MAG_API]: pmipv6_mag_message_handler called
07:52:08.053: [PMIPV6_BINDING_API]: pmipv6_get_binding API called
07:52:08.053: [PMIPV6_BINDING_API]: pmipv6_get_binding API called
07:52:08.053: [PMIPV6_MAG_API]: mag_bul_do_state_transition API called
07:52:08.053: [PMIPV6_MAG_API]: pmipv6_mag_bul_init_state_hndlr API called
07:52:08.053: [PMIPV6_MAG_API]: pmipv6_mag_bul_init_state_exit API called
07:52:08.053: MIP_PDL_API: pmipv6_pdl_create_vintf API Called
16 07:52:08.054: MIP_PDL_API: pmipv6_pdl_set_ip4address API Called
16 07:52:08.054: MIP_PDL_API: pmipv6_pdl_set_macaddr API Called
16 07:52:08.054: MIP_PDL_API: mip_pdl_setupv4_route API Called
07:52:08.054: MIP_PDL_API: mip_pdl_setupv6_tunnel API Called
07:52:08.054: %LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel0, changed state to
down
07:52:08.054: MIP_PDL_API: mip_pdl_get_handle_for_tunnel API Called
07:52:08.054: MIP_PDL_API: mip_pdl_populate_rtunnel API Called
07:52:08.054: MIP_PDL_API: mip_pdl_get_handle_for_tunnel API Called
07:52:08.055: [PMIPV6_BINDING_API]: pmipv6_update_binding_key API called
07:52:08.055: [PMIPV6_MAG_API]:  pmipv6_mag_bul_active_state_entry API called
```

The following is sample output from the **debug ipv6 mobile mag events** command:

```
Device# debug ipv6 mobile mag events

PMIPv6 MAG Event debug is turned on
```

The following line shows that the DHCP Discover trigger is received from the mobile node (MN):

```
07:48:31.638: [PMIPV6_MAG_EVENT]: Trigger request received (DHCP Discover trigger) from
(example3@example.com)
```

The following line shows the MAG machine state change. A new MN attaches to the MAG and the state changes from NULL to INIT:

```
07:48:31.638: [PMIPV6_MAG_EVENT]: Event received New MN intf attached in state: NULL, new
state: INIT
```

The following line shows that the Proxy Binding Update (PBU) message is sent from a MAG to an MN:

```
07:48:31.638: [PMIPV6_MAG_EVENT]: PBU message sent
```

The following lines show that the Proxy Binding Acknowledgment (PBA) is received from the LMA for the MN. The incoming parameters are link layer identifier (lli) length, value, and access technology type (att). The status 0 indicates success.

```
07:48:31.639: [PMIPV6_MAG_EVENT]: message received: PBA
07:48:31.639: [PMIPV6_MAG_EVENT]: PBA: nai(example3@example.com),nai len: 14, lli
(aabb.cc00.ce00), ll len: 16, att:3, status:0
```

The following line shows that the refresh timer has started:

```
07:48:31.639: [PMIPV6_MAG_EVENT]: Starting Refresh timer, period (300000)
```

The following lines show that a v4 route is added to the MN, which has a new address assigned. A new v6 tunnel is created and a reverse tunnel entry is added for the MN.

```
07:48:31.640: [PMIPV6_MAG_EVENT]: Adding V4 route, address (0x11110103), Prefix len (24),
handle: (GigabitEthernet0/0/0)
!
07:48:31.640: [PMIPV6_MAG_EVENT]: Adding V6 Tunnel, Handle (Tunnel1), mode: (IPV6_IN_IPV6)
07:48:31.641: [PMIPV6_MAG_EVENT]: Populating  Reverse V4 Tunnel entry, l2 address
(0xaabb.cc00.ce00), ipv4 add: 0x11110103 phy handle: (GigabitEthernet0/0/0)
```

The following is sample out from **debug ipv6 mobile mag info** command:

```
Device# debug ipv6 mobile mag info
```

```
PMIPv6 MAG INFO debug is turned on
```

The following lines show that the new binding is created and added to the AV tree:

```
07:50:31.714: [PMIPV6_PDB_INFO]: MN entry example3@example.com found in hashset
07:50:31.714: [PMIPV6_BINDING_INFO]: binding added New NAI AVL node created
```

The following line provides more information about the PBUs that are sent:

```
07:50:31.714: [PMIPV6_MAG_INFO]: PBU message nai(example3@example.com), nai len: 14, hoa(0),
 att(3) llid(aabb.cc00.ce00) , ll len: 16
```

The following line shows that a binding for the MN using the Network Access Identifier (NAI) example3@example.com is found:

```
07:50:31.717: [PMIPV6_BINDING_INFO_KEY]: Keytype as NAI. NAI: example3@example.com
07:50:31.717: [PMIPV6_BINDING_INFO]: binding found on NAI tree
```

The following line shows that a virtual interface is created in the MAG and assigned the MAC address aaaa.aaaa.aaaa:

```
07:50:31.717: [PMIPV6_MAG_EVENT]: Creating virtual interface handle (IFNAME_PMIP_VIF4)
07:50:31.717: [PMIPV6_MAG_INFO]: Setting Mac Address (aaaa.aaaa.aaaa) on (IFNAME_PMIP_VIF4)
The following line shows that a route for the MN is added in the MAG:
07:50:31.717: MIP_PDL_INFO: Successfully added route 10.10.1.4/24 to GigabitEthernet0/0/0
07:50:31.717: MIP_PDL_INFO: Route via: GigabitEthernet0/1/0 (IPv6)
The following line shows that a tunnel is created with a source address and a destination
address:
07:50:31.718: MIP_PDL_INFO: Tunnel0 (IPv6) created with src 2000::4 dst 2001::2
07:50:31.718: MIP_PDL_INFO: Rev. Tunnel acl entry added for subnet (10.10.0.0)
```

Related Commands

Command	Description
ipv6 mobile pmipv6-mag	Configures the MAG for the PMIPV6 domain.

debug ipv6 mobile networks

To display debugging messages for IPv6 mobile networks, use the **debug ipv6 mobile networks**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ipv6 mobile networks

no debug ipv6 mobile networks

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.4(20)T	This command was introduced.

Usage Guidelines

The **debug ipv6 mobile networks**command enables the display of selected debugging information.

Examples

The following example shows how to enable the display of debugging messages for IPv6 mobile networks:

```
Router# debug ipv6 mobile networks
```

Related Commands

Command	Description
ipv6 mobile router	Enables IPv6 NEMO functionality on a router and places the router in IPv6 mobile router configuration mode.

debug ipv6 mobile packets

To debug the proxy mobile IPv4 or IPv6 packets, use the **debug ipv6 mobile packets** command in privileged EXEC mode. To disable the debugging output, use the **no** form of this command.

debug ipv6 mobile packets

no debug ipv6 mobile packets

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging is not enabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS XE Release 3.4S	This command was introduced.
15.2(4)M	This command was integrated into Cisco IOS Releases 15.2(4)M.

Examples

The following is sample output from the **debug ipv6 mobile packets** command:

```
Device# debug ipv6 mobile packets
```

```
PMIPv6 PKT debug is turned on
```

The following lines show the newly allocated packet size and the inner packet details:

```
07:51:17.693: [PMIPv6-MM]:Allocated packet of size 164 with tlv length 84
07:51:17.693: [PMIPV6_MM] Sending UDP Packet, src: 0x2020202, dst: 0x6060602, sport: 5436,
 dport:5436
```

The following lines shows the mobility options, the value, and the length:

```
07:51:17.693: [PMIPV6_MM] NAI option included len 14
!
2A986107E0:                4D 4E334063 6973636F         example3@example
2A986107F0: 2E636F6D 1702                         .com..
07:51:17.693:
07:51:17.693: [PMIPV6_MM] HI option included len 2 val 4
07:51:17.694: [PMIPV6_MM] ATT option included len 2 val 3
07:51:17.694: [PMIPV6_MM] TIMESTAMP option included len 8 value 3517199477
07:51:17.694: [PMIPV6_MM] LLI option included len 16
!
2A98610810: 61616262 2E636330 302E6365 30300100  aabb.cc00.ce00..
2A98610820: 24                                   $
07:51:17.694:
07:51:17.694: [PMIPV6_MM] V4HOAREQ option included len 6 val 0.0.0.0
07:51:17.694: [PMIPV6_MM] V4DFT_RTR option included len 6 val 0.0.0.0
07:51:17.694: **** Dumping the TLVs ****
```

```
!
2A986107E0: 01020000 080E014D 4E334063 6973636F  .......example3@example
2A986107F0: 2E636F6D 17020004 18020003 01001B08  .com............
2A98610800: 00000000 D1A43475 01020000 19100000  ....Q$4u........
2A98610810: 61616262 2E636330 302E6365 30300100  aabb.cc00.ce00..
2A98610820: 24060000 00000000 26060000 00000000  $.......&.......
2A98610830: 01020000                             ....
07:51:17.694:
07:51:17.695: [PMIPV6_MM] NAI option received len 14
!
2A97DBE560:       4D 4E334063 6973636F 2E636F6D     example3@example.com
2A97DBE570: 0017                                 ..
07:51:17.696:
07:51:17.696: [PMIPV6_MM] HI option received len 2 val 4
07:51:17.696: [PMIPV6_MM] ATT option received len 2 val 3
07:51:17.696: [PMIPV6_MM] TIMESTAMP option received len 8 value 3517199477
07:51:17.696: [PMIPV6_MM] LLI option received len 16
!
2A97DBE580:                            61616262              aabb
2A97DBE590: 2E636330 302E6365 30300100 00        .cc00.ce00...
07:51:17.696:
07:51:17.696: [PMIPV6_MM] V4HOAREPLY option received len 6 val 10.10.1.5
07:51:17.696: [PMIPV6_MM] V4DFT_RTR option received len 6 val 10.10.1.1
```

The following lines show the dump of the packet with all the Type Length Values (TLVs):

```
07:51:17.696: **** Dumping the TLVs ****
!
2A97DBE550:                            01020000              ....
2A97DBE560: 080E014D 4E334063 6973636F 2E636F6D  ...example3@example.com
2A97DBE570: 00170200 04180200 03001B08 00000000  ................
2A97DBE580: D1A43475 01020000 19100000 61616262  Q$4u........aabb
2A97DBE590: 2E636330 302E6365 30300100 00000000  .cc00.ce00......
2A97DBE5A0: 00000000 00000000 00000000 00000000  ................
2A97DBE5B0: 25060060 11110105 26060000 11110101  %..`....&.......
2A97DBE5C0:
07:51:17.696:
```

Related Commands

Command	Description
ipv6 mobile pmipv6-mag	Configures the MAG for the PMIPv6 domain.

debug ipv6 mobile router

To display debugging messages for the IPv6 mobile router, use the **debug ipv6 mobile router** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ipv6 mobile router [detail]

no debug ipv6 mobile router

Syntax Description

detail	(Optional) Displays detailed mobile router debug messages.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.4(20)T	This command was introduced.

Usage Guidelines

The IPv6 mobile router operations can be debugged. The following conditions trigger debugging messages:

- Agent discovery
- Registration
- Mobile router state change
- Routes and tunnels created or deleted
- Roaming information

Debugging messages are prefixed with "MobRtr," and detail messages are prefixed with "MobRtrX."

Examples

The following example shows how to enable the display of debugging messages for the IPv6 mobile router:

```
Router# debug ipv6 mobile router
```

Related Commands

Command	Description
ipv6 mobile router	Enables IPv6 NEMO functionality on a router and places the router in IPv6 mobile router configuration mode.

debug ipv6 mrib client

To enable debugging on Multicast Routing Information Base (MRIB) client management activity, use the **debug ipv6 mrib client**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ipv6 mrib [**vrf** *vrf-name*] **client**

no debug ipv6 mrib client

Syntax Description

vrf *vrf-name*	(Optional) Specifies a virtual routing and forwarding (VRF) configuration.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(2)T	This command was introduced.
12.2(18)S	This command was integrated into Cisco IOS Release 12.2(18)S.
12.0(26)S	This command was integrated into Cisco IOS Release 12.0(26)S.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(25)SG	This command was integrated into Cisco IOS Release 12.2(25)SG.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.
Cisco IOS XE Release 2.1	This command was introduced on Cisco ASR 1000 Series Routers.
15.1(4)M	The **vrf** *vrf-name* keyword and argument were added.

Usage Guidelines

The **debug ipv6 mrib client** command is used to display the activity in the MRIB associated with clients such as Protocol Independent Multicast (PIM) and Multicast Listener Discovery (MLD). If you are having difficulty with your client connections, use this command to display new clients being added and deleted.

The **debug ipv6 mrib client** command also displays information on when a new client is added to or deleted from the MRIB, when a client connection is established or torn down, when a client binds to a particular MRIB table, and when a client is informed that there are updates to be read.

Examples

The following example enables debugging on MRIB client management activity:

```
Router# debug ipv6 mrib client
```

Related Commands

Command	Description
debug ipv6 mrib route	Displays MRIB routing entry-related activity.

debug ipv6 mrib io

To enable debugging on Multicast Routing Information Base (MRIB) I/O events, use the **debug ipv6 mrib io**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ipv6 mrib [**vrf** *vrf-name*] **io**

no debug ipv6 mrib io

Syntax Description

vrf *vrf-name*	(Optional) Specifies a virtual routing and forwarding (VRF) configuration.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(2)T	This command was introduced.
12.2(18)S	This command was integrated into Cisco IOS Release 12.2(18)S.
12.0(26)S	This command was integrated into Cisco IOS Release 12.0(26)S.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(25)SG	This command was integrated into Cisco IOS Release 12.2(25)SG.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.
Cisco IOS XE Release 2.1	This command was introduced on Cisco ASR 1000 Series Routers.
15.1(4)M	The **vrf** *vrf-name* keyword and argument were added.

Usage Guidelines

Use the **debug ipv6 mrib io** command to display information on when clients open and close MRIB I/O connections, when MRIB entry and interface updates are received and processed from clients, and when MRIB entry and interface updates are sent to clients.

Examples

The following example enables debugging on MRIB I/O events:

```
Router# debug ipv6 mrib io
```

debug ipv6 mrib proxy

To enable debugging on multicast routing information base (MRIB) proxy activity between the route processor and line cards on distributed router platforms, use the **debug ipv6 mrib proxy**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ipv6 mrib proxy

no debug ipv6 mrib proxy

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(26)S	This command was introduced.
12.3(4)T	This command was integrated into Cisco IOS Release 12.3(4)T.
12.2(25)S	This command was integrated into Cisco IOS Release 12.2(25)S.
12.2(25)SG	This command was integrated into Cisco IOS Release 12.2(25)SG.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
Cisco IOS XE Release 2.1	This command was introduced on Cisco ASR 1000 Series Routers.

Usage Guidelines

Use the **debug ipv6 mrib proxy** command to display information on connections that are being opened and closed and on MRIB transaction messages that are being passed between the route processor and line cards.

Examples

The following example enables debugging on MRIB proxy events:

```
Router# debug ipv6 mrib proxy
```

debug ipv6 mrib route

To display information about Multicast Routing Information Base (MRIB) routing entry-related activity, use the **debug ipv6 mrib route**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ipv6 mrib [**vrf** *vrf-name*] **route** [*group-name*| *group-address*]

no debug ipv6 mrib route

Syntax Description

vrf *vrf-name*	(Optional) Specifies a virtual routing and forwarding (VRF) configuration.
group-name | *group-address*	(Optional) IPv6 address or name of the multicast group.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(2)T	This command was introduced.
12.2(18)S	This command was integrated into Cisco IOS Release 12.2(18)S.
12.0(26)S	This command was integrated into Cisco IOS Release 12.0(26)S.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(25)SG	This command was integrated into Cisco IOS Release 12.2(25)SG.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.
Cisco IOS XE Release 2.1	This command was introduced on Cisco ASR 1000 Series Routers.
15.1(4)M	The **vrf** *vrf-name* keyword and argument were added.

Usage Guidelines

This command displays update information related to the route database made by MRIB clients, which is then redistributed to the clients.

Use this command to monitor MRIB route activity when discontinuity is found between the MRIB and the client database or between the individual client databases.

Examples

The following example enables the display of information about MRIB routing entry-related activity:

```
Router# debug ipv6 mrib route
```

Related Commands

Command	Description
show ipv6 mrib client	Displays information about the MRIB client management activity.

debug ipv6 mrib table

To enable debugging on Multicast Routing Information Base (MRIB) table management activity, use the **debug ipv6 mrib table**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ipv6 mrib [**vrf** *vrf-name*] **table**

no debug ipv6 mrib table

Syntax Description

vrf *vrf-name*	(Optional) Specifies a virtual routing and forwarding (VRF) configuration.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(2)T	This command was introduced.
12.2(18)S	This command was integrated into Cisco IOS Release 12.2(18)S.
12.0(26)S	This command was integrated into Cisco IOS Release 12.0(26)S.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(25)SG	This command was integrated into Cisco IOS Release 12.2(25)SG.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.
Cisco IOS XE Release 2.1	This command was introduced on Cisco ASR 1000 Series Routers.
15.1(4)M	The **vrf** *vrf-name* keyword and argument were added.

Usage Guidelines

Use the **debug ipv6 mrib table** command to display information on new MRIB tables being added and deleted.

Examples

The following example enables debugging on MRIB table management activity:

```
Router# debug ipv6 mrib table
```

debug ipv6 multicast aaa

To enable debugging of authentication, authorization, and accounting (AAA) events related to IPv6 multicast routing, use the **debug ipv6 multicast aaa** command in privileged EXEC mode. To disable debugging of events, use the **no** form of this command.

debug ipv6 multicast aaa {**detail** | **error** | **verbose**}

no debug ipv6 multicast aaa {**detail** | **error** | **verbose**}

Syntax Description

aaa	Enables debugging of IPv6 AAA events.
detail	Enables debugging of IPv6 multicast AAA details.
error	Enables debugging of IPv6 multicast AAA errors.
verbose	Enables debugging of IPv6 multicast AAA verbose.

Command Modes

Privileged EXEC(#)

Command History

Release	Modification
15.3(1)S	This command was introduced.

Usage Guidelines

You must configure multicast routing in an IPv6 environment. Use the **ipv6 multicast-routing** command in global configuration mode to enable IPv6 multicast routing. The **ipv6 multicast-routing** command applies on all IPv6-enabled interfaces on a device, which are then automatically enabled for Protocol-Independent Multicast version 6 (PIMv6). PIM is used between devices so that the devices can track which multicast packets to forward to each other and to the devices that are on the directly connected LANs.

Examples

The following example shows how to enable debugging of IPv6 multicast AAA information:

```
Device# debug ipv6 multicast aaa detail
AAA details debugging is on
Device# debug ipv6 multicast aaa error
AAA errors debugging is on
Device# debug ipv6 multicast aaa verbose
AAA verbose debugging is on
```

Related Commands

Command	Description
ipv6 multicast-routing	Enables multicast routing using MLD on all IPv6-enabled interfaces of the device and enables multicast forwarding.

debug ipv6 multicast rpf

To enable debugging of Reverse Path Forwarding (RPF) events related to IPv6 multicast routing, use the **debug ipv6 multicast rpf** command in privileged EXEC mode. To disable debugging of events, use the **no** form of this command.

debug ipv6 multicast rpf

no debug ipv6 multicast rpf

Syntax Description

rpf	Enables debugging of IPv6 multicast RPF events.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
15.3(1)S	This command was introduced.

Usage Guidelines

You must configure multicast routing in an IPv6 environment. Use the **ipv6 multicast-routing** command in global configuration mode to enable IPv6 multicast routing. The **ipv6 multicast-routing** command applies on all IPv6-enabled interfaces on a device, which are then automatically enabled for Protocol-Independent Multicast version 6 (PIMv6). PIM is used between devices so that the devices can track which multicast packets to forward to each other and to the devices that are on the directly connected LANs.

Examples

The following example shows how to enable debugging of IPv6 multicast RPF events:

```
Device# debug ipv6 multicast rpf
IPv6 Multicast RPF debugging is on
```

Related Commands

Command	Description
ipv6 multicast-routing	Enables multicast routing using MLD on all IPv6-enabled interfaces of the device and enables multicast forwarding.

debug ipv6 multicast rwatch

To enable debugging of route watch tracking events related to IPv6 multicast routing, use the **debug ipv6 multicast rwatch** command in privileged EXEC mode. To disable debugging of events, use the **no** form of this command.

debug ipv6 multicast rwatch

no debug ipv6 multicast rwatch

Syntax Description

rwatch	Enables debugging of IPv6 multicast route watch tracking events.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
15.3(1)S	This command was introduced.

Usage Guidelines

You must configure multicast routing in an IPv6 environment. Use the **ipv6 multicast-routing** command in global configuration mode to enable IPv6 multicast routing. The **ipv6 multicast-routing** command applies on all IPv6-enabled interfaces on a device, which are then automatically enabled for Protocol-Independent Multicast version 6 (PIMv6). PIM is used between devices so that the devices can track which multicast packets to forward to each other and to the devices that are on the directly connected LANs.

Examples

The following example shows how to enable debugging of IPv6 multicast route watch tracking events:

```
Device# debug ipv6 multicast rwatch
IPv6 Route-watch debugging is on
```

Related Commands

Command	Description
ipv6 multicast-routing	Enables multicast routing using MLD on all IPv6-enabled interfaces of the device and enables multicast forwarding.

debug ipv6 nat

To display debug messages for Network Address Translation--Protocol Translation (NAT-PT) translation events, use the **debug ipv6 nat**command in privileged EXEC mode. To disable debug messages for NAT-PT translation events, use the **no** form of this command.

debug ipv6 nat [**detailed**| **port**]

no debug ipv6 nat [**detailed**| **port**]

Syntax Description

detailed	(Optional) Displays detailed information about NAT-PT translation events.
port	(Optional) Displays port allocation events.

Command Default

Debugging for NAT-PT translation events is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(13)T	This command was introduced.
12.3(2)T	The **port** keyword was added to support Port Address Translation (PAT), or overload, multiplexing multiple IPv6 addresses to a single IPv4 address or to an IPv4 address pool.

Usage Guidelines

The **debug ipv6 nat**command can be used to troubleshoot NAT-PT translation issues. If no keywords are specified, debugging messages for all NAT-PT protocol translation events are displayed.

Note By default, the network server sends the output from **debug** commands and system error messages to the console. To redirect debugging output, use the logging command options within global configuration mode. Destinations are the console, virtual terminals, internal buffer, and UNIX hosts running a syslog server.

Caution Because the **debug ipv6 nat** command generates a substantial amount of output, use it only when traffic on the IPv6 network is low, so other activity on the system is not adversely affected.

Examples

The following example shows output for the **debug ipv6 nat**command:

```
Router# debug ipv6 nat
00:06:06: IPv6 NAT: icmp src (3002::8) -> (192.168.124.8), dst (2001::2) -> (192.168.123.2)

00:06:06: IPv6 NAT: icmp src (192.168.123.2) -> (2001::2), dst (192.168.124.8) -> (3002::8)

00:06:06: IPv6 NAT: icmp src (3002::8) -> (192.168.124.8), dst (2001::2) -> (192.168.123.2)

00:06:06: IPv6 NAT: icmp src (192.168.123.2) -> (2001::2), dst (192.168.124.8) -> (3002::8)

00:06:06: IPv6 NAT: tcp src (3002::8) -> (192.168.124.8), dst (2001::2) -> (192.168.123.2)

00:06:06: IPv6 NAT: tcp src (192.168.123.2) -> (2001::2), dst (192.168.124.8) -> (3002::8)

00:06:06: IPv6 NAT: tcp src (3002::8) -> (192.168.124.8), dst (2001::2) -> (192.168.123.2)

00:06:06: IPv6 NAT: tcp src (3002::8) -> (192.168.124.8), dst (2001::2) -> (192.168.123.2)

00:06:06: IPv6 NAT: tcp src (3002::8) -> (192.168.124.8), dst (2001::2) -> (192.168.123.2)

00:06:06: IPv6 NAT: tcp src (192.168.123.2) -> (2001::2), dst (192.168.124.8) -> (3002::8)
```

The table below describes the significant fields shown in the display.

Table 78: debug ipv6 nat Field Descriptions

Field	Description
IPv6 NAT:	Indicates that this is a NAT-PT packet.
icmp	Protocol of the packet being translated.
src (3000::8) -> (192.168.124.8)	The source IPv6 address and the NAT-PT mapped IPv4 address. **Note** If mapping IPv4 hosts to IPv6 hosts the first address would be an IPv4 address, and the second address an IPv6 address.
dst (2001::2) -> (192.168.123.2)	The destination IPv6 address and the NAT-PT mapped IPv4 address. **Note** If mapping IPv4 hosts to IPv6 hosts the first address would be an IPv4 address, and the second address an IPv6 address.

The following example shows output for the **debug ipv6 nat**command with the **detailed** keyword:

```
Router# debug ipv6 nat detailed
00:14:12: IPv6 NAT: address allocated 192.168.124.8
00:14:16: IPv6 NAT: deleted a NAT entry after timeout
```

debug ipv6 nd

To display debug messages for IPv6 Internet Control Message Protocol (ICMP) neighbor discovery transactions, use the **debug ipv6 nd**command in privileged EXEC mode. To disable debug messages for IPv6 ICMP neighbor discovery transactions, use the **no** form of this command.

debug ipv6 nd

no debug ipv6 nd

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging for IPv6 ICMP neighbor discovery is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(2)T	This command was introduced.
12.2(4)T	The DAD: <*nnnn* ::*nn* :> is unique, DAD: duplicate link-local <*nnnn* ::*nn* :> on <*interface type* >, interface stalled, and Received NA for <*nnnn* ::*nn* :> on <*interface type* > from <*nnnn* ::*nn* :> fields were added to the command output.
12.0(21)ST	This command was integrated into Cisco IOS Release 12.0(21)ST.
12.0(22)S	This command was integrated into Cisco IOS Release 12.0(22)S.
12.2(14)S	This command was integrated into Cisco IOS Release 12.2(14)S.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(25)SG	This command was integrated into Cisco IOS Release 12.2(25)SG.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.

Usage Guidelines

This command can help determine whether the router is sending or receiving IPv6 ICMP neighbor discovery messages.

Note

By default, the network server sends the output from debug commands and system error messages to the console. To redirect debug output, use the logging command options within global configuration mode. Destinations include the console, virtual terminals, internal buffer, and UNIX hosts running a syslog server. For complete information on debug commands and redirecting debug output, refer to the *Cisco IOS Debug Command Reference* .

Examples

The following example shows output for the **debug ipv6 nd**command:

```
Router# debug ipv6 nd
13:22:40:ICMPv6-ND:STALE -> DELAY:2000:0:0:3::2
13:22:45:ICMPv6-ND:DELAY -> PROBE:2000:0:0:3::2
13:22:45:ICMPv6-ND:Sending NS for 2000:0:0:3::2 on FastEthernet0/0
13:22:45:ICMPv6-ND:Received NA for 2000:0:0:3::2 on FastEthernet0/0 from 2000:0:0:3::2
13:22:45:ICMPv6-ND:PROBE -> REACH:2000:0:0:3::2
13:22:45:ICMPv6-ND:Received NS for 2000:0:0:3::1 on FastEthernet0/0 from
FE80::203:A0FF:FED6:1400
13:22:45:ICMPv6-ND:Sending NA for 2000:0:0:3::1 on FastEthernet0/0
13:23:15: ICMPv6-ND: Sending NS for FE80::1 on Ethernet0/1
13:23:16: ICMPv6-ND: DAD: FE80::1 is unique.
13:23:16: ICMPv6-ND: Sending NS for 2000::2 on Ethernet0/1
13:23:16: ICMPv6-ND: Sending NS for 3000::3 on Ethernet0/1
13:23:16: ICMPv6-ND: Sending NA for FE80::1 on Ethernet0/1
13:23:17: ICMPv6-ND: DAD: 2000::2 is unique.
13:23:53: ICMPv6-ND: Sending NA for 2000::2 on Ethernet0/1
13:23:53: ICMPv6-ND: DAD: 3000::3 is unique.
13:23:53: ICMPv6-ND: Sending NA for 3000::3 on Ethernet0/1
3d19h: ICMPv6-ND: Sending NS for FE80::2 on Ethernet0/2
3d19h: ICMPv6-ND: Received NA for FE80::2 on Ethernet0/2 from FE80::2
3d19h: ICMPv6-ND: DAD: duplicate link-local FE80::2 on Ethernet0/2,interface stalled
3d19h: %IPV6-4-DUPLICATE: Duplicate address FE80::2 on Ethernet0/2
3d19h: ICMPv6-ND: Sending NS for 3000::4 on Ethernet0/3
3d19h: ICMPv6-ND: Received NA for 3000::4 on Ethernet0/3 from 3000::4
3d19h: %IPV6-4-DUPLICATE: Duplicate address 3000::4 on Ethernet0/3
```

The table below describes the significant fields shown in the display.

Table 79: debug ipv6 nd Field Descriptions

Field	Description
13:22:40:	Indicates the time (hours:minutes:seconds) at which the ICMP neighbor discovery event occrred.
ICMPv6-ND	Indicates that a state change is occurring for an entry in the IPv6 neighbors cache.
STALE	Stale state. This state of an neighbor discovery cache entry used to be "reachable," but is now is "stale" due to the entry not being used. In order to use this address, the router must go through the neighbor discovery process in order to confirm reachability.

Field	Description
DELAY	Delayed state. Reachability for this ND cache entry is currently being reconfirmed. While in the delay state, upper-layer protocols may inform IPv6 that they have confirmed reachability to the entry. Therefore, there is no need to send a neighbor solicitation for the entry.
PROBE	Probe state. While in the probe state, if no confirmation is received from the upper-layer protocols about the reachability of the entry, a neighbor solicitation message is sent. The entry remains in the "probe" state until a neighbor advertisement message is received in response to the neighbor solicitation message.
Sending NS for...	Sending a neighbor solicitation message. In the example output, a neighbor solicitation message is sent on Fast Ethernet interface 0/0 to determine the link-layer address of 2000:0:0:3::2 on Fast Ethernet interface 0/0.
Received NA for...	Received a neighbor advertisement message. In the example output, a neighbor advertisement message is received from the address 2000:0:0:3::2 (the second address) that includes the link-layer address of 2000:0:0:3::2 (first address) from Ethernet interface 0/0.
REACH	Reachable state. An ND cache entry in this state is considered reachable, and the corresponding link-layer address can be used without needing to perform neighbor discovery on the address.
Received NS for...	Received neighbor solicitations. In the example output, the address FE80::203:A0FF:FED6:1400 (on Fast Ethernet interface 0/0) is trying to determine the link-local address of 2000:0:0:3::1.
Sending NA for...	Sending for neighbor advertisements. In the example output, a neighbor advertisement containing the link-layer address of 2000:0:0:3::1 (an address assigned to the Fast Ethernet interface 0/0 address) was sent.
DAD: FE80::1 is unique.	Duplicate address detection processing was performed on the unicast IPv6 address (a neighbor solicitation message was not received in response to a neighbor advertisement message that contained the unicast IPv6 address) and the address is unique.

Field	Description
3d19h:	Indicates time (days, hours) since the last reboot of the event occurring; 3d19h: indicates the time (since the last reboot) of the event occurring was 3 days and 19 hours ago.
DAD: duplicate link-local FE80::2 on Ethernet0/2, interface stalled	Duplicate address detection processing was performed on the link-local IPv6 address (the link-local address FE80::2 is used in the example). A neighbor advertisement message was received in response to a neighbor solicitation message that contained the link-local IPv6 address. The address is not unique, and the processing of IPv6 packets is disabled on the interface.
%IPV6-4-DUPLICATE: Duplicate address...	System error message indicating the duplicate address.
Received NA for 3000::4 on Ethernet0/3 from 3000::4	Duplicate address detection processing was performed on the global IPv6 address (the global address 3000::4 is used in the example). A neighbor advertisement message was received in response to a neighbor solicitation message that contained the global IPv6 address. The address is not unique and is not used.

Related Commands

Command	Description
debug ipv6 icmp	Displays debug messages for IPv6 ICMP transactions.
show ipv6 neighbors	Displays IPv6 neighbor discovery cache information.

debug ipv6 ospf

To display debugging information for Open Shortest Path First (OSPF) for IPv6, use the **debug ipv6 ospf** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ipv6 ospf [**adj**| **ipsec**| **database-timer**| **flood**| **hello**| **lsa-generation**| **retransmission**]

no debug ipv6 ospf [**adj**| **ipsec**| **database-timer**| **flood**| **hello**| **lsa-generation**| **retransmission**]

Syntax Description

adj	(Optional) Displays adjacency information.
ipsec	(Optional) Displays the interaction between OSPF and IPSec in IPv6 networks, including creation and removal of policy definitions.
database-timer	(Optional) Displays database-timer information.
flood	(Optional) Displays flooding information.
hello	(Optional) Displays hello packet information.
l2api	(Optional) Enables layer 2 and layer 3 application program interface (API) debugging.
lsa-generation	(Optional) Displays link-state advertisement (LSA) generation information for all LSA types.
retransmission	(Optional) Displays retransmission information.

Command Default

Debugging of OSPF for IPv6 is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(24)S	This command was introduced.
12.2(15)T	This command was integrated in Cisco IOS Release 12.2(15)T.
12.2(18)S	This command was integrated in Cisco IOS Release 12.2(18)S.
12.3(4)T	The **ipsec** keyword was added to support OSPF for IPv6 authentication for IPSec.

Release	Modification
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.
12.4(25)T	The **l2api** keyword was added.

Usage Guidelines

Consult Cisco technical support before using this command.

Examples

The following example displays adjacency information for OSPF for IPv6:

```
Router# debug ipv6 ospf adj
```

debug ipv6 ospf database-timer rate-limit

To display debugging information about the current wait-time used for SPF scheduling, use the **debug ipv6 ospf database-timer rate-limit**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ipv6 ospf database-timer rate-limit [*acl-number*]

no debug ipv6 ospf database-timer rate-limit

Syntax Description

acl-number	(Optional) Access list number.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(33)SRC	This command was introduced.
12.2(33)SB	This command was integrated into Cisco IOS Release 12.2(33)SB.

Usage Guidelines

Consult Cisco technical support before using this command.

Examples

The following example shows how to turn on debugging for SPF scheduling:

```
Router# debug ipv6 ospf database-timer rate-limit
```

debug ipv6 ospf events

To display information on Open Shortest Path First (OSPF)-related events, such as designated router selection and shortest path first (SPF) calculation, use the **debug ipv6 ospf events** command in privileged EXEC com mand. To disable debugging output, use the **no** form of this command.

debug ipv6 ospf events

no debug ipv6 ospf events

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(24)S	This command was introduced.
12.2(15)T	This command was integrated into Cisco IOS Release 12.2(15)T.
12.2(18)S	This command was integrated into Cisco IOS Release 12.2(18)S.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.

Command History

12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.
12.2(33)XNE	This command was modified. It was integrated into Cisco IOS Release 12.2(33)XNE.

Usage Guidelines

Consult Cisco technical support before using this command.

Examples

The following example displays information on OSPF-related events:

```
Router#
debug ipv6 ospf events
```

debug ipv6 ospf graceful-restart

To enable debugging for IPv6 graceful-restart-related events, use the **debug ipv6 ospf graceful-restart**command in privileged EXEC mode.

debug ipv6 ospf graceful-restart

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging is not enabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS XE Release 2.1	This command was introduced.
15.0(1)M	This command was integrated into Cisco IOS Release 15.0(1)M.
12.2(33)SRE	This command was modified. It was integrated into Cisco IOS Release 12.2(33)SRE.

Usage Guidelines

The **debug ipv6 ospf graceful-restart** command helps troubleshoot graceful-restart-related events on both graceful-restart-capable and graceful-restart-aware routers.

Examples

The following example enables debugging for graceful-restart-related events:

```
Router# debug ipv6 ospf graceful-restart
00:03:41: OSPFv3: GR timer started for ospf process 1 for 120 secs,
00:03:43: OSPFv3: GR  Build Grace LSA for interface Ethernet0/0
00:03:43: OSPFv3: GR Flood grace lsa on Ethernet0/0
00:03:43: OSPFv3: GR complete check for area 0 process 1
00:03:43: OSPFv3: GR wait, Ethernet0/0 in area 0 not yet complete
00:03:45: OSPFv3: GR Re-flood Grace LSA on Ethernet0/0
00:04:01: OSPFv3: GR  initial wait expired
00:04:01: OSPFv3: GR complete check for area 0 process 1
00:04:01: OSPFv3: GR wait, Ethernet0/0 in area 0 not yet complete
00:04:07: OSPFv3: GR complete check for area 0 process 1
00:04:07: OSPFv3: GR re-sync completed in area 0, process 1
00:04:07: OSPFv3: GR complete check for process 1
00:04:07: OSPFv3: process 1: GR re-sync completed for all neighbors
00:04:07: OSPFv3: scheduling rtr lsa for area 0 process 1
00:04:07: OSPFv3: Post GR, flood maxaged grace-LSA on Ethernet0/0
```

Related Commands

Command	Description
graceful-restart	Enables the OSPFv3 graceful restart feature on a graceful-restart-capable router.
graceful-restart helper	Enables the OSPFv3 graceful restart feature on a graceful-restart-aware router.
show ipv6 ospf graceful-restart	Displays OSPFv3 graceful restart information.

debug ipv6 ospf lsdb

To display database modifications for Open Shortest Path First (OSPF) for IPv6, use the **d ebug ipv6 ospf lsdb**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ipv6 ospf lsdb

no debug ipv6 ospf lsdb

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(24)S	This command was introduced.
12.2(15)T	This command was integrated into Cisco IOS Release 12.2(15)T.
12.2(18)S	This command was integrated into Cisco IOS Release 12.2(18)S.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.

Usage Guidelines

Consult Cisco technical support before using this command.

Examples

The following example displays database modification information for OSPF for IPv6:

```
Router# debug ipv6 ospf lsdb
```

debug ipv6 ospf monitor

To display debugging information about the current wait-time used for shortest path first (SPF) scheduling, use the **debug ipv6 ospf monitor** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ipv6 ospf monitor

no debug ipv6 ospf monitor

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(33)SRC	This command was introduced.

Command History

12.2(33)SB	This command was integrated into Cisco IOS Release 12.2(33)SB.

Usage Guidelines

Consult Cisco technical support before using this command.

Examples

The following example shows debugging information about SPF scheduling:

```
Router# debug ipv6 ospf monitor
Sep 27 08:29:49.319: OSPFv3: Schedule SPF in area 0
        Change in LS ID 0.0.0.0, LSA type P
*Sep 27 08:29:49.327: OSPFv3: reset throttling to 5000ms next wait-interval 10000ms
*Sep 27 08:29:49.327: OSPFv3: schedule SPF: spf_time 00:09:36.032 wait_interval 5000ms
IOU_Topvar#
*Sep 27 08:29:54.331: OSPFv3: Begin SPF at 581.036ms, process time 40ms
*Sep 27 08:29:54.331:       spf_time 00:09:36.032, wait_interval 5000ms
*Sep 27 08:29:54.331: OSPFv3: Setting next wait-interval to 10000ms
*Sep 27 08:29:54.331: OSPFv3: End SPF at 581.036ms, Total elapsed time 0ms
*Sep 27 08:29:54.331:       Schedule time 00:09:41.036, Next wait_interval 10000ms
*Sep 27 08:29:54.331:       Intra: 0ms, Inter: 0ms, External: 0ms
*Sep 27 08:29:54.331:       R: 0, N: 0
*Sep 27 08:29:54.331:       SN: 0, SA: 0, X5: 0, X7: 0
*Sep 27 08:29:54.331:       SPF suspends: 0 intra, 0 total
```

debug ipv6 ospf packet

To display information about each Open Shortest Path First (OSPF) for IPv6 packet received, use the **debug ipv6 ospf packet** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ipv6 ospf packet

no debug ipv6 ospf packet

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(24)S	This command was introduced.
12.2(15)T	This command was integrated into Cisco IOS Release 12.2(15)T.
12.2(18)S	This command was integrated into Cisco IOS Release 12.2(18)S.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.

Usage Guidelines

Consult Cisco technical support before using this command.

Examples

The following example displays information about each OSPF for IPv6 packet received:

```
Router# debug ipv6 ospf packet
```

debug ipv6 ospf spf statistic

To display statistical information while running the shortest path first (SPF) algorithm, use the **debug ipv6 ospf spf statistic**command in privileged EXEC mode. To disable the debugging output, use the **no** form of this command.

debug ipv6 ospf spf statistic

no debug ipv6 ospf spf statistic

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(24)S	This command was introduced.
12.2(15)T	This command was integrated into Cisco IOS Release 12.2(15)T.
12.2(18)S	This command was integrated into Cisco IOS Release 12.2(18)S.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.

Usage Guidelines

The **debug ipv6 ospf spf statistic** command displays the SPF calculation times in milliseconds, the node count, and a time stamp. Consult Cisco technical support before using this command.

Examples

The following example displays statistical information while running the SPF algorithm:

```
Router# debug ipv6 ospf spf statistics
```

Related Commands

Command	Description
debug ipv6 ospf	Displays debugging information for the OSPFv3 for IPv6 feature.
debug ipv6 ospf events	Displays information on OSPFv3-related events.

Command	Description
debug ipv6 ospf packet	Displays information about each OSPFv3 packet received.

debug ipv6 packet

To display debug messages for IPv6 packets, use the **debug ipv6 packet**command in privileged EXEC mode. To disable debug messages for IPv6 packets, use the **no** form of this command.

debug ipv6 packet [**access-list** *access-list-name*] **[detail]**

no debug ipv6 packet [**access-list** *access-list-name*] **[detail]**

Syntax Description

access-list *access-list-name*	(Optional) Specifies an IPv6 access list. The access list name cannot contain a space or quotation mark, or begin with a numeric
detail	(Optional) May display additional detailed information about the IPv6 packet.

Command Default

Debugging for IPv6 packets is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(2)T	This command was introduced.
12.0(21)ST	This command was integrated into Cisco IOS Release 12.0(21)ST.
12.0(22)S	This command was integrated into Cisco IOS Release 12.0(22)S.
12.0(23)S	The **access-list** and **detail** keywords, and the *access-list-name*argument, were added.
12.2(13)T	The **access-list** and **detail** keywords, and the *access-list-name*argument, were added.
12.2(14)S	This command was integrated into Cisco IOS Release 12.2(14)S.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(25)SG	This command was integrated into Cisco IOS Release 12.2(25)SG.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.

Usage Guidelines

The **debug ipv6 packet**command is similar to the **debug ip packet**command, except that it is IPv6-specific.

Note

By default, the network server sends the output from debug commands and system error messages to the console. To redirect debug output, use the logging command options within global configuration mode. Destinations include the console, virtual terminals, internal buffer, and UNIX hosts running a syslog server. For complete information on debug commands and redirecting debug output, refer to the *Cisco IOS Debug Command Reference* .

IPv6 debugging information includes packets received, generated, and forwarded. Fast-switched packets do not generate messages. When an IPv6 access list is specified by using the **access-list**keyword and *access-list-name*argument, only packets matching the access list permit entries are displayed.

Caution

Because the **debug ipv6 packet** command generates a substantial amount of output, use it only when traffic on the IPv6 network is low, so other activity on the system is not adversely affected.

Examples

The following example shows output for the **debug ipv6 packet**command:

```
Router# debug ipv6 packet
13:25:40:IPV6:source 2000:0:0:3::1 (local)
13:25:40:      dest 2000:0:0:3::2 (FastEthernet0/0)
13:25:40:      traffic class 96, flow 0x0, len 143+195, prot 6, hops 64, originating
13:25:40:IPv6:Sending on FastEthernet0/0
13:25:40:IPV6:source 2000:0:0:3::2 (FastEthernet0/0)
13:25:40:      dest 2000:0:0:3::1
13:25:40:      traffic class 96, flow 0x0, len 60+14, prot 6, hops 64, forward to ulp
13:25:45:IPV6:source FE80::203:E4FF:FE12:CC1D (local)
13:25:45:      dest FF02::9 (Ethernet1/1)
13:25:45:      traffic class 112, flow 0x0, len 72+1428, prot 17, hops 255, originating
13:25:45:IPv6:Sending on Ethernet1/1
13:25:45:IPV6:source FE80::203:E4FF:FE12:CC00 (local)
13:25:45:      dest 2000:0:0:3::2 (FastEthernet0/0)
13:25:45:      traffic class 112, flow 0x0, len 72+8, prot 58, hops 255, originating
13:25:45:IPv6:Sending on FastEthernet0/0
13:25:45:IPV6:source 2000:0:0:3::2 (FastEthernet0/0)
13:25:45:      dest FE80::203:E4FF:FE12:CC00
13:25:45:      traffic class 112, flow 0x0, len 64+14, prot 58, hops 255, forward to ulp
13:25:45:IPV6:source FE80::203:A0FF:FED6:1400 (FastEthernet0/0)
13:25:45:      dest 2000:0:0:3::1
13:25:45:      traffic class 112, flow 0x0, len 72+14, prot 58, hops 255, forward to ulp
```

The table below describes the significant fields shown in the display.

Table 80: debug ipv6 packet Field Descriptions

Field	Description
IPV6:	Indicates that this is an IPv6 packet.
source 2000:0:0:3::1 (local)	The source address in the IPv6 header of the packet.
dest 2000:0:0:3::2 (FastEthernet0/0)	The destination address in the IPv6 header of the packet.

Field	Description
traffic class 96	The contents of the traffic class field in the IPv6 header.
flow 0x0	The contents of the flow field of the IPv6 header. The flow field is used to label sequences of packets for which special handling is necessary by IPv6 routers.
len 64+14	The length of the IPv6 packet. The length is expressed as two numbers with a plus (+) character between the numbers. The first number is the length of the IPv6 portion (IPv6 header length plus payload length). The second number is the entire datagram size minus the first number.
prot 6	The protocol field in the IPv6 header. Describes the next layer protocol that is carried by the IPv6 packet. In the example, the protocol 58 signifies that the next layer protocol is ICMPv6.
hops 64	The hops field in the IPv6 packet. This field is similar in function to the IPv4 time-to-live field.
originating	The presence of this field indicates that the packet shown was originated by the router.
Sending on FastEthernet0/0	Specifies the interface on which the packet was sent.
forward to ulp	Indicates that the packet was received by the router at the destination address and was forwarded to an upper-layer protocol (ulp) for processing.

debug ipv6 pim

To enable debugging on Protocol Independent Multicast (PIM) protocol activity, use the **debug ipv6 pim**command in privileged EXEC mode. To restore the default value, use the **no** form of this command.

debug ipv6 pim [*group-name*| *group-address*| **interface** *interface-type*| **bsr**| **group**| **neighbor**]

no debug ipv6 pim [*group-name*| *group-address*| **interface** *interface-type*| **bsr**| **group**| **neighbor**]

Syntax Description

group-name | *group-address*	(Optional) IPv6 address or name of the multicast group.
interface *interface-type*	(Optional) Displays debugging statistics about a specific interface type.
bsr	(Optional) Displays debugging statistics specific to bootstrap router (BSR) protocol operation.
group	(Optional) Displays debugging information about group-related activity.
neighbor	(Optional) Displays debugging statistics related to hello message processing and neighbor cache management.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(2)T	This command was introduced.
12.2(18)S	This command was integrated into Cisco IOS Release 12.2(18)S.
12.0(26)S	This command was integrated into Cisco IOS Release 12.0(26)S.
12.0(28)S	The **bsr** keyword was added.
12.2(25)S	The **bsr** keyword was added.
12.3(11)T	The **bsr** keyword was added.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(25)SG	This command was integrated into Cisco IOS Release 12.2(25)SG.

Release	Modification
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.
Cisco IOS XE Release 2.1	This command was introduced on Cisco ASR 1000 Series Routers.

Usage Guidelines

This command helps discover whether the PIM protocol activities are working correctly.

The messages displayed by the **debug ipv6 pim**command show all PIM protocol messages, such as joins and prunes, received from or sent to other routers. Use this command in conjunction with **debug ipv6 mld**to display additional multicast activity, to learn more information about the multicast routing process, or to learn why packets are forwarded out of particular interfaces.

Examples

The following example enables debugging on PIM activity:

```
Router# debug ipv6 pim
```

Related Commands

Command	Description
debug ipv6 mld	Enables debugging on MLD protocol activity.

debug ipv6 pim df-election

To display debug messages for Protocol Independent Multicast (PIM) bidirectional designated forwarder (DF) election message processing, use the **debug ipv6 pim df-election**command in privileged EXEC mode. To disable debug messages for PIM bidirectional DF election message processing, use the **no** form of this command.

debug ipv6 pim df-election [**interface** *type number*] [**rp** *rp-name*| *rp-address*]

no debug ipv6 pim df-election [**interface** *type number*] [**rp** *rp-name*| *rp-address*]

Syntax Description

interface	(Optional) Specifies that debug messages on a specified interface will be displayed.
type number	(Optional) Interface type and number. For more information, use the question mark (?) online help function.
rp	(Optional) Specifies that debug messages on a specified Route Processor (RP) will be displayed.
rp-name	(Optional) The name of the specified RP.
rp-address	(Optional) The IPv6 address of the specified RP.

Command Default

Debugging for PIM bidirectional DF election message processing is not enabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.3(7)T	This command was introduced.
12.2(25)S	This command was integrated into Cisco IOS Release 12.2(25)S.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.

Usage Guidelines

Use the **debug ipv6 pim df-election**command if traffic is not flowing properly when operating in PIM bidirectional mode or if the **show ipv6 pim df** and **show ipv6 pim df winner**commandsdo not display the expected information.

Examples

The following example shows how to enable debugging for PIM bidirectional DF election message processing on Ethernet interface 1/0 and at 200::1:

```
Route# debug ipv6 pim df-election interface ethernet 1/0 rp 200::1
```

Related Commands

Command	Description
ipv6 pim rp-address	Configures the address of a PIM RP for a particular group range.
show ipv6 pim df	Displays the DF-election state of each interface for each RP.
show ipv6 pim df winner	Displays the DF-election winner on each interface for each RP.

debug ipv6 pim limit

To enable debugging for Protocol Independent Multicast (PIM) interface limits, use the **debug ipv6 pim limit** command in privileged EXEC mode. To restore the default value, use the **no** form of this command.

debug ipv6 pim limit [*group*]

no debug ipv6 pim limit

Syntax Description

group	(Optional) Specific group to be debugged.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(33)SRE	This command was introduced.

Usage Guidelines

Use the **debug ipv6 pim limit** command to display debugging information for interface limits and costs. Use the optional *group* argument to specify a particular group to debug.

Examples

The following example enables PIM interface limit debugging:

```
Router# debug ipv6 pim limit
```

Related Commands

Command	Description
ipv6 multicast limit	Configures per-interface mroute state limiters in IPv6.
ipv6 multicast limit cost	Applies a cost to mroutes that match per interface mroute state limiters in IPv6.

debug ipv6 policy

To enable debugging of IPv6 policy routing packet activity, use the **debug ipv6 policy** command in user EXEC or privileged EXEC mode. To disable debugging, use the **no** form of this command.

debug ipv6 policy [*access-list-name*]

no debug ipv6 policy [*access-list-name*]

Syntax Description

access-list-name	(Optional) Name of the IPv6 access list. Names cannot contain a space or quotation mark or begin with a numeric.

Command Default

If no access list is specified using the optional *access-list-name* argument, information about all policy-matched and policy-routed packets is displayed.

Command Modes

User EXEC (>)

Privileged EXEC (#)

Command History

Release	Modification
12.3(7)T	This command was introduced.
12.2(30)S	This command was integrated into Cisco IOS Release 12.2(30)S.
12.2(33)SXI4	This command was integrated into Cisco IOS Release 12.2(33)SXI4.
Cisco IOS XE Release 3.2S	This command was integrated into Cisco IOS XE Release 3.2S.
15.1(1)SY	This command was integrated into Cisco IOS Release 15.1(1)SY.

Usage Guidelines

After you configure IPv6 policy routing, use the **debug ipv6 policy** command to verify that IPv6 policy-based routing (PBR) is policy-routing packets normally. Policy routing analyzes various parts of the packet and then routes the packet based on certain user-defined attributes in the packet. The **debug ipv6 policy** command helps you determine what policy is followed during routing. It displays information about whether a packet matches the given criteria, and if yes, the resulting routing information for the packet.

Do not use the **debug ipv6 policy** command unless you suspect a problem with IPv6 PBR policy routing.

Examples

The following example shows how to enable debugging of IPv6 policy routing packet activity. The output of this command is self-explanatory:

```
Device# debug ipv6 policy

00:02:38:IPv6 PBR:Ethernet0/0, matched src 2003::90 dst 2001:DB8::1 protocol 58
00:02:38:IPv6 PBR:set nexthop 2001:DB8::F, interface Ethernet1/0
00:02:38:IPv6 PBR:policy route via Ethernet1/0/2001:DB8::F
```

debug ipv6 pool

To enable debugging on IPv6 prefix pools, use the debug ipv6 pool command in privileged EXEC mode. To disable debugging, use the **no** form of this command.

debug ipv6 pool

no debug ipv6 pool

Syntax Description

This command has no keywords or arguments.

Command Default

No debugging is active.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(13)T	This command was introduced.

Examples

The following example enables debugging for IPv6 prefix pools:

```
Router# debug ipv6 pool
2w4d: IPv6 Pool: Deleting route/prefix 2001:0DB8::/29 to Virtual-Access1 for cisco
2w4d: IPv6 Pool: Returning cached entry 2001:0DB8::/29 for cisco on Virtual-Access1 to
pool1
2w4d: IPv6 Pool: Installed route/prefix 2001:0DB8::/29 to Virtual-Access1 for cisco
```

Related Commands

Command	Description
ipv6 local pool	Configures a local IPv6 prefix pool.
show ipv6 interface	Displays the usability status of interfaces configured for IPv6.
show ipv6 local pool	Displays information about defined IPv6 prefix pools.

debug ipv6 rip

To display debug messages for IPv6 Routing Information Protocol (RIP) transactions, use the **debug ipv6 rip** command in privileged EXEC mode. To disable debug messages for IPv6 RIP routing transactions, use the **no** form of this command.

Cisco IOS XE Release 3.9S, Cisco IOS Release 15.3(2)S, and Later Releases

debug ipv6 rip [*interface-type interface-number*] [**vrf** *vrf-name*]

no debug ipv6 rip [*interface-type interface-number*] [**vrf** *vrf-name*]

Releases Prior to Cisco IOS XE Release 3.9S and Cisco IOS Release 15.3(2)S

debug ipv6 rip [*interface-type interface-number*]

no debug ipv6 rip [*interface-type interface-number*]

Syntax Description

interface-type	(Optional) Interface type for which to display the debug messages.
interface-number	(Optional) Interface number for which to display the debug messages.
vrf *vrf-name*	(Optional) Displays information about the specified virtual routing and forwarding (VRF) instance.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(2)T	This command was introduced.
12.0(21)ST	This command was integrated into Cisco IOS Release 12.0(21)ST.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(25)SG	This command was integrated into Cisco IOS Release 12.2(25)SG.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.
Cisco IOS XE Release 2.1	This command was implemented on Cisco 1000 Series Aggregation Services Routers.

Release	Modification
Cisco IOS XE Release 3.9S	This command was modified. The **vrf** *vrf-name* keyword-argument pair was added.
15.3(2)S	This command was integrated into Cisco IOS Release 15.3(2)S.
15.3(3)M	This command was integrated into Cisco IOS Release 15.3(3)M.

Usage Guidelines

The **debug ipv6 rip** command is similar to the **debug ip rip** command, except that it is IPv6-specific.

Note By default, the network server sends the output from debug commands and system error messages to the console. To redirect debug output, use the **logging** command in global configuration mode. Destinations include the console, virtual terminals, internal buffer, and UNIX hosts running a syslog server. For complete information on debug commands and redirecting debug output, refer to the *Cisco IOS Debug Command Reference*.

Use the **debug ipv6 rip** command to enable IPv6 RIP debugging for RIP packets that are sent and received on all device interfaces. Use the **debug ipv6 rip** *interface-type interface-number* command to enable IPv6 RIP debugging for RIP packets that are sent and received only on the specified interface.

Use the **debug ipv6 rip vrf** *vrf-name* command to troubleshoot issues in the IPv6 RIP functionality when the VRF has already been enabled using a **vrf definition** *vrf-name* command. Ensure that the specified VRF name has already been defined. If a VRF name has not been defined, the following message is displayed:

```
% VRF <undefined VRF name> does not exist or does not have a RD.
```

Examples

The following is sample output from the **debug ipv6 rip** command:

```
Device# debug ipv6 rip

13:09:10:RIPng:Sending multicast update on Ethernet1/1 for as1_rip
13:09:10:       src=2001:DB8::1
13:09:10:       dst=2001:DB8:0:ABCD::1 (Ethernet1/1)
13:09:10:       sport=521, dport=521, length=32
13:09:10:       command=2, version=1, mbz=0, #rte=1
13:09:10:       tag=0, metric=1, prefix=::/0
13:09:28:RIPng:response received from 2001:DB8:0:0:E000::F on Ethernet1/1 for as1_rip
13:09:28:       src=FE80::202:FDFF:FE77:1E42 (Ethernet1/1)
13:09:28:       dst=FF02::9
13:09:28:       sport=521, dport=521, length=32
13:09:28:       command=2, version=1, mbz=0, #rte=1
13:09:28:       tag=0, metric=1, prefix=2000:0:0:1:1::/80
```

The above example shows two RIP packets; both are known as “responses” in RIP terminology and indicated by a “command” value of 2. The first is an update sent by the device, and the second is an update received by the device. Multicast update packets are sent to all neighboring IPv6 RIP devices (all devices that are on the same links as the device sending the update and have IPv6 RIP enabled). An IPv6 RIP device advertises the contents of its routing table to its neighbors by periodically sending update packets over those interfaces on which IPv6 RIP is configured. An IPv6 device may also send “triggered” updates immediately following a routing table change. In this case, the updates include only the changes to the routing table. An IPv6 RIP device may solicit the contents of the routing table of a neighboring device by sending a Request (command =1) message to the device. The device responds by sending an update (Response, command=2) containing its routing table. In the example, the received response packet could be a periodic update from the address 2001:DB8:0:0:E000::F or a response to a RIP request message that was previously sent by the local device.

The following is sample output from the **debug ipv6 rip vrf** command:

```
Device# debug ipv6 rip vrf blue

RIP Routing Protocol debugging is on for vrf blue

Sending:
*Mar 15 11:23:08.508: RIPng: Sending multicast update on Ethernet0/0 for vrf for vrf blue
*Mar 15 11:23:08.508:        src=2001:DB8:0:1:FFFF:1234::5
*Mar 15 11:23:08.508:        dst=2001:DB8:0:1::1 (Ethernet0/0)
*Mar 15 11:23:08.508:        sport=521, dport=521, length=52
*Mar 15 11:23:08.508:        command=2, version=1, mbz=0, #rte=2
*Mar 15 11:23:08.508:        tag=0, metric=1, prefix=6000::/64
*Mar 15 11:23:08.508:        tag=0, metric=1, prefix=2000::/64
*Mar 15 11:23:08.508: RIPng: Packet waiting
*Mar 15 11:23:08.508: RIPng: Process vrf received own response on Loopback1

Receiving
*Mar 15 11:23:20.316: RIPng: Packet waiting
*Mar 15 11:23:20.316: RIPng: response received from FE80::A8BB:CCFF:FE00:7C00 on Ethernet0/0
 for vrf
*Mar 15 11:23:20.316:        src=2001:DB8:0:1:FFFF:1234::4 (Ethernet0/0)
*Mar 15 11:23:20.316:        dst=2001:DB8::1
*Mar 15 11:23:20.316:        sport=521, dport=521, length=32
*Mar 15 11:23:20.316:        command=2, version=1, mbz=0, #rte=1
*Mar 15 11:23:20.316:        tag=0, metric=1, prefix=AAAA::/64
```

The table below describes the significant fields shown in the display.

Table 81: debug ipv6 rip vrf Field Descriptions

Field	Description
src	The address from which the update was originated.
dst	The destination address for the update.
sport, dport, length	The source, destination ports and the length for the update. (IPv6 RIP uses port 521, as shown in the display.)
command	The command field within the RIP packet. A value of 2 indicates that the RIP packet is a response (update); a value of 1 indicates that the RIP packet is a request.
version	The version of IPv6 RIP being used. The current version is 1.
mbz	There must be a 0 (mbz) field within the RIP packet.
#rte	Indicates the number of routing table entries (RTEs) that the RIP packet contains.

Field	Description
tag metric prefix	The tag, metric, and prefix fields are specific to each RTE contained in the update. The tag field is intended to allow for the flagging of IPv6 RIP “internal” and “external” routes. The metric field is the distance metric from the device (sending this update) to the prefix. The prefix field is the IPv6 prefix of the destination being advertised.

Related Commands

Command	Description
clear ipv6 rip	Deletes routes from the IPv6 RIP routing table.
ipv6 rip vrf-mode enable	Enables VRF support for IPv6 RIP.
show ipv6 rip	Displays information about current IPv6 RIP processes.
vrf definition	Configures a VRF routing table instance.

debug ipv6 routing

To display debug messages for IPv6 routing table updates and route cache updates, use the **debug ipv6 routing**command in privileged EXEC mode. To disable debug messages for IPv6 routing table updates and route cache updates, use the **no** form of this command.

debug ipv6 routing

no debug ipv6 routing

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging for IPv6 routing table updates and route cache updates is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(2)T	This command was introduced.
12.0(21)ST	This command was integrated into Cisco IOS Release 12.0(21)ST.
12.0(22)S	This command was integrated into Cisco IOS Release 12.0(22)S.
12.2(14)S	This command was integrated into Cisco IOS Release 12.2(14)S.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(25)SG	This command was integrated into Cisco IOS Release 12.2(25)SG.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.

Usage Guidelines

The **debug ipv6 routing**command is similar to the **debug ip routing**command, except that it is IPv6-specific.

Note By default, the network server sends the output from debug commands and system error messages to the console. To redirect debug output, use the logging command options within global configuration mode. Destinations include the console, virtual terminals, internal buffer, and UNIX hosts running a syslog server. For complete information on debug commands and redirecting debug output, refer to the *Cisco IOS Debug Command Reference* .

Examples

The following example shows output for the **debug ipv6 routing**command:

```
Router# debug ipv6 routing
13:18:43:IPv6RT0:Add 2000:0:0:1:1::/80 to table
13:18:43:IPv6RT0:Better next-hop for 2000:0:0:1:1::/80, [120/2]
13:19:09:IPv6RT0:Add 2000:0:0:2::/64 to table
13:19:09:IPv6RT0:Better next-hop for 2000:0:0:2::/64, [20/1]
13:19:09:IPv6RT0:Add 2000:0:0:2:1::/80 to table
13:19:09:IPv6RT0:Better next-hop for 2000:0:0:2:1::/80, [20/1]
13:19:09:IPv6RT0:Add 2000:0:0:4::/64 to table
13:19:09:IPv6RT0:Better next-hop for 2000:0:0:4::/64, [20/1]
13:19:37:IPv6RT0:Add 2000:0:0:6::/64 to table
13:19:37:IPv6RT0:Better next-hop for 2000:0:0:6::/64, [20/2]
```

The **debug ipv6 routing** command displays messages whenever the routing table changes. For example, the following message indicates that a route to the prefix 2000:0:0:1:1::/80 was added to the routing table at the time specified in the message.

```
13:18:43:IPv6RT0:Add 2000:0:0:1:1::/80 to table
```

The following message indicates that the prefix 2000:0:0:2::/64 was already in the routing table; however, a received advertisement provided a lower cost path to the prefix. Therefore, the routing table was updated with the lower cost path. (The [20/1] in the example is the administrative distance [20] and metric [1] of the better path.)

```
13:19:09:IPv6RT0:Better next-hop for 2000:0:0:2::/64, [20/1]
```

Related Commands

Command	Description
debug ipv6 rip	Displays debug messages for IPv6 RIP routing transactions.

debug ipv6 snooping

To enable debugging for security snooping information in IPv6, use the **debug ipv6 snooping**command in privileged EXEC mode.

debug ipv6 snooping [**binding-table**| **classifier**| **errors**| **feature-manager**| **filter** *acl*| **ha**| **hw-api**| **interface** *interface*| **memory**| **ndp-inspection**| **policy**| **vlan** *vlanid*| **switcher**| **filter** *acl*| **interface** *interface*| *vlan-id*]

no debug ipv6 snooping

Syntax Description

binding-table	(Optional) Displays information about the neighbor binding table.
classifier	(Optional) Displays information about the classifier.
errors	(Optional) Displays information about snooping security errors.
feature-manager	(Optional) Displays feature manager information.
filter *acl*	(Optional) Allows users to configure an access list to filter debugged traffic.
ha	(Optional) Displays information about high availability (HA) and stateful switchover (SSO).
hw-api	(Optional) Displays information about the hardware API.
interface *interface*	(Optional) Provides debugging information on a specified interface.
memory	(Optional) Displays information about security snooping memory.
ndp-inspection	(Optional) Displays information about Neighbor Discovery inspection.
policy	(Optional)
switcher	(Optional) Displays packets handled by the switcher.
vlan-id	(Optional) Provides debugging information about a specified VLAN ID.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(50)SY	This command was introduced.

Usage Guidelines

The **debug ipv6 snooping** command provides debugging output for IPv6 snooping information.

Because debugging output is assigned high priority in the CPU process, you should use debug commands only to troubleshoot specific problems or during troubleshooting sessions with Cisco technical support staff.

Examples

The following example enables debugging for all IPv6 snooping information:

```
Router# debug ipv6 snooping
```

debug ipv6 snooping raguard

To enable debugging for security snooping information in the IPv6 router advertisement (RA) guard feature, use the **debug ipv6 snooping raguard** command in privileged EXEC mode.

debug ipv6 snooping raguard [*filter*| *interface*| *vlanid*]

no debug ipv6 snooping raguard

Syntax Description

filter	(Optional) Allows users to configure an access list to filter debugged traffic.
interface	(Optional) Provides debugging information about a specified interface configured with the IPv6 RA guard feature.
vlanid	(Optional) Provides debugging information about a specified VLAN ID configured with the IPv6 RA guard feature.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(54)SG	This command was introduced.
12.2(50)SY	This command was integrated into Cisco IOS Release 12.2(50)SY.
15.2(4)S	This command was integrated into Cisco IOS Release 15.2(4)S.

Usage Guidelines

The **debug ipv6 snooping raguard** command provides debugging output for IPv6 RA guard events and errors that may occur.

Because debugging output is assigned high priority in the CPU process, you should use debug commands only to troubleshoot specific problems or during troubleshooting sessions with Cisco technical support staff. Also, you should use debug commands during periods of lower network traffic and fewer users. Debugging during these periods decreases the likelihood that increased debug command processing overhead will affect system use.

Examples

The following example shows the command enabling debugging for the IPv6 RA guard feature:

```
Router# debug ipv6 snooping raguard
```

Related Commands

Command	Description
ipv6 nd raguard	Applies the IPv6 RA guard feature.

debug ipv6 spd

To enable debugging output for the most recent Selective Packet Discard (SPD) state transition, use the **debug ipv6 spd**command in privileged EXEC mode.

debug ipv6 spd

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
15.1(3)T	This command was introduced.

Usage Guidelines

The **debug ipv6 spd** command enables debugging information to be reviewed for the most recent SPD state transition and any trend historical data.

Examples

The following example shows how to enable debugging for the most recent SPD state transition:

```
Router# debug ipv6 spd
```

debug ipv6 static

To enable Bidirectional Forwarding Detection for IPv6 (BFDv6) debugging, use the **debug ipv6 static**command in privileged EXEC mode.

debug ipv6 static

Command Default

Debugging is not enabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS XE Release 2.1.0	This command was introduced.
15.1(2)T	This command was modified. It was integrated into Cisco IOS Release 15.1(2)T.
15.1(1)SG	This command was integrated into Cisco IOS Release 15.1(1)SG.
15.1(1)SY	This command was modified. Support for IPv6 was added to Cisco IOS Release 15.1(1)SY.

Usage Guidelines

Use the **debug ipv6 static**command to monitor BFDv6 operation.

Examples

The following example enables BFDv6 debugging:

```
Router# debug ipv6 static
```

Related Commands

Command	Description
monitor event ipv6 static	Monitors the operation of the IPv6 static and IPv6 static BFDv6 neighbors using event trace.
show ipv6 static	Displays the current contents of the IPv6 routing table.

debug ipv6 wccp

To display information about IPv6 Web Cache Communication Protocol (WCCP) services, use the **debug ipv6 wccp** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ipv6 wccp {**default**| **vrf** *vrf-name* {**events**| **packets [control]**}| **events**| **packets** [**bypass**| **control**| **redirect**]| **platform**| **subblocks**}

no debug ipv6 wccp {**default**| **vrf** *vrf-name* {**events**| **packets [control]**}| **events**| **packets** [**bypass**| **control**| **redirect**]| **platform**| **subblocks**}

Syntax Description

default	Displays information about default WCCP services.
vrf *vrf-name*	Specifies a virtual routing and forwarding (VRF) instance to associate with a service group.
events	Displays information about significant WCCP events.
packets	Displays information about every WCCP packet received or sent by the router.
control	(Optional) Displays information about WCCP control packets.
bypass	(Optional) Displays information about WCCP bypass packets.
redirect	(Optional) Displays information about WCCP redirect packets.
platform	Displays information about the WCCP platform application programming interface (API).
subblocks	Displays information about WCCP subblocks.

Command Default

Debug information is not displayed.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
15.2(3)T	This command was introduced.
15.1(1)SY1	This command was integrated into Cisco IOS Release 15.1(1)SY1.

Usage Guidelines

When the **vrf** keyword is not used, the command displays debug information about all WCCP services on the router. The **default** keyword is used to specify default WCCP services.

Examples

The following is sample output from the **debug ipv6 wccp events** command when a Cisco Cache Engine is added to the list of available Web caches:

```
Router# debug ipv6 wccp events
WCCP-EVNT: Built I_See_You msg body w/1 usable web caches, change # 0000000A
WCCP-EVNT: Web Cache 2001:DB8:1::1 added
WCCP-EVNT: Built I_See_You msg body w/2 usable web caches, change # 0000000B
WCCP-EVNT: Built I_See_You msg body w/2 usable web caches, change # 0000000C
```

The following is sample output from the **debug ipv6 wccp packets** command. The router is sending keepalive packets to the Cisco Cache Engines at 2001:DB8:1::2 and 2001:DB8:1::1. Each keepalive packet has an identification number associated with it. When the Cisco Cache Engine receives a keepalive packet from the router, it sends a reply with the identification number back to the router.

```
Router# debug ipv6 wccp packets
WCCP-PKT: Received valid Here_I_Am packet from 2001:DB8:1::2 w/rcvd_id 00003532
WCCP-PKT: Sending I_See_You packet to 2001:DB8:1::2 w/ rcvd_id 00003534
WCCP-PKT: Received valid Here_I_Am packet from 2001:DB8:1::1 w/rcvd_id 00003533
WCCP-PKT: Sending I_See_You packet to 2001:DB8:1::1 w/ rcvd_id 00003535
WCCP-PKT: Received valid Here_I_Am packet from 2001:DB8:1::2 w/rcvd_id 00003534
WCCP-PKT: Sending I_See_You packet to 2001:DB8:1::2 w/ rcvd_id 00003536
WCCP-PKT: Received valid Here_I_Am packet from 2001:DB8:1::1 w/rcvd_id 00003535
WCCP-PKT: Sending I_See_You packet to 2001:DB8:1::1 w/ rcvd_id 00003537
WCCP-PKT: Received valid Here_I_Am packet from 2001:DB8:1::2 w/rcvd_id 00003536
WCCP-PKT: Sending I_See_You packet to 2001:DB8:1::2 w/ rcvd_id 00003538
WCCP-PKT: Received valid Here_I_Am packet from 2001:DB8:1::1 w/rcvd_id 00003537
WCCP-PKT: Sending I_See_You packet to 2001:DB8:1::1 w/ rcvd_id 00003539
```

Related Commands

Command	Description
clear ipv6 wccp	Clears the counter for packets redirected using WCCP.
ipv6 wccp	Enables support of the specified WCCP service for participation in a service group.
ipv6 wccp redirect	Enables packet redirection on an outbound or inbound interface using WCCP.
show ipv6 interface	Lists a summary of the IP information and status of an interface.

debug ipx ipxwan

To display debugging information for interfaces configured to use IPX wide-area network (IPXWAN), use the **debug ipx ipxwan** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ipx ipxwan

no debug ipx ipxwan

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

The **debug ipx ipxwan** command is useful for verifying the startup negotiations between two routers running the IPX protocol through a WAN. This command produces output only during state changes or startup. During normal operations, no output is produced.

Examples

The following is sample output from the **debug ipx ipxwan** command during link startup:

```
Router# debug ipx ipxwan
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial1, changed state to up
IPXWAN: state (Disconnect -> Sending Timer Requests) [Serial1/6666:200 (IPX line
 state brought up)]
IPXWAN: state (Sending Timer Requests -> Disconnect) [Serial1/6666:200 (IPX line
 state brought down)]
IPXWAN: state (Disconnect -> Sending Timer Requests) [Serial1/6666:200 (IPX line
 state brought up)]
IPXWAN: Send TIMER_REQ [seq 0] out Serial1/6666:200
IPXWAN: Send TIMER_REQ [seq 1] out Serial1/6666:200
IPXWAN: Send TIMER_REQ [seq 2] out Serial1/6666:200
IPXWAN: Send TIMER_REQ [seq 0] out Serial1/6666:200
IPXWAN: Rcv TIMER_REQ on Serial1/6666:200, NodeID 1234, Seq 1
IPXWAN: Send TIMER_REQ [seq 1] out Serial1/6666:200
IPXWAN: Rcv TIMER_RSP on Serial1/6666:200, NodeID 1234, Seq 1, Del 6
IPXWAN: state (Sending Timer Requests -> Master: Sent RIP/SAP) [Serial1/6666:200
 (Received Timer Response as master)]
IPXWAN: Send RIPSAP_INFO_REQ [seq 0] out Serial1/6666:200
IPXWAN: Rcv RIPSAP_INFO_RSP from Serial1/6666:200, NodeID 1234, Seq 0
IPXWAN: state (Master: Sent RIP/SAP -> Master: Connect) [Serial1/6666:200 (Received Router
 Info Rsp as Master)]
```

The following line indicates that the interface has initialized:

```
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial1, changed state to up
```

The following lines indicate that the startup process failed to receive a timer response, brought the link down, then brought the link up and tried again with a new timer set:

```
IPXWAN: state (Sending Timer Requests -> Disconnect) [Serial1/6666:200 (IPX line
 state brought down)]
IPXWAN: state (Disconnect -> Sending Timer Requests) [Serial1/6666:200 (IPX line
 state brought up)]
```

The following lines indicate that the interface is sending timer requests and waiting for a timer response:

```
IPXWAN: Send TIMER_REQ [seq 0] out Serial1/6666:200
IPXWAN: Send TIMER_REQ [seq 1] out Serial1/6666:200
```

The following lines indicate that the interface has received a timer request from the other end of the link and has sent a timer response. The fourth line shows that the interface has come up as the master on the link.

```
IPXWAN: Rcv TIMER_REQ on Serial1/6666:200, NodeID 1234, Seq 1
IPXWAN: Send TIMER_REQ [seq 1] out Serial1/6666:200
IPXWAN: Rcv TIMER_RSP on Serial1/6666:200, NodeID 1234, Seq 1, Del 6
IPXWAN: state (Sending Timer Requests -> Master: Sent RIP/SAP) [Serial1/6666:200
 (Received Timer Response as master)]
```

The following lines indicate that the interface is sending RIP/SAP requests:

```
IPXWAN: Send RIPSAP_INFO_REQ [seq 0] out Serial1/6666:200
IPXWAN: Rcv RIPSAP_INFO_RSP from Serial1/6666:200, NodeID 1234, Seq 0
IPXWAN: state (Master: Sent RIP/SAP -> Master: Connect) [Serial1/6666:200 (Received Router
 Info Rsp as Master)]
```

debug ipx nasi

To display information about NetWare Asynchronous Services Interface (NASI) connections, use the **debug ipx nasi** command in Privileged EXEC configuration mode. To disable debugging output, use the **no** form of this command.

debug ipx nasi {**packets**| **error**| **activity**}

no debug ipx nasi {**packets**| **error**| **activity**}

Syntax Description

packets	Displays normal operating messages relating to incoming and outgoing NASI packets. This is the default.
error	Displays messages indicating an error or failure in the protocol processing.
activity	Displays messages relating to internal NASI processing of NASI connections. The activity option includes all NASI activity such as traffic indication, timer events, and state changes.

Command Default

Nasi protocol debugging is disabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
11.1	This command was introduced.

Usage Guidelines

Use the **debug ipx nasi**command to display handshake or negotiation details between Sequenced Packet Exchange (SPX), NASI protocol, and other protocols or applications. Use the **packets** option to determine the NASI traffic flow, and use the **error** option as a quick check to see why NASI connections failed.

Examples

The following is sample output from the **debug ipx nasi**command with the **packet** and **error**options.

```
Router# debug ipx nasi packet
Router# debug ipx nasi error
NASI0: 6E6E Check server info
NASI0: 6E6E sending server-info 4F00   Good response: 43 bytes
NASI0: 7A6E Query Port. Find first
NASI0: FFirst: line 0 DE, port: TTY1-__________ASYNC___^, group: ASYNC___^
```

```
NASI0: 7A6E sending Qport find-first response: 300 bytes
NASI0: 7B6E port request. setting up port
NASI: Check-login User: c h r i s
NASI: Check-login PW hash: C7 A6 C5 C7 C4 C0 C5 C3 C4 CC C5 CF C4 C8 C5 CB C4 D4 C5 D7 C4
D0 C5 D3 C4
NASI: Check-login PW: l a b
NASI1: 7B6E sending NCS Good server Data Ack in 0 bytes pkt in 13 size pkt
NASI1: 7B6E sending Preq response: 303 bytes Good
NASI1: 7B6E port request. setting up port
NASI1: 7B6E sending NCS Good server Data Ack in 0 bytes pkt in 13 size pkt
NASI1: 7B6E sending Preq response: 303 bytes Good
NASI1: 7B6E Unknown NASI code 4500 Pkt Size: 13
 45 0 0 FC 0 2 0 20 0 0 FF 1 0
NASI1: 7B6E Flush Rx Buffers
NASI1: 7B6E sending NASI server TTY data: 1 byte in 14 size pkt
NASI1: 7B6E sending NCS Good server Data Ack in 1 bytes pkt in 13 size pkt
```

In the following line:

- 0 in NASI0 is the number of the terminal (TTY) to which this NASI connection is attached.
- 0 in NASI0 is used by all NASI control connections.
- 6E6E is the associated SPX connection pointer for this NASI connection.
- Check server info is a type of incoming NASI packet.

```
NASI0: 6E6E Check server info
```

The following message indicates that the router is sending back a server-info packet with a positive acknowledgment, and the packet size is 43 bytes:

```
NASI0: 6E6E sending server-info 4F00 Good response: 43 bytes
```

The following line is a NASI packet type. Find first and Find next are NASI packet types.

```
NASI0: 7A6E Query Port. Find first
```

The following line indicates that the outgoing find first packet for the NASI connection 7A6E has line 0 DE, port name TTY1, and general name ASYNC:

```
NASI0: FFirst: line 0 DE, port: TTY1-__________ASYNC___^, group: ASYNC___^
```

The following two lines indicate:

- Received NASI packet for NASI connection in line 1. 7B6E is the NASI connection pointer. The packet code is 4500 and is not recognizable by Cisco.
- Hexadecimal dump of the packet in line 2.

```
NASI1: 7B6E Unknown NASI code 4500 Pkt Size: 13
 45 0 0 FC 0 2 0 20 0 0 FF 1 0
```

Related Commands

Command	Description
debug ipx spx	Displays debugging messages related to the SPX protocol.

debug ipx packet

To display information about packets received, sent, and forwarded, use the **debug ipx packet** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ipx packet

no debug ipx packet

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

This command is useful for learning whether Internetwork Packet Exchange (IPX) packets are traveling over a router.

Note In order to generate **debug ipx packet** information on all IPX traffic traveling over the router, you must first configure the router so that fast switching is disabled. Use the **no ipx route-cache** command on all interfaces on which you want to observe traffic. If the router is configured for IPX fast switching, only non fast-switched packets will produce output. When the IPX cache is invalidated or cleared, one packet for each destination is displayed as the cache is repopulated.

Examples

The following is sample output from the **debug ipx packet** command:

```
Router# debug ipx packet
IPX: src=160.0260.8c4c.4f22, dst=1.0000.0000.0001, packet received
IPX: src=160.0260.8c4c.4f22, dst=1.0000.0000.0001,gw=183.0000.0c01.5d85,
sending packet
```

The first line indicates that the router receives a packet from a Novell station (address 160.0260.8c4c.4f22); this trace does not indicate the address of the immediate router sending the packet to this router. In the second line, the router forwards the packet toward the Novell server (address 1.0000.0000.0001) through an immediate router (183.0000.0c01.5d85).

The table below describes the significant fields shown in the display.

Table 82: debug ipx packet Field Descriptions

Field	Description
IPX	Indicates that this is an IPX packet.
src=160.0260.8c4c.4f22	Source address of the IPX packet. The Novell network number is 160. Its MAC address is 0260.8c4c.4f22.

Field	Description
dst=1.0000.0000.0001	Destination address for the IPX packet. The address 0000.0000.0001 is an internal MAC address, and the network number 1 is the internal network number of a Novell 3.11 server.
packet received	Router received this packet from a Novell station, possibly through an intermediate router.
gw=183.0000.0c01.5d85	Router is sending the packet over to the next hop router; its address of 183.0000.0c01.5d85 was learned from the IPX routing table.
sending packet	Router is attempting to send this packet.

debug ipx routing

To display information on Internetwork Packet Exchange (IPX) routing packets that the router sends and receives, use the **debug ipx routing** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ipx routing {**activity**| **events**}

no debug ipx routing {**activity**| **events**}

Syntax Description

activity	Displays messages relating to IPX routing activity.
events	Displays messages relating to IPX routing events.

Command Modes

Privileged EXEC

Usage Guidelines

Normally, a router or server sends out one routing update per minute. Each routing update packet can include up to 50 entries. If many networks exist on the internetwork, the router sends out multiple packets per update. For example, if a router has 120 entries in the routing table, it would send three routing update packets per update. The first routing update packet would include the first 50 entries, the second packet would include the next 50 entries, and the last routing update packet would include the last 20 entries.

Examples

The following is sample output from the **debug ipx routing**command:

```
Router# debug ipx routing
IPXRIP: update from 9999.0260.8c6a.1733
        110801 in 1 hops, delay 2
IPXRIP: sending update to 12FF02:ffff.ffff.ffff via Ethernet 1
        network 555, metric 2, delay 3
        network 1234, metric 3, delay 4
```

The table below describes the significant fields shown in the display.

Table 83: debug ipx routing Field Descriptions

Field	Description
IPXRIP	IPX RIP packet.
update from 9999.0260.8c6a.1733	Routing update packet from an IPX server at address 9999.0260.8c6a.1733.
110801 in 1 hops	Network 110801 is one hop away from the router at address 9999.0260.8c6a.1733.

Field	Description
delay 2	Delay is a time measurement (1/18th second) that the NetWare shell uses to estimate how long to wait for a response from a file server. Also known as ticks.
sending update to 12FF02:ffff.ffff.ffff via Ethernet 1	Router is sending this IPX routing update packet to address 12FF02:ffff.ffff.ffff through Ethernet interface 1.
network 555	Packet includes routing update information for network 555.
metric 2	Network 555 is two metrics (or hops) away from the router.
delay 3	Network 555 is a delay of 3 away from the router. Delay is a measurement that the NetWare shell uses to estimate how long to wait for a response from a file server. Also known as ticks.

Related Commands

Command	Description
debug ipx sap	Displays information about IPX SAP packets.

debug ipx sap

To display information about Internetwork Packet Exchange (IPX) S ervice Advertisement Protocol (SAP) packets, use the **debug ipx sap** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ipx sap [**activity**| **events**]

no debug ipx sap [**activity**| **events**]

Syntax Description

activity	(Optional) Provides more detailed output of SAP packets, including displays of services in SAP packets.
events	(Optional) Limits amount of detailed output for SAP packets to those that contain interesting events.

Command Modes

Privileged EXEC

Usage Guidelines

Normally, a router or server sends out one SAP update per minute. Each SAP packet can include up to seven entries. If many servers are advertising on the network, the router sends out multiple packets per update. For example, if a router has 20 entries in the SAP table, it would send three SAP packets per update. The first SAP would include the first seven entries, the second SAP would include the next seven entries, and the last update would include the last six entries.

Obtain the most meaningful detail by using the **debug ipx sap activity** and the **debug ipx sap events** commands together.

Caution

Because the **debug ipx sap**command can generate a substantial amount of output, use it with caution on networks that have many interfaces and large service tables.

Examples

The following is sample output from the **debug ipx sap** command:

```
Router# debug ipx sap
IPXSAP: at 0023F778:
I SAP Response type 0x2 len 160 src:160.0000.0c00.070d dest:160.ffff.ffff.ffff(452)
 type 0x4, "Hello2", 199.0002.0004.0006 (451), 2 hops
 type 0x4, "Hello1", 199.0002.0004.0008 (451), 2 hops
IPXSAP: sending update to 160
IPXSAP: at 00169080:
 O SAP Update type 0x2 len 96 ssoc:0x452 dest:160.ffff.ffff.ffff(452)
IPX: type 0x4, "Magnolia", 42.0000.0000.0001 (451), 2hops
```

The **debug ipx sap** command generates multiple lines of output for each SAP packet--a packet summary message and a service detail message.

The first line displays the internal router memory address of the packet. The technical support staff may use this information in problem debugging.

```
IPXSAP: at 0023F778:
```

The table below describes the significant fields shown in the display.

Table 84: debug ipx sap Field Descriptions

Field	Description
I	Indicates whether the router received the SAP packet as input (I) or is sending an update as output (O).
SAP Response type 0x2	Packet type. Format is 0x*n*; possible values for *n* include: 1--General query 2--General response 3--Get Nearest Server request 4--Get Nearest Server response
len 160	Length of this packet (in bytes).
src: 160.000.0c00.070d	Source address of the packet.
dest:160.ffff.ffff.ffff	IPX network number and broadcast address of the destination IPX network for which the message is intended.
(452)	I PX socket number of the process sending the packet at the source address. This number is always 452, which is the socket number for the SAP process.

Field	Description
type 0x4	

Field	Description
	Indicates the type of service the server sending the packet provides. Format is 0x*n.* Some of the values for *n* are proprietary to Novell. Those values for *n*that have been published include the following (contact Novell for more information): 0--Unknown 1--User 2--User group 3--Print queue 4--File server 5--Job server 6--Gateway 7--Print server 8--Archive queue 9--Archive server A--Job queue B--Administration 21--NAS SNA gateway 24--Remote bridge server 2D--Time Synchronization VAP 2E--Dynamic SAP 47--Advertising print server 4B--Btrieve VAP 5.0 4C--SQL VAP 7A--TES--NetWare for VMS 98--NetWare access server 9A--Named Pipes server 9E--Portable NetWare--UNIX 111--Test server 166--NetWare management 233--NetWare management agent 237--NetExplorer NLM 239--HMI hub 23A--NetWare LANalyzer agent 26A--NMS management FFFF--Wildcard (any SAP service)

Field	Description
	Contact Novell for more information.
"Hello2"	Name of the server being advertised.
199.0002.0004.0006 (451)	Indicates the network number and address (and socket) of the server generating the SAP packet.
2 hops	Number of hops to the server from the router.

The fifth line of output indicates that the router sent a SAP update to network 160:

```
IPXSAP: sending update to 160
```

The format for **debug ipx sap** output describing a SAP update the router sends is similar to that describing a SAP update the router receives, except that the ssoc: field replaces the src: field, as the following line of output indicates:

O SAP Update type 0x2 len 96 ssoc:0x452 dest:160.ffff.ffff.ffff(452)

The ssoc:0x452 field indicates the IPX socket number of the process sending the packet at the source address. Possible values include the following:

451--Network Core Protocol

452--Service Advertising Protocol

453--Routing Information Protocol

455--NetBIOS

456--Diagnostics

4000 to 6000--Ephemeral sockets used for interaction with file servers and other network communications

Related Commands

Command	Description
debug ipx routing	Displays information on IPX routing packets that the router sends and receives.

debug ipx spoof

To display information about Sequenced Packet Exchange (SPX) keepalive and Internetwork Packet Exchange (IPX) watchdog packets when **ipx watchdog** and **ipx spx-spoof** are configured on the router, use the **debug ipx spoof**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ipx spoof

no debug ipx spoof

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

Use this command to troubleshoot connections that use SPX spoofing when SPX keepalive spoofing is enabled.

Examples

The following is sample output from the **debug ipx spoof** command:

```
Router# debug ipx spoof
```

```
IPX: Tu1:200.0260.8c8d.da75->CC0001.0000.0000.0001 ln= 42 tc=02, SPX: 80 0 7004 4B8 8 1D
23 (new) (changed:yes) Last Changed 0
IPX: Tu1:200.0260.8c8d.c558->CC0001.0000.0000.0001 ln= 42 tc=02, SPX: 80 0 7104 2B8 7 29
2E (new) (changed:yes) Last Changed 0
IPX: Et1:CC0001.0000.0000.0001->200.0260.8c8d.c558 ln= 42 tc=02, SPX: 80 0 2B8 7104 29 7 7
 (early)
IPX: Et1:CC0001.0000.0000.0001->200.0260.8c8d.da75 ln= 42 tc=02, SPX: 80 0 4B8 7004 1D 8 8
 (early)
IPX: Et1:CC0001.0000.0000.0001->200.0260.8c8d.da75 ln= 32 tc=02, watchdog
IPX: local:200.0260.8c8d.da75->CC0001.0000.0000.0001 ln= 32 tc=00, watchdog snet
IPX: Tu1:200.0260.8c8d.da75->CC0001.0000.0000.0001 ln= 42 tc=02, SPX: 80 0 7004 4B8 8 1D
23 (changed:clear) Last Changed 0
IPX: Et1:CC0001.0000.0000.0001->200.0260.8c8d.c558 ln= 42 tc=02, SPX: C0 0 2B8 7104 29 7 7
 (early)
IPX: Tu1:200.0260.8c8d.c558->CC0001.0000.0000.0001 ln= 42 tc=02, SPX: 80 0 7104 2B8 7 29
2E (changed:clear) Last Changed 0
IPX: Et1:CC0001.0000.0000.0001->200.0260.8c8d.c558 ln= 42 tc=02, SPX: C0 0 2B8 7104 29 7 7
 (Last Changed 272 sec)
IPX: local:200.0260.8c8d.c558->CC0001.0000.0000.0001 ln= 42 tc=02, spx keepalive sent 80 0
 7104 2B8 7 29 2E
```

The following lines show that SPX packets were seen, but they are not seen for a connection that exists in the SPX table:

```
IPX: Tu1:200.0260.8c8d.da75->CC0001.0000.0000.0001 ln= 42 tc=02, SPX: 80 0 7004 4B8 8 1D
23 (new) (changed:yes) Last Changed 0
IPX: Tu1:200.0260.8c8d.c558->CC0001.0000.0000.0001 ln= 42 tc=02, SPX: 80 0 7104 2B8 7 29
2E (new) (changed:yes) Last Changed 0
```

The following lines show SPX packets for connections that exist in the SPX table but that SPX idle time has not yet elapsed and spoofing has not started:

```
IPX: Et1:CC0001.0000.0000.0001->200.0260.8c8d.c558 ln= 42 tc=02, SPX: 80 0 2B8 7104 29 7 7
 (early)
IPX: Et1:CC0001.0000.0000.0001->200.0260.8c8d.da75 ln= 42 tc=02, SPX: 80 0 4B8 7004 1D 8 8
 (early)
```

The following lines show an IPX watchdog packet and the spoofed reply:

```
IPX: Et1:CC0001.0000.0000.0001->200.0260.8c8d.da75 ln= 32 tc=02, watchdog
IPX: local:200.0260.8c8d.da75->CC0001.0000.0000.0001 ln= 32 tc=00, watchdog sent
```

The following lines show SPX packets that arrived more than two minutes after spoofing started. This situation occurs when the other sides of the SPX table are cleared. When the table is cleared, the routing processes stop spoofing the connection, which allows SPX keepalives from the local side to travel to the remote side and repopulate the SPX table.

```
IPX: Tu1:200.0260.8c8d.da75->CC0001.0000.0000.0001 ln= 42 tc=02, SPX: 80 0 7004 4B8 8 1D
23 (changed:clear) Last Changed 0
IPX: Et1:CC0001.0000.0000.0001->200.0260.8c8d.c558 ln= 42 tc=02, SPX: C0 0 2B8 7104 29 7 7
 (early)
IPX: Tu1:200.0260.8c8d.c558->CC0001.0000.0000.0001 ln= 42 tc=02, SPX: 80 0 7104 2B8 7 29
2E (changed:clear) Last Changed 0
```

The following lines show that an SPX keepalive packet came in and was spoofed:

```
IPX: Et1:CC0001.0000.0000.0001->200.0260.8c8d.c558 ln= 42 tc=02, SPX: C0 0 2B8 7104 29 7 7
 (Last Changed 272 sec)
IPX: local:200.0260.8c8d.c558->CC0001.0000.0000.0001 ln= 42 tc=02, spx keepalive sent 80 0
 7104 2B8 7 29 2E
```

debug ipx spx

To display debugging messages related to the Sequenced Packet Exchange (SPX) protocol, use the **debug ipx spx**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ipx spx

no debug ipx spx

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

Use the **debug ipx spx** command to display handshaking or negotiating details between the SPX protocol and the other protocols or applications. SPX debugging messages indicate various states of SPX connections such as incoming and outgoing traffic information, timer events, and related processing of SPX connections.

Examples

The following is sample output from the **debug ipx spx**command:

```
Router# debug ipx spx
SPX: Sent an SPX packet
SPX: I Con Src/Dst 776E/20A0 d-strm 0 con-ctl 80
SPX: I Con Src/Dst 776E/20A0 d-strm FE con-ctl 40
SPX: C847C Connection close requested by peer
SPX: Sent an SPX packet
SPX: purge timer fired. Cleaning up C847C
SPX: purging spxcon C847C from conQ
SPX: returning inQ buffers
SPX: returning outQ buffers
SPX: returning unackedQ buffers
SPX: returning spxcon
SPX: I Con Src/Dst 786E/FFFF d-strm 0 con-ctl C0
SPX: new connection request for listening socket
SPX: Sent an SPX packet
SPX: I Con Src/Dst 786E/20B0 d-strm 0 con-ctl 40
SPX: 300 bytes data recvd
SPX: Sent an SPX packet
```

The following line indicates an incoming SPX packet that has a source connection ID of 776E and a destination connection ID of 20A0 (both in hexadecimal). The data stream value in the SPX packet is indicated by *d-strm*, and the connection control value in the SPX packet is indicated by *con-ctl* (both in hexadecimal). All data packets received are followed by an SPX debugging message indicating the size of the packet. All control packets received are consumed internally.

```
SPX: I Con Src/Dst 776E/20A0 d-strm 0 con-ctl 80
```

debug isdn

To display messages about activity in the structure and operation of ISDN in the Cisco IOS software, use the **debug isdn** command in privileged EXEC mode. To disable the ISDN debugging command, use the **no** form of this command.

debug isdn {**all**| **api name**| **cc** [**detail**| **interface** {**bri number**| **serial port/number**}]| **error** [**interface** {**bri number**| **serial port/number**}]| **events**| **mgmnt** [**detail**| **interface** {**bri number**| **serial port/number**}]| **q921**| **q931**| **standard** [**interface** {**bri number**| **serial port/number**}]| **tgrm**}

no debug isdn {**all**| **api name**| **cc** [**detail**| **interface** {**bri number**| **serial port/number**}]| **error** [**interface** {**bri number**| **serial port/number**}]| **events**| **mgmnt** [**detail**| **interface** {**bri number**| **serial port/number**}]| **q921**| **q931**| **standard** [**interface** {**bri number**| **serial port/number**}]| **tgrm**}

Syntax Description

all	Enables all **debug isdn** commands on all interfaces.
api *name*	Enables application programming interfaces (APIs) contained in ISDN on all interfaces. The *name* argument can be any one of the following APIs. The APIs must be entered one per command-line interface (CLI) command. To enable all of the APIs, use the **all** keyword. • **accept** --ISDN call acceptance • **all** --All ISDN API tracing • **bkhl** --ISDN backhaul API tracing • **cdapi** --ISDN API tracing • **csm** --ISDN Compact Subscriber Module API tracing • **l2sock** --ISDN Layer 2 socket API tracing • **nfas** --Non-Facility Associated Signaling • **packet** --ISDN packet API tracing • **qsig** --ISDN PRI Q Signaling API tracing • **rlm** --Redundant Link Manager API tracing
cc	Enables ISDN Call Control debug messages on all interfaces or, optionally, on a specific interface if you use the **interface** keyword. Call Control is a layer of processing within ISDN that is above the Q.931 protocol processing layer, but below the host and API layers.

detail	(Optional) Generates more information during the processing of a specific request.
interface	(Optional) Limits the **debug isdn** capability to one BRI or serial interface.
bri *number*	(Optional) Identifies a single BRI interface number (BRI 2, for example) to which the **debug isdn** command is applied.
serial *port* / *number*	(Optional) Identifies a single serial port and number (serial 1/0, for example) to which the **debug isdn** command is applied. Acceptable values are 0 through 7.
error	Generates error messages for normal exception conditions in the software on all interfaces or on a specific interface if you use the **interface** keyword. The actual significance of the message can be determined only by a detailed examination of surrounding debug messages.
events	Displays ISDN events occurring on the user side of the ISDN interface. See the **debug isdn event s** command page.
mgmnt	Enables ISDN Management Entity messages on all interfaces or, optionally, on a specific interface. Management Entity controls the activation and deactivation of Q.921 resources.
q921	Displays data link layer access procedures that are taking place at the router on the Link Access Protocol D-channel (LAPD) of its ISDN interface. See the **debug isdn q921** command page.
q931	Displays information about call setup and teardown of ISDN network connections between the local router and the network. See the **debug isdn q931** command page.
standard	Enables a selected set of **isdn debug** command messages on all interfaces or, optionally, on a specific interface if you use the **interface** keyword, that should provide sufficient information to determine why a problem is occurring.
tgrm	Displays ISDN trunk group resource manager information. See the **debug isdn tgrm**command page.

Command Default

Commands are enabled on all interfaces unless a specific interface is specified.

Command Modes

Privileged EXEC

Command History

Release	Modification
10.0	This command was introduced.
12.2T	This command was enhanced with the **all api cc error mgmnt**, and **standard** keywords.
12.4(6)T	The **mgmnt** keyword was enhanced to display information about sharing the terminal endpoint identifier (TEI) when the **isdn x25 dchannel q93-broadcast** command is enabled for service access point identifier (SAPI) procedures that accept X.25 calls on the BRI D channel.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

Please read the following caution before using this command.

Caution With the exception of the **debug isdn events**, **debug isdn q921**, **debug isdn q931**, and **debug isdn tgrm** commands, the commands described on this page are not intended for customer use and can cause ISDN or the Cisco IOS software to fail. The **debug isdn events**, **debug isdn q921**, **debug isdn q931**, and **debug isdn tgrm** commands are described on separate command pages.

Follow all instructions from Cisco technical support personnel when enabling and disabling these commands.

Examples

The general format of the **debug isdn** command messages is as follows:

date and time: ISDN interface feature: text message

The text message can be used to determine activity in the structure and operation of ISDN in the Cisco IOS software, ISDN messages, and ISDN signaling procedures. The message must be interpreted by Cisco technical personnel.

The following example shows a typical message for the **debug isdn cc** command:

```
*Mar  1 02:29:27.751: ISDN Se1/0:23 CC: CCPRI_Go: source id 0x300, call id 0x8008, event
0x341 (pre-ccb recovery)
```

The following example enables a selected set of **debug isdn**messages that should provide sufficient information for Cisco technical personnel to determine why a problem is occurring on BRI interface 2:

```
Router# debug isdn standard interface bri 2
```

The following report (highlighted in bold for purpose of example) is displayed when the isdn x25 dchannel q931-broadcast command is used to enable sharing the TEI:

```
Router# debug isdn mgmnt
*Jun 8 22:38:56.535: ISDN BR0 Q921: User TX -> IDREQ ri=29609 ai=127
*Jun 8 22:38:56.595: ISDN BR0 Q921: User RX <- IDASSN ri=29609 ai=86
*Jun 8 22:38:56.595: ISDN BR0 SERROR: L2_Go: at bailout DLCB is NULL
L2: sapi 63 tei 127 ces 0 ev 0x3
*Jun 8 22:38:56.595: ISDN BR0 MGMNT: LM_MDL_UI_DATA_IND: message 2 ri 29609 ai 86 switch
type 9
*Jun 8 22:38:56.595: ISDN BR0 MGMNT: LM_MDL_UI_DATA_IND: OVERLAP REQUEST: ces 9 using lmtr
 tei 85 tei 85
```

debug isdn event

To display ISDN events occurring on the user side (on the router) of the ISDN interface, use the **debug isdn event** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug isdn event

no debug isdn event

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
1x.x(x)	This command was introduced.
12.4(3rd)T	This command was enhanced to display reports about SAPI 0 procedures that accept X.25 calls on the BRI D channel.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

Although the **debug isdn event** and the **debug isdn q931** commands provide similar debug information, the information is displayed in a different format. If you want to see the information in both formats, enable both commands at the same time. The displays will be intermingled.

The ISDN events that can be displayed are Q.931 events (call setup and teardown of ISDN network connections).

Use the **show dialer** command to retrieve information about the status and configuration of the ISDN interface on the router.

Use the **service timestamps debug datetime msec** global configuration command to include the time with each message.

For more information on ISDN switch types, codes, and values, see Appendix B, "ISDN Switch Types, Codes, and Values."

Examples

The following is sample output from the **debug isdn event** command of call setup events for an outgoing call:

```
Router# debug isdn event
ISDN Event: Call to 415555121202
received HOST_PROCEEDING
 Channel ID i = 0x0101
 -------------------
 Channel ID i = 0x89
received HOST_CONNECT
```

```
 Channel ID i = 0x0101
ISDN Event: Connected to 415555121202 on B1 at 64 Kb/s
```

The following shows sample **debug isdn event** output of call setup events for an incoming call. The values used for internal purposes are unpacked information elements. The values that follow the ISDN specification are an interpretation of the unpacked information elements.

```
Router# debug isdn event
received HOST_INCOMING_CALL
 Bearer Capability i = 0x080010
 -------------------
 Channel ID i = 0x0101
 Calling Party Number i = 0x0000, '415555121202'
 IE out of order or end of 'private' IEs --
 Bearer Capability i = 0x8890
 Channel ID i = 0x89
 Calling Party Number i = 0x0083, '415555121202'
ISDN Event: Received a call from 415555121202 on B1 at 64 Kb/s
ISDN Event: Accepting the call
received HOST_CONNECT
 Channel ID i = 0x0101
ISDN Event: Connected to 415555121202 on B1 at 64 Kb/s
```

The following is sample output from the **debug isdn event** command of call teardown events for a call that has been disconnected by the host side of the connection:

```
Router# debug isdn event
received HOST_DISCONNECT
ISDN Event: Call to 415555121202 was hung up
```

The following is sample output from the **debug isdn event** command of a call teardown event for an outgoing or incoming call that has been disconnected by the ISDN interface on the router side:

```
Router# debug isdn event
ISDN Event: Hangup call to call id 0x8008
```

The table below describes the significant fields shown in the display.

Table 85: debug isdn event Field Descriptions

Field	Description
Bearer Capability	Indicates the requested bearer service to be provided by the network. See Table B-4 in Appendix B, "ISDN Switch Types, Codes, and Values."
i=	Indicates the information element identifier. The value depends on the field it is associated with. Refer to the ITU-T Q.931 specification for details about the possible values associated with each field for which this identifier is relevant.

Field	Description
Channel ID	Channel Identifier. The values and corresponding channels might be identified in several ways: • Channel ID i=0x0101--Channel B1 • Channel ID i=0x0102--Channel B2 ITU-T Q.931 defines the values and channels as exclusive or preferred: • Channel ID i=0x83--Any B channel • Channel ID i=0x89--Channel B1 (exclusive) • Channel ID i=0x8A--Channel B2 (exclusive) • Channel ID i=0x81--B1 (preferred) • Channel ID i=0x82--B2 (preferred)
Calling Party Number	Identifies the called party. This field is only present in outgoing calls. The Calling Party Number field uses the IA5 character set. Note that it may be replaced by the Keypad facility field.
IE out of order or end of 'private' IEs	Indicates that an information element identifier is out of order or there are no more private network information element identifiers to interpret.
Received a call from 415555121202 on B1 at 64 Kb/s	Identifies the origin of the call. This field is present only in incoming calls. Note that the information about the incoming call includes the channel and speed. Whether the channel and speed are displayed depends on the network delivering the calling party number.

The following is sample output from the **debug isdn event** command of a call teardown event for a call that has passed call screening and then has been hung up by the ISDN interface on the far end side:

```
Router# debug isdn event
Jan  3 11:29:52.559: ISDN BR0: RX <-  DISCONNECT pd = 8  callref = 0x81
Jan  3 11:29:52.563:         Cause i = 0x8090 - Normal call clearing
```

The following is sample output from the **debug isdn event** command of a call teardown event for a call that has not passed call screening and has been rejected by the ISDN interface on the router side:

```
Router# debug isdn event
Jan  3 11:32:03.263: ISDN BR0: RX <-  DISCONNECT pd = 8  callref = 0x85
Jan  3 11:32:03.267:         Cause i = 0x8095 - Call rejected
```

The following is sample output from the **debug isdn event** command of a call teardown event for an outgoing call that uses a dialer subaddress:

```
Router# debug isdn event
```

```
Jan  3 11:41:48.483: ISDN BR0: Event: Call to 61885:1212 at 64 Kb/s
Jan  3 11:41:48.495: ISDN BR0: TX ->  SETUP pd = 8  callref = 0x04
Jan  3 11:41:48.495:         Bearer Capability i = 0x8890
Jan  3 11:41:48.499:         Channel ID i = 0x83
Jan  3 11:41:48.503:         Called Party Number i = 0x80, '61885'
Jan  3 11:41:48.507:         Called Party SubAddr i = 0x80, 'P1212'
Jan  3 11:41:48.571: ISDN BR0: RX <-  CALL_PROC pd = 8  callref = 0x84
Jan  3 11:41:48.575:         Channel ID i = 0x89
Jan  3 11:41:48.587: ISDN BR0: Event: incoming ces value = 1
Jan  3 11:41:48.587: ISDN BR0: received HOST_PROCEEDING
                        Channel ID i = 0x0101
Jan  3 11:41:48.591:    -------------------
                        Channel ID i = 0x89
Jan  3 11:41:48.731: ISDN BR0: RX <-  CONNECT pd = 8  callref = 0x84
Jan  3 11:41:48.743: ISDN BR0: Event: incoming ces value = 1
Jan  3 11:41:48.743: ISDN BR0: received HOST_CONNECT
                        Channel ID i = 0x0101
Jan  3 11:41:48.747:    -------------------
%LINK-3-UPDOWN: Interface BRI0:1 changed state to up
Jan  3 11:41:48.771: ISDN BR0: Event: Connected to 61885:1212 on B1 at 64 Kb/s
Jan  3 11:41:48.775: ISDN BR0: TX ->  CONNECT_ACK pd = 8  callref = 0x04
%LINEPROTO-5-UPDOWN: Line protocol on Interface BRI0:1, changed state to up
%ISDN-6-CONNECT: Interface BRI0:1 is now connected to 61885:1212 goodie
```

The output is similar to the output of **debug isdn q931**. Refer to the **debug isdn q931**command for detailed field descriptions.

The following is sample output from the **debug isdn event** command of call setup events for a successful callback for legacy DDR:

```
Router# debug isdn event
BRI0:Caller id Callback server starting to spanky 81012345678902
: Callback timer expired
BRI0:beginning callback to spanky 81012345678902
BRI0: Attempting to dial 81012345678902
```

The following is sample output from the **debug isdn event** command for a callback that was unsuccessful because the router had no dialer map for the calling number:

```
Router# debug isdn event
BRI0:Caller id 81012345678902 callback - no matching map
```

The table below describes the significant fields shown in the display.

Table 86: debug isdn event Field Descriptions for Caller ID Callback and Legacy DDR

Field	Description
BRI0:Caller id Callback server starting to ...	Caller ID callback has started, plus host name and number called. The callback enable timer starts now.
: Callback timer expired	Callback timer has expired; callback can proceed.
BRI0:beginning callback to ... BRI0: Attempting to dial ...	Actions proceeding after the callback timer expired, plus host name and number called.

The following is sample output from the **debug isdn event** command for a callback that was successful when the dialer profiles DDR feature is configured:

```
*Mar  1 00:46:51.827: BR0:1:Caller id 81012345678901 matched to profile delorean
*Mar  1 00:46:51.827: Dialer1:Caller id Callback server starting to delorean 81012345678901
*Mar  1 00:46:54.151: : Callback timer expired
*Mar  1 00:46:54.151: Dialer1:beginning callback to delorean 81012345678901
```

```
*Mar  1 00:46:54.155: Freeing callback to delorean 81012345678901
*Mar  1 00:46:54.155: BRI0: Dialing cause Callback return call
*Mar  1 00:46:54.155: BRI0: Attempting to dial 81012345678901
*Mar  1 00:46:54.503: %LINK-3-UPDOWN: Interface BRI0:2, changed state to up
*Mar  1 00:46:54.523: %DIALER-6-BIND: Interface BRI0:2 bound to profile Dialer1
*Mar  1 00:46:55.139: %LINEPROTO-5-UPDOWN: Line protocol on Interface BRI0:2, changed state
 to up
*Mar  1 00:46:58.187: %ISDN-6-CONNECT: Interface BRI0:2 is now connected to 81012345678901
 delorean
```

The following example provides information about accepting X.25 calls on the ISDN D channel (for purpose of example, bold type indicates messages of interest in the following output):

```
Router# debug isdn event
*Sep 28 12:34:29.747: ISDN BR1/1 EVENTd: isdn_host_packet_mode_events: Host packet call
received  call id 0xB
```

The table below describes significant fields of call setup events for a successful callback for the sample output from the **debug isdn event** command when the dialer profiles DDR feature is configured.

Table 87: debug isdn event Field Descriptions for Caller ID Callback and Dialer Profiles

Field	Description
BR0:1:Caller id ... matched to profile ...	Interface, channel number, caller ID that are matched, and the profile to bind to the interface.
: Callback timer expired	Callback timer has expired; callback can proceed.
Dialer1:beginning callback to...	Callback process is beginning to the specified number.
Freeing callback to...	Callback has been started to the specified number, and the number has been removed from the callback list.
BRI0: Dialing cause Callback return call BRI0: Attempting to dial	The reason for the call and the number being dialed.
%LINK-3-UPDOWN: Interface BRI0:2, changed state to up	Interface status: up.
%DIALER-6-BIND: Interface BRI0:2 bound to profile Dialer1	Profile bound to the interface.
%LINEPROTO-5-UPDOWN: Line protocol on Interface BRI0:2, changed state to up	Line protocol status: up.
%ISDN-6-CONNECT: Interface BRI0:2 is now connected to ...	Interface is now connected to the specified host and number.
isdn_host_packet_mode_events: Host packet call received call id 0xB	Host is accepting incoming X.25 call using ITU Q.931 SAPI value 0 procedures.

Related Commands

Command	Description
debug isdn q931	Displays call setup and teardown information of ISDN Layer 3 network connections.

debug isdn q921

To display data link layer (Layer 2) access procedures that are taking place at the router on the D channel (Link Access Procedure or LAPD) of its ISDN interface, use the **debug isdn q921**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug isdn q921 [**detail|** **frame|** **interface** [**bri** *number*]]

no debug isdn q921 [**detail|** **frame|** **interface**]

Syntax Description

detail	(Optional) Displays ISDN Q.921 packet detail.
frame	(Optional) Displays ISDN Q.921 frame contents.
interface	(Optional) Specifies an interface for debugging.
bri *number*	(Optional) Specifies the BRI interface and selects the interface number. Valid values are from 0 to 6.

Command Default

No default behavior or values.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0	This command was introduced.
12.2(15)ZJ	The **detail** and **frame**keywords were added.
12.3(4)T	This command was integrated into Cisco IOS Release 12.3(4)T.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

The ISDN data link layer interface provided by the router conforms to the user interface specification defined by ITU-T recommendation Q.921. The **debug isdn q921** command output is limited to commands and responses exchanged during peer-to-peer communication carried over the D channel. This debug information does not include data transmitted over the B channels that are also part of the router ISDN interface. The peers (data link layer entities and layer management entities on the routers) communicate with each other with an ISDN switch over the D channel.

Note The ISDN switch provides the network interface defined by Q.921. This debug command does not display data link layer access procedures taking place within the ISDN network (that is, procedures taking place on the network side of the ISDN connection). Refer to Appendix B, "ISDN Switch Types, Codes, and Values ," in the *ISDN Switch Types, Codes, and Values* document on Cisco.com for a list of the supported ISDN switch types.

A router can be the calling or called party of the ISDN Q.921 data link layer access procedures. If the router is the calling party, the command displays information about an outgoing call. If the router is the called party, the command displays information about an incoming call and the keepalives.

The **debug isdn q921** command can be used with the **debug isdn event**, **debug isdn q931**, **debug isdn q921 frame**, and **debug isdn q921 detail**commands at the same time. The displays are intermingled.

Use the **service timestamps debug datetime msec** global configuration command to include the time with each message.

Examples

The following is example output for a single active data link connection (DLC). The debugs turned on are **debug isdn q921**, **debug isdn q921 frame**, and **debug isdn q921 detail**. In the debugs below, "Q921" followed by a colon (:) indicates that **debug isdn q921** has been entered. "Q921" followed by the letter "f" indicates that **debug isdn q921 frame** has been entered. "Q921" followed by the letter "d" indicates that **debug isdn q921 detail** has been entered.

The following output shows that the L2 frame is received. The first two octets form the address field; the third octet forms the control field. The address field identifies the originator of a frame and whether it is a command or a response. The second octet of the address field identifies the DLC with which the frame is associated. The control field (third octet) contains the frame type code and sequence number information.

```
00:12:10:ISDN  Q921d:isdn_from_driver_process:QUEUE_EVENT
00:12:10:ISDN Se1:15 Q921f:PBXb RX <- 0x0E03EF
```

The following output interprets the octet information. String "PBXb" indicates that the side receiving (RX) this frame is acting as a PBXb (as opposed to PBXa, which is the other possibility). This example also gives information about the type of frame received (SABMR), the associated DLC (1), the frame type code received from the control field (cntl=SABMR), and the sequence number (indicated by nbit, which is 0 in this case).

```
00:12:10:ISDN Se1:15 Q921:PBXb RX <- SABMR dlci=1 cntl=SABMR nbit=0
```

The following output shows information received from the driver (source_id of x200) showing an L2 frame (event x141). This results from the SABMR frame that was received from the peer PBX (v_bit and chan do not have any significance in this case).

```
00:12:10:ISDN Se1:15 Q921d:process_rxdata:Frame sent to L2
00:12:10:ISDN  Q921d:isdn_from_driver_process:event_count 3
00:12:10:ISDN Se1:15 Q921d:dpnss_l2_main:source_id x200 event x141 v_bit x0 chan x0
```

The following output shows that DPNSS L2 for DLC 1 (chan 1) has received an SABMR frame (event x0) in the IDLE state (s_dpnss_idle):

```
00:12:10:ISDN Se1:15 Q921d:s_dpnss_idle:event x0 chan 1
```

The following output shows that for DLC 1 (chan 1 above), a UA frame (event x1) needs to be sent to the driver (dest x200):

```
00:12:10:ISDN Se1:15 Q921d:dpnss_l2_mail:dest x200 event x1 v_bit 1 chan 1 out_pkt x630531A4
```

The following output shows that for DLC 1, a DL_EST_IND (event x201) needs to be sent to L3 (DUA in this case because of the backhauling) indicating that this DLC is now up (in RESET COMPLETE state):

```
00:12:10:ISDN Se1:15 Q921d:dpnss_l2_mail:dest x300 event x201 v_bit 1 chan 1 out_pkt x0
```

The following output shows that the L2 frame is transmitted (TX):

```
00:12:10:ISDN  Q921d:isdn_l2d_srq_process:QUEUE_EVENT
00:12:10:ISDN Se1:15 Q921f:PBXb TX -> 0x0E0363
```

The following output shows that string “PBXb” is the side transmitting (TX) and that this frame is acting as PBX B. This example also gives information about the associated DLC (1), the frame type code transmitted from the control field (cntl=UA), and the sequence number (indicated by nbit, which is 0 in this case).

```
00:12:10:ISDN Se1:15 Q921:PBXb TX -> UA dlci=1 cntl=UA nbit=0
```

The following is complete debugging output from a DPNSS call:

```
Jan  8 17:24:43.499:ISDN  Q921d:isdn_l2d_srq_process:QUEUE_EVENT
Jan  8 17:24:43.499:ISDN Se2/0:15 Q921f:PBXa TX -> 0x440303
Jan  8 17:24:43.499:ISDN Se2/0:15 Q921:PBXa TX -> UI(R) dlci=1 cntl=UI nbit=0
Jan  8 17:24:43.499:ISDN  Q921d:isdn_l2d_srq_process:event_count 1
Jan  8 17:24:43.503:ISDN  Q921d:isdn_from_driver_process:QUEUE_EVENT
Jan  8 17:24:43.503:ISDN Se2/0:15 Q921f:PBXa RX <-
        0x44030300102A34232A35302A33333330
Jan  8 17:24:43.503:   3030303123343434303030303031
Jan  8 17:24:43.503:ISDN Se2/0:15 Q921:PBXa RX <- UI(C) dlci=1 cntl=UI nbit=0
        i=0x00102A34232A35302A333333303030303123343434303030303031
Jan  8 17:24:43.503:ISDN Se2/0:15 Q921d:process_rxdata:Frame sent to L2
Jan  8 17:24:43.503:ISDN  Q921d:isdn_from_driver_process:event_count 1
Jan  8 17:24:43.507:ISDN Se2/0:15 Q921d:dpnss_l2_main:source_id x200 event
        x141 v_bit x0 chan x0
Jan  8 17:24:43.507:ISDN Se2/0:15 Q921d:s_dpnss_information_transfer:event x2
        chan 1
Jan  8 17:24:43.507:ISDN Se2/0:15 Q921d:dpnss_l2_mail:dest x200 event x3
        v_bit 1 chan 1 out_pkt x63F183D4
Jan  8 17:24:43.507:ISDN  Q921d:isdn_l2d_srq_process:QUEUE_EVENT
Jan  8 17:24:43.507:ISDN Se2/0:15 Q921f:PBXa TX -> 0x440303
Jan  8 17:24:43.507:ISDN Se2/0:15 Q921:PBXa TX -> UI(R) dlci=1 cntl=UI nbit=0
Jan  8 17:24:43.507:ISDN  Q921d:isdn_l2d_srq_process:event_count 1
Jan  8 17:24:43.515:ISDN  Q921d:isdn_from_driver_process:QUEUE_EVENT
Jan  8 17:24:43.515:ISDN Se2/0:15 Q921f:PBXa RX <-
        0x44030300102A34232A35302A33333330
Jan  8 17:24:43.515:   3030303123343434303030303031
Jan  8 17:24:43.515:ISDN Se2/0:15 Q921:PBXa RX <- UI(C) dlci=1 cntl=UI nbit=0
        i=0x00102A34232A35302A333333303030303123343434303030303031
Jan  8 17:24:43.515:ISDN Se2/0:15 Q921d:process_rxdata:Frame sent to L2
Jan  8 17:24:43.515:ISDN  Q921d:isdn_from_driver_process:event_count 1
Jan  8 17:24:43.515:ISDN Se2/0:15 Q921d:dpnss_l2_main:source_id x200 event
        x141 v_bit x0 chan x0
Jan  8 17:24:43.515:ISDN Se2/0:15 Q921d:s_dpnss_information_transfer:event x2
        chan 1
Jan  8 17:24:43.515:ISDN Se2/0:15 Q921d:dpnss_l2_mail:dest x200 event x3
        v_bit 1 chan 1 out_pkt x63F183D4
Jan  8 17:24:43.515:ISDN  Q921d:isdn_l2d_srq_process:QUEUE_EVENT
Jan  8 17:24:43.519:ISDN Se2/0:15 Q921f:PBXa TX -> 0x440303
Jan  8 17:24:43.519:ISDN Se2/0:15 Q921:PBXa TX -> UI(R) dlci=1 cntl=UI nbit=0
Jan  8 17:24:43.519:ISDN  Q921d:isdn_l2d_srq_process:event_count 1
Jan  8 17:24:43.599:ISDN Se2/1:15 Q921d:dpnss_l2_main:source_id x4 event x240
        v_bit x0 chan x2
Jan  8 17:24:43.599:ISDN Se2/1:15 Q921d:s_dpnss_information_transfer:event
        x240 chan 1
Jan  8 17:24:43.599:ISDN Se2/1:15 Q921d:dpnss_l2_mail:dest x200 event x2
        v_bit 1 chan 1 out_pkt x63EE5780
Jan  8 17:24:43.599:ISDN Se2/1:15 LIFd:LIF_StartTimer:timer (0x63E569A8),
        ticks (500), event (0x1201)
Jan  8 17:24:43.599:ISDN  Q921d:isdn_l2d_srq_process:QUEUE_EVENT
Jan  8 17:24:43.599:ISDN Se2/1:15 Q921f:PBXa TX ->
        0x46030300102A31232A35302A33333330
Jan  8 17:24:43.599:   3030303123343434303030303031
Jan  8 17:24:43.599:ISDN Se2/1:15 Q921:PBXa TX -> UI(C) dlci=1 cntl=UI nbit=0
```

```
    i=0x00102A31232A35302A333333303030303123343434303030303031
Jan  8 17:24:43.599:ISDN  Q921d:isdn_l2d_srq_process:event_count 1
Jan  8 17:24:43.623:ISDN  Q921d:isdn_from_driver_process:QUEUE_EVENT
Jan  8 17:24:43.623:ISDN Se2/1:15 Q921f:PBXa RX <- 0x460303
Jan  8 17:24:43.623:ISDN Se2/1:15 Q921:PBXa RX <- UI(R) dlci=1 cntl=UI nbit=0
Jan  8 17:24:43.623:ISDN Se2/1:15 Q921d:process_rxdata:Frame sent to L2
Jan  8 17:24:43.623:ISDN  Q921d:isdn_from_driver_process:event_count 1
Jan  8 17:24:43.627:ISDN Se2/1:15 Q921d:dpnss_l2_main:source_id x200 event
    x141 v_bit x0 chan x0
Jan  8 17:24:43.627:ISDN Se2/1:15 Q921d:s_dpnss_information_transfer:event x3
    chan 1
Jan  8 17:24:43.719:ISDN  Q921d:isdn_from_driver_process:QUEUE_EVENT
Jan  8 17:24:43.719:ISDN Se2/1:15 Q921f:PBXa RX <-
    0x440313092A34232A35302A3434343030
Jan  8 17:24:43.719:   303031232A31382A33312A33312A3331
Jan  8 17:24:43.719:   23
Jan  8 17:24:43.719:ISDN Se2/1:15 Q921:PBXa RX <- UI(C) dlci=1 cntl=UI nbit=1
    i=0x092A34232A35302A3434343030303031232A31382A33312A33312A333123
Jan  8 17:24:43.719:ISDN Se2/1:15 Q921d:process_rxdata:Frame sent to L2
Jan  8 17:24:43.719:ISDN  Q921d:isdn_from_driver_process:event_count 1
Jan  8 17:24:43.719:ISDN Se2/1:15 Q921d:dpnss_l2_main:source_id x200 event
    x141 v_bit x0 chan x0
Jan  8 17:24:43.719:ISDN Se2/1:15 Q921d:s_dpnss_information_transfer:event x2
    chan 1
Jan  8 17:24:43.719:ISDN Se2/1:15 Q921d:dpnss_l2_mail:dest x300 event x241
    v_bit 1 chan 1 out_pkt x63EE5780
Jan  8 17:24:43.719:ISDN Se2/1:15 Q921d:dpnss_l2_mail:dest x200 event x3
    v_bit 1 chan 1 out_pkt x63EE57CC
Jan  8 17:24:43.723:ISDN  Q921d:isdn_l2d_srq_process:QUEUE_EVENT
Jan  8 17:24:43.723:ISDN Se2/1:15 Q921f:PBXa TX -> 0x440313
Jan  8 17:24:43.723:ISDN Se2/1:15 Q921:PBXa TX -> UI(R) dlci=1 cntl=UI nbit=1
Jan  8 17:24:43.723:ISDN  Q921d:isdn_l2d_srq_process:event_count 1
Jan  8 17:24:43.727:ISDN  Q921d:isdn_from_driver_process:QUEUE_EVENT
Jan  8 17:24:43.727:ISDN Se2/1:15 Q921f:PBXa RX <-
    0x440313092A34232A35302A3434343030
Jan  8 17:24:43.727:   303031232A31382A33312A33312A3331
Jan  8 17:24:43.727:   23
Jan  8 17:24:43.727:ISDN Se2/1:15 Q921:PBXa RX <- UI(C) dlci=1 cntl=UI nbit=1
    i=0x092A34232A35302A3434343030303031232A31382A33312A33312A333123
Jan  8 17:24:43.727:ISDN Se2/1:15 Q921d:process_rxdata:Frame sent to L2
Jan  8 17:24:43.727:ISDN  Q921d:isdn_from_driver_process:event_count 1
Jan  8 17:24:43.731:ISDN Se2/1:15 Q921d:dpnss_l2_main:source_id x200 event
    x141 v_bit x0 chan x0
Jan  8 17:24:43.731:ISDN Se2/1:15 Q921d:s_dpnss_information_transfer:event x2
    chan 1
Jan  8 17:24:43.731:ISDN Se2/1:15 Q921d:dpnss_l2_mail:dest x200 event x3
    v_bit 1 chan 1 out_pkt x63EE57CC
Jan  8 17:24:43.731:ISDN  Q921d:isdn_l2d_srq_process:QUEUE_EVENT
Jan  8 17:24:43.731:ISDN Se2/1:15 Q921f:PBXa TX -> 0x440313
Jan  8 17:24:43.731:ISDN Se2/1:15 Q921:PBXa TX -> UI(R) dlci=1 cntl=UI nbit=1
Jan  8 17:24:43.731:ISDN  Q921d:isdn_l2d_srq_process:event_count 1
Jan  8 17:24:43.739:ISDN  Q921d:isdn_from_driver_process:QUEUE_EVENT
Jan  8 17:24:43.739:ISDN Se2/1:15 Q921f:PBXa RX <-
    0x440313092A34232A35302A3434343030
Jan  8 17:24:43.739:   303031232A31382A33312A33312A3331
Jan  8 17:24:43.739:   23
Jan  8 17:24:43.739:ISDN Se2/1:15 Q921:PBXa RX <- UI(C) dlci=1 cntl=UI nbit=1
    i=0x092A34232A35302A3434343030303031232A31382A33312A33312A333123
Jan  8 17:24:43.739:ISDN Se2/1:15 Q921d:process_rxdata:Frame sent to L2
Jan  8 17:24:43.739:ISDN  Q921d:isdn_from_driver_process:event_count 1
Jan  8 17:24:43.739:ISDN Se2/1:15 Q921d:dpnss_l2_main:source_id x200 event
    x141 v_bit x0 chan x0
Jan  8 17:24:43.739:ISDN Se2/1:15 Q921d:s_dpnss_information_transfer:event x2
    chan 1
Jan  8 17:24:43.739:ISDN Se2/1:15 Q921d:dpnss_l2_mail:dest x200 event x3
    v_bit 1 chan 1 out_pkt x63EE57CC
Jan  8 17:24:43.739:ISDN  Q921d:isdn_l2d_srq_process:QUEUE_EVENT
Jan  8 17:24:43.743:ISDN Se2/1:15 Q921f:PBXa TX -> 0x440313
Jan  8 17:24:43.743:ISDN Se2/1:15 Q921:PBXa TX -> UI(R) dlci=1 cntl=UI nbit=1
Jan  8 17:24:43.743:ISDN  Q921d:isdn_l2d_srq_process:event_count 1
Jan  8 17:24:43.787:ISDN Se2/0:15 Q921d:dpnss_l2_main:source_id x4 event x240
    v_bit x0 chan x2
Jan  8 17:24:43.787:ISDN Se2/0:15 Q921d:s_dpnss_information_transfer:event
```

```
        x240 chan 1
Jan  8 17:24:43.787:ISDN Se2/0:15 Q921d:dpnss_l2_mail:dest x200 event x2
        v_bit 1 chan 1 out_pkt x636B1B64
Jan  8 17:24:43.787:ISDN Se2/0:15 LIFd:LIF_StartTimer:timer (0x63A4AFBC),
        ticks (500), event (0x1201)
Jan  8 17:24:43.791:ISDN  Q921d:isdn_l2d_srq_process:QUEUE_EVENT
Jan  8 17:24:43.791:ISDN Se2/0:15 Q921f:PBXa TX ->
        0x460313092A31232A35302A3434343030
Jan  8 17:24:43.791:   30303123
Jan  8 17:24:43.791:ISDN Se2/0:15 Q921:PBXa TX -> UI(C) dlci=1 cntl=UI nbit=1
        i=0x092A31232A35302A34343430303030303123
Jan  8 17:24:43.791:ISDN  Q921d:isdn_l2d_srq_process:event_count 1
Jan  8 17:24:43.811:ISDN  Q921d:isdn_from_driver_process:QUEUE_EVENT
Jan  8 17:24:43.811:ISDN Se2/0:15 Q921f:PBXa RX <- 0x460313
Jan  8 17:24:43.811:ISDN Se2/0:15 Q921:PBXa RX <- UI(R) dlci=1 cntl=UI nbit=1
Jan  8 17:24:43.811:ISDN Se2/0:15 Q921d:process_rxdata:Frame sent to L2
Jan  8 17:24:43.811:ISDN  Q921d:isdn_from_driver_process:event_count 1
Jan  8 17:24:43.811:ISDN Se2/0:15 Q921d:dpnss_l2_main:source_id x200 event
        x141 v_bit x0 chan x0
Jan  8 17:24:43.811:ISDN Se2/0:15 Q921d:s_dpnss_information_transfer:event x3
        chan 1
Jan  8 17:24:52.107:ISDN  Q921d:isdn_from_driver_process:QUEUE_EVENT
Jan  8 17:24:52.107:ISDN Se2/1:15 Q921f:PBXa RX <-
        0x440303052A34232A35302A3434343030
Jan  8 17:24:52.107:   303031232A31382A33312A33312A3331
Jan  8 17:24:52.107:   23
Jan  8 17:24:52.107:ISDN Se2/1:15 Q921:PBXa RX <- UI(C) dlci=1 cntl=UI nbit=0
        i=0x052A34232A35302A343434303030303031232A31382A33312A33312A333123
Jan  8 17:24:52.107:ISDN Se2/1:15 Q921d:process_rxdata:Frame sent to L2
Jan  8 17:24:52.107:ISDN  Q921d:isdn_from_driver_process:event_count 1
Jan  8 17:24:52.111:ISDN Se2/1:15 Q921d:dpnss_l2_main:source_id x200 event
        x141 v_bit x0 chan x0
Jan  8 17:24:52.111:ISDN Se2/1:15 Q921d:s_dpnss_information_transfer:event x2
        chan 1
Jan  8 17:24:52.111:ISDN Se2/1:15 Q921d:dpnss_l2_mail:dest x300 event x241
        v_bit 1 chan 1 out_pkt x63F19CC8
Jan  8 17:24:52.111:ISDN Se2/1:15 Q921d:dpnss_l2_mail:dest x200 event x3
        v_bit 1 chan 1 out_pkt x63F19D14
Jan  8 17:24:52.111:ISDN  Q921d:isdn_l2d_srq_process:QUEUE_EVENT
Jan  8 17:24:52.111:ISDN Se2/1:15 Q921f:PBXa TX -> 0x440303
Jan  8 17:24:52.111:ISDN Se2/1:15 Q921:PBXa TX -> UI(R) dlci=1 cntl=UI nbit=0
Jan  8 17:24:52.111:ISDN  Q921d:isdn_l2d_srq_process:event_count 1
Jan  8 17:24:52.119:ISDN  Q921d:isdn_from_driver_process:QUEUE_EVENT
Jan  8 17:24:52.119:ISDN Se2/1:15 Q921f:PBXa RX <-
        0x440303052A34232A35302A3434343030
Jan  8 17:24:52.119:   303031232A31382A33312A33312A3331
Jan  8 17:24:52.119:   23
Jan  8 17:24:52.119:ISDN Se2/1:15 Q921:PBXa RX <- UI(C) dlci=1 cntl=UI nbit=0
        i=0x052A34232A35302A343434303030303031232A31382A33312A33312A333123
Jan  8 17:24:52.119:ISDN Se2/1:15 Q921d:process_rxdata:Frame sent to L2
Jan  8 17:24:52.119:ISDN  Q921d:isdn_from_driver_process:event_count 1
Jan  8 17:24:52.119:ISDN Se2/1:15 Q921d:dpnss_l2_main:source_id x200 event
        x141 v_bit x0 chan x0
        x141 v_bit x0 chan x0
Jan  8 17:24:52.119:ISDN Se2/1:15 Q921d:s_dpnss_information_transfer:event x2
        chan 1
Jan  8 17:24:52.119:ISDN Se2/1:15 Q921d:dpnss_l2_mail:dest x200 event x3
        v_bit 1 chan 1 out_pkt x63F19D14
Jan  8 17:24:52.119:ISDN  Q921d:isdn_l2d_srq_process:QUEUE_EVENT
Jan  8 17:24:52.123:ISDN Se2/1:15 Q921f:PBXa TX -> 0x440303
Jan  8 17:24:52.123:ISDN Se2/1:15 Q921:PBXa TX -> UI(R) dlci=1 cntl=UI nbit=0
Jan  8 17:24:52.123:ISDN  Q921d:isdn_l2d_srq_process:event_count 1
Jan  8 17:24:52.127:ISDN  Q921d:isdn_from_driver_process:QUEUE_EVENT
Jan  8 17:24:52.127:ISDN Se2/1:15 Q921f:PBXa RX <-
        0x440303052A34232A35302A3434343030
Jan  8 17:24:52.127:   303031232A31382A33312A33312A3331
Jan  8 17:24:52.127:   23
Jan  8 17:24:52.127:ISDN Se2/1:15 Q921:PBXa RX <- UI(C) dlci=1 cntl=UI nbit=0
        i=0x052A34232A35302A343434303030303031232A31382A33312A33312A333123
Jan  8 17:24:52.127:ISDN Se2/1:15 Q921d:process_rxdata:Frame sent to L2
Jan  8 17:24:52.127:ISDN  Q921d:isdn_from_driver_process:event_count 1
Jan  8 17:24:52.131:ISDN Se2/1:15 Q921d:dpnss_l2_main:source_id x200 event
        x141 v_bit x0 chan x0
```

```
Jan  8 17:24:52.131:ISDN Se2/1:15 Q921d:s_dpnss_information_transfer:event x2
      chan 1
Jan  8 17:24:52.131:ISDN Se2/1:15 Q921d:dpnss_l2_mail:dest x200 event x3
      v_bit 1 chan 1 out_pkt x63F19D14
Jan  8 17:24:52.131:ISDN  Q921d:isdn_l2d_srq_process:QUEUE_EVENT
Jan  8 17:24:52.131:ISDN Se2/1:15 Q921f:PBXa TX -> 0x440303
Jan  8 17:24:52.131:ISDN Se2/1:15 Q921:PBXa TX -> UI(R) dlci=1 cntl=UI nbit=0
Jan  8 17:24:52.131:ISDN  Q921d:isdn_l2d_srq_process:event_count 1
Jan  8 17:24:52.159:ISDN Se2/0:15 Q921d:dpnss_l2_main:source_id x4 event x240
      v_bit x0 chan x2
Jan  8 17:24:52.159:ISDN Se2/0:15 Q921d:s_dpnss_information_transfer:event
      x240 chan 1
Jan  8 17:24:52.159:ISDN Se2/0:15 Q921d:dpnss_l2_mail:dest x200 event x2
      v_bit 1 chan 1 out_pkt x63F19CC8
Jan  8 17:24:52.159:ISDN Se2/0:15 LIFd:LIF_StartTimer:timer (0x63A4AFBC),
      ticks (500), event (0x1201)
Jan  8 17:24:52.159:ISDN  Q921d:isdn_l2d_srq_process:QUEUE_EVENT
Jan  8 17:24:52.159:ISDN Se2/0:15 Q921f:PBXa TX ->
      0x460303052A35302A343434303030303031
Jan  8 17:24:52.159:   23
Jan  8 17:24:52.159:ISDN Se2/0:15 Q921:PBXa TX -> UI(C) dlci=1 cntl=UI nbit=0
      i=0x052A35302A34343430303030303123
Jan  8 17:24:52.159:ISDN  Q921d:isdn_l2d_srq_process:event_count 1
Jan  8 17:24:52.179:ISDN  Q921d:isdn_from_driver_process:QUEUE_EVENT
Jan  8 17:24:52.179:ISDN Se2/0:15 Q921f:PBXa RX <- 0x460303
Jan  8 17:24:52.179:ISDN Se2/0:15 Q921:PBXa RX <- UI(R) dlci=1 cntl=UI nbit=0
Jan  8 17:24:52.179:ISDN Se2/0:15 Q921d:process_rxdata:Frame sent to L2
Jan  8 17:24:52.183:ISDN  Q921d:isdn_from_driver_process:event_count 1
Jan  8 17:24:52.183:ISDN Se2/0:15 Q921d:dpnss_l2_main:source_id x200 event
      x141 v_bit x0 chan x0
Jan  8 17:24:52.183:ISDN Se2/0:15 Q921d:s_dpnss_information_transfer:event x3
      chan 1
Jan  8 17:25:31.811:ISDN  Q921d:isdn_from_driver_process:QUEUE_EVENT
Jan  8 17:25:31.811:ISDN Se2/0:15 Q921f:PBXa RX <- 0x4403130830
Jan  8 17:25:31.811:ISDN Se2/0:15 Q921:PBXa RX <- UI(C) dlci=1 cntl=UI nbit=1
      i=0x0830
Jan  8 17:25:31.811:ISDN Se2/0:15 Q921d:process_rxdata:Frame sent to L2
Jan  8 17:25:31.811:ISDN  Q921d:isdn_from_driver_process:event_count 1
Jan  8 17:25:31.811:ISDN Se2/0:15 Q921d:dpnss_l2_main:source_id x200 event
      x141 v_bit x0 chan x0
Jan  8 17:25:31.811:ISDN Se2/0:15 Q921d:s_dpnss_information_transfer:event x2
      chan 1
Jan  8 17:25:31.811:ISDN Se2/0:15 Q921d:dpnss_l2_mail:dest x300 event x241
      v_bit 1 chan 1 out_pkt x63F1806C
Jan  8 17:25:31.811:ISDN Se2/0:15 Q921d:dpnss_l2_mail:dest x200 event x3
      v_bit 1 chan 1 out_pkt x636710B8
Jan  8 17:25:31.815:ISDN  Q921d:isdn_l2d_srq_process:QUEUE_EVENT
Jan  8 17:25:31.815:ISDN Se2/0:15 Q921f:PBXa TX -> 0x440313
Jan  8 17:25:31.815:ISDN Se2/0:15 Q921:PBXa TX -> UI(R) dlci=1 cntl=UI nbit=1
Jan  8 17:25:31.815:ISDN  Q921d:isdn_l2d_srq_process:event_count 1
Jan  8 17:25:31.819:ISDN  Q921d:isdn_from_driver_process:QUEUE_EVENT
Jan  8 17:25:31.819:ISDN Se2/0:15 Q921f:PBXa RX <- 0x4403130830
Jan  8 17:25:31.819:ISDN Se2/0:15 Q921:PBXa RX <- UI(C) dlci=1 cntl=UI nbit=1
      i=0x0830
Jan  8 17:25:31.819:ISDN Se2/0:15 Q921d:process_rxdata:Frame sent to L2
Jan  8 17:25:31.819:ISDN  Q921d:isdn_from_driver_process:event_count 1
Jan  8 17:25:31.823:ISDN Se2/0:15 Q921d:dpnss_l2_main:source_id x200 event
      x141 v_bit x0 chan x0
Jan  8 17:25:31.823:ISDN Se2/0:15 Q921d:s_dpnss_information_transfer:event x2
      chan 1
Jan  8 17:25:31.823:ISDN Se2/0:15 Q921d:dpnss_l2_mail:dest x200 event x3
      v_bit 1 chan 1 out_pkt x63F19CC8
Jan  8 17:25:31.823:ISDN  Q921d:isdn_l2d_srq_process:QUEUE_EVENT
Jan  8 17:25:31.823:ISDN Se2/0:15 Q921f:PBXa TX -> 0x440313
Jan  8 17:25:31.823:ISDN Se2/0:15 Q921:PBXa TX -> UI(R) dlci=1 cntl=UI nbit=1
Jan  8 17:25:31.823:ISDN  Q921d:isdn_l2d_srq_process:event_count 1
Jan  8 17:25:31.831:ISDN  Q921d:isdn_from_driver_process:QUEUE_EVENT
Jan  8 17:25:31.831:ISDN Se2/0:15 Q921f:PBXa RX <- 0x4403130830
Jan  8 17:25:31.831:ISDN Se2/0:15 Q921:PBXa RX <- UI(C) dlci=1 cntl=UI nbit=1
      i=0x0830
Jan  8 17:25:31.831:ISDN Se2/0:15 Q921d:process_rxdata:Frame sent to L2
Jan  8 17:25:31.831:ISDN  Q921d:isdn_from_driver_process:event_count 1
Jan  8 17:25:31.831:ISDN Se2/0:15 Q921d:dpnss_l2_main:source_id x200 event
```

```
        x141 v_bit x0 chan x0
Jan  8 17:25:31.831:ISDN Se2/0:15 Q921d:s_dpnss_information_transfer:event x2
        chan 1
Jan  8 17:25:31.831:ISDN Se2/0:15 Q921d:dpnss_l2_mail:dest x200 event x3
        v_bit 1 chan 1 out_pkt x636710B8
Jan  8 17:25:31.835:ISDN  Q921d:isdn_l2d_srq_process:QUEUE_EVENT
Jan  8 17:25:31.835:ISDN Se2/0:15 Q921f:PBXa TX -> 0x440313
Jan  8 17:25:31.835:ISDN Se2/0:15 Q921:PBXa TX -> UI(R) dlci=1 cntl=UI nbit=1
Jan  8 17:25:31.835:ISDN  Q921d:isdn_l2d_srq_process:event_count 1
Jan  8 17:25:31.851:ISDN Se2/1:15 Q921d:dpnss_l2_main:source_id x4 event x240
        v_bit x0 chan x2
Jan  8 17:25:31.851:ISDN Se2/1:15 Q921d:s_dpnss_information_transfer:event
        x240 chan 1
Jan  8 17:25:31.851:ISDN Se2/1:15 Q921d:dpnss_l2_mail:dest x200 event x2
        v_bit 1 chan 1 out_pkt x63F1806C
Jan  8 17:25:31.851:ISDN Se2/1:15 LIFd:LIF_StartTimer:timer (0x63E569A8),
        ticks (500), event (0x1201)
Jan  8 17:25:31.851:ISDN  Q921d:isdn_l2d_srq_process:QUEUE_EVENT
Jan  8 17:25:31.855:ISDN Se2/1:15 Q921f:PBXa TX -> 0x4603130830
Jan  8 17:25:31.855:ISDN Se2/1:15 Q921:PBXa TX -> UI(C) dlci=1 cntl=UI nbit=1
        i=0x0830
Jan  8 17:25:31.855:ISDN  Q921d:isdn_l2d_srq_process:event_count 1
Jan  8 17:25:31.875:ISDN  Q921d:isdn_from_driver_process:QUEUE_EVENT
Jan  8 17:25:31.875:ISDN Se2/1:15 Q921f:PBXa RX <- 0x460313
Jan  8 17:25:31.875:ISDN Se2/1:15 Q921:PBXa RX <- UI(R) dlci=1 cntl=UI nbit=1
Jan  8 17:25:31.875:ISDN Se2/1:15 Q921d:process_rxdata:Frame sent to L2
Jan  8 17:25:31.875:ISDN  Q921d:isdn_from_driver_process:event_count 1
Jan  8 17:25:31.875:ISDN Se2/1:15 Q921d:dpnss_l2_main:source_id x200 event
        x141 v_bit x0 chan x0
Jan  8 17:25:31.875:ISDN Se2/1:15 Q921d:s_dpnss_information_transfer:event x3
        chan 1
Jan  8 17:25:31.879:ISDN Se2/0:15 Q921d:dpnss_l2_main:source_id x4 event x240
        v_bit x0 chan x2
Jan  8 17:25:31.879:ISDN Se2/0:15 Q921d:s_dpnss_information_transfer:event
        x240 chan 1
Jan  8 17:25:31.879:ISDN Se2/0:15 Q921d:dpnss_l2_mail:dest x200 event x2
        v_bit 1 chan 1 out_pkt x63EFC5AC
Jan  8 17:25:31.879:ISDN Se2/0:15 LIFd:LIF_StartTimer:timer (0x63A4AFBC),
        ticks (500), event (0x1201)
Jan  8 17:25:31.879:ISDN  Q921d:isdn_l2d_srq_process:QUEUE_EVENT
Jan  8 17:25:31.879:ISDN Se2/0:15 Q921f:PBXa TX -> 0x4603130830
Jan  8 17:25:31.879:ISDN Se2/0:15 Q921:PBXa TX -> UI(C) dlci=1 cntl=UI nbit=1
        i=0x0830
Jan  8 17:25:31.883:ISDN  Q921d:isdn_l2d_srq_process:event_count 1
Jan  8 17:25:31.899:ISDN  Q921d:isdn_from_driver_process:QUEUE_EVENT
Jan  8 17:25:31.899:ISDN Se2/0:15 Q921f:PBXa RX <- 0x460313
Jan  8 17:25:31.899:ISDN Se2/0:15 Q921:PBXa RX <- UI(R) dlci=1 cntl=UI nbit=1
Jan  8 17:25:31.899:ISDN Se2/0:15 Q921d:process_rxdata:Frame sent to L2
Jan  8 17:25:31.899:ISDN  Q921d:isdn_from_driver_process:event_count 1
Jan  8 17:25:31.903:ISDN Se2/0:15 Q921d:dpnss_l2_main:source_id x200 event
        x141 v_bit x0 chan x0
Jan  8 17:25:31.903:ISDN Se2/0:15 Q921d:s_dpnss_information_transfer:event x3
        chan 1
Jan  8 17:25:32.063:ISDN  Q921d:isdn_from_driver_process:QUEUE_EVENT
Jan  8 17:25:32.063:ISDN Se2/1:15 Q921f:PBXa RX <- 0x4403130830
Jan  8 17:25:32.063:ISDN Se2/1:15 Q921:PBXa RX <- UI(C) dlci=1 cntl=UI nbit=1
        i=0x0830
Jan  8 17:25:32.063:ISDN Se2/1:15 Q921d:process_rxdata:Frame sent to L2
Jan  8 17:25:32.063:ISDN  Q921d:isdn_from_driver_process:event_count 1
Jan  8 17:25:32.067:ISDN Se2/1:15 Q921d:dpnss_l2_main:source_id x200 event
        x141 v_bit x0 chan x0
Jan  8 17:25:32.067:ISDN Se2/1:15 Q921d:s_dpnss_information_transfer:event x2
        chan 1
Jan  8 17:25:32.067:ISDN Se2/1:15 Q921d:dpnss_l2_mail:dest x300 event x241
        v_bit 1 chan 1 out_pkt x63EFC5AC
Jan  8 17:25:32.067:ISDN Se2/1:15 Q921d:dpnss_l2_mail:dest x200 event x3
        v_bit 1 chan 1 out_pkt x6367175C
Jan  8 17:25:32.067:ISDN  Q921d:isdn_l2d_srq_process:QUEUE_EVENT
Jan  8 17:25:32.067:ISDN Se2/1:15 Q921f:PBXa TX -> 0x440313
Jan  8 17:25:32.067:ISDN Se2/1:15 Q921:PBXa TX -> UI(R) dlci=1 cntl=UI nbit=1
Jan  8 17:25:32.067:ISDN  Q921d:isdn_l2d_srq_process:event_count 1
Jan  8 17:25:32.075:ISDN  Q921d:isdn_from_driver_process:QUEUE_EVENT
Jan  8 17:25:32.075:ISDN Se2/1:15 Q921f:PBXa RX <- 0x4403130830
```

```
Jan  8 17:25:32.075:ISDN Se2/1:15 Q921:PBXa RX <- UI(C) dlci=1 cntl=UI nbit=1
        i=0x0830
Jan  8 17:25:32.075:ISDN Se2/1:15 Q921d:process_rxdata:Frame sent to L2
Jan  8 17:25:32.075:ISDN  Q921d:isdn_from_driver_process:event_count 1
Jan  8 17:25:32.075:ISDN Se2/1:15 Q921d:dpnss_l2_main:source_id x200 event
        x141 v_bit x0 chan x0
Jan  8 17:25:32.075:ISDN Se2/1:15 Q921d:s_dpnss_information_transfer:event x2
        chan 1
Jan  8 17:25:32.075:ISDN Se2/1:15 Q921d:dpnss_l2_mail:dest x200 event x3
        v_bit 1 chan 1 out_pkt x6367175C
Jan  8 17:25:32.075:ISDN  Q921d:isdn_l2d_srq_process:QUEUE_EVENT
Jan  8 17:25:32.075:ISDN Se2/1:15 Q921f:PBXa TX -> 0x440313
Jan  8 17:25:32.079:ISDN Se2/1:15 Q921:PBXa TX -> UI(R) dlci=1 cntl=UI nbit=1
Jan  8 17:25:32.079:ISDN  Q921d:isdn_l2d_srq_process:event_count 1
Jan  8 17:25:32.083:ISDN  Q921d:isdn_from_driver_process:QUEUE_EVENT
Jan  8 17:25:32.083:ISDN Se2/1:15 Q921f:PBXa RX <- 0x4403130830
Jan  8 17:25:32.083:ISDN Se2/1:15 Q921:PBXa RX <- UI(C) dlci=1 cntl=UI nbit=1
        i=0x0830
Jan  8 17:25:32.083:ISDN Se2/1:15 Q921d:process_rxdata:Frame sent to L2
Jan  8 17:25:32.083:ISDN  Q921d:isdn_from_driver_process:event_count 1
Jan  8 17:25:32.087:ISDN Se2/1:15 Q921d:dpnss_l2_main:source_id x200 event
        x141 v_bit x0 chan x0
Jan  8 17:25:32.087:ISDN Se2/1:15 Q921d:s_dpnss_information_transfer:event x2
        chan 1
Jan  8 17:25:32.087:ISDN Se2/1:15 Q921d:dpnss_l2_mail:dest x200 event x3
        v_bit 1 chan 1 out_pkt x6367175C
Jan  8 17:25:32.087:ISDN  Q921d:isdn_l2d_srq_process:QUEUE_EVENT
Jan  8 17:25:32.087:ISDN Se2/1:15 Q921f:PBXa TX -> 0x440313
Jan  8 17:25:32.087:ISDN Se2/1:15 Q921:PBXa TX -> UI(R) dlci=1 cntl=UI nbit=1
Jan  8 17:25:32.087:ISDN  Q921d:isdn_l2d_srq_process:event_count 1
```

The following output shows details of the preceding debugging events.

The first two octets (0x4403) form the address field, while the third octet (0x03) is the control field. All the octets starting from the fourth constitute DPNSS L3 information, which needs to be backhauled to the Cisco PGW2200.

```
Jan  8 17:24:43.495:ISDN  Q921d:isdn_from_driver_process:QUEUE_EVENT
Jan  8 17:24:43.495:ISDN Se2/0:15 Q921f:PBXa RX <- 0x44030300102A34232A35302A33333330
Jan  8 17:24:43.495:   303030312334343430303031
```

All of the octets following “i=” constitute DPNSS L3 information received from the peer:

```
Jan  8 17:24:43.495:ISDN Se2/0:15 Q921:PBXa RX <- UI(C) dlci=1 cntl=UI nbit=0
        i=0x00102A34232A35302A33333330303030312334343430303031
```

In the INFORMATION TRANSFER state, DLC 1 received a UI(C) frame (event x2) from the peer carrying DPNSS L3 information:

```
Jan  8 17:24:43.495:ISDN Se2/0:15 Q921d:process_rxdata:Frame sent to L2
Jan  8 17:24:43.495:ISDN  Q921d:isdn_from_driver_process:event_count 1
Jan  8 17:24:43.495:ISDN Se2/0:15 Q921d:dpnss_l2_main:source_id x200 event
        x141 v_bit x0 chan x0
Jan  8 17:24:43.495:ISDN Se2/0:15 Q921d:s_dpnss_information_transfer:event x2 chan 1
```

For DLC 1, event information is sent to L3 (IUA BACKHAUL, indicated by dest x300). In this case, DL_DATA_IND (event x241) indicates that some L3 information has been received from the peer.

```
Jan  8 17:24:43.495:ISDN Se2/0:15 Q921d:dpnss_l2_mail:dest x300 event x241
        v_bit 1 chan 1 out_pkt x6367175C
```

Information is sent to the driver (dest x200), which is then sent to the peer): An Unnumbered Information--Response [UI(R)] (event x3) acknowledges the received Unnumbered Information--Command [UI(C)].

```
Jan  8 17:24:43.495:ISDN Se2/0:15 Q921d:dpnss_l2_mail:dest x200 event x3
        v_bit 1 chan 1 out_pkt x63F183D4
```

The following is sample output from the **debug isdn q921** command for an outgoing call:

```
Router# debug isdn q921
Jan  3 14:52:24.475: ISDN BR0: TX ->  INFOc sapi = 0  tei = 64  ns = 5  nr = 2
                              i = 0x08010705040288901801837006803631383835
Jan  3 14:52:24.503: ISDN BR0: RX <-  RRr sapi = 0  tei = 64  nr = 6
Jan  3 14:52:24.527: ISDN BR0: RX <-  INFOc sapi = 0  tei = 64  ns = 2  nr = 6
                              i = 0x08018702180189
Jan  3 14:52:24.535: ISDN BR0: TX ->  RRr sapi = 0  tei = 64  nr = 3
Jan  3 14:52:24.643: ISDN BR0: RX <-  INFOc sapi = 0  tei = 64  ns = 3  nr = 6
                              i = 0x08018707
Jan  3 14:52:24.655: ISDN BR0: TX ->  RRr sapi = 0  tei = 64  nr = 4
%LINK-3-UPDOWN: Interface BRI0:1, changed state to up
Jan  3 14:52:24.683: ISDN BR0: TX ->  INFOc sapi = 0  tei = 64  ns = 6  nr = 4
                              i = 0x0801070F
Jan  3 14:52:24.699: ISDN BR0: RX <-  RRr sapi = 0  tei = 64  nr = 7
%LINEPROTO-5-UPDOWN: Line protocol on Interface BRI0:1, changed state to up
%ISDN-6-CONNECT: Interface BRI0:1 is now connected to 61885 goodie
Jan  3 14:52:34.415: ISDN BR0: RX <-  RRp sapi = 0  tei = 64 nr = 7
Jan  3 14:52:34.419: ISDN BR0: TX ->  RRf sapi = 0  tei = 64  nr = 4
```

In the following lines, the seventh and eighth most significant hexadecimal numbers indicate the type of message. 0x05 indicates a Call Setup message, 0x02 indicates a Call Proceeding message, 0x07 indicates a Call Connect message, and 0x0F indicates a Connect Ack message.

```
Jan  3 14:52:24.475: ISDN BR0: TX ->  INFOc sapi = 0  tei = 64  ns = 5  nr = 2
                              i = 0x08010705040288901801837006803631383835
Jan  3 14:52:24.527: ISDN BR0: RX <-  INFOc sapi = 0  tei = 64  ns = 2  nr = 6
                              i = 0x08018702180189
Jan  3 14:52:24.643: ISDN BR0: RX <-  INFOc sapi = 0  tei = 64  ns = 3  nr = 6
                              i = 0x08018707
Jan  3 14:52:24.683: ISDN BR0: TX ->  INFOc sapi = 0  tei = 64  ns = 6  nr = 4
                              i = 0x0801070F
```

The following is sample output from the **debug isdn q921** command for a startup message on a DMS-100 switch:

```
Router# debug isdn q921
Jan  3 14:47:28.455: ISDN BR0: RX <-  IDCKRQ  ri = 0  ai = 127 0
Jan  3 14:47:30.171: ISDN BR0: TX ->  IDREQ  ri = 31815  ai = 127
Jan  3 14:47:30.219: ISDN BR0: RX <-  IDASSN  ri = 31815  ai = 64
Jan  3 14:47:30.223: ISDN BR0: TX ->  SABMEp sapi = 0  tei = 64
Jan  3 14:47:30.227: ISDN BR0: RX <-  IDCKRQ  ri = 0  ai = 127
Jan  3 14:47:30.235: ISDN BR0: TX ->  IDCKRP  ri = 16568  ai = 64
Jan  3 14:47:30.239: ISDN BR0: RX <-  UAf sapi = 0  tei = 64
Jan  3 14:47:30.247: ISDN BR0: TX ->  INFOc sapi = 0  tei = 64  ns = 0  nr = 0
                              i = 0x08007B3A03313233
Jan  3 14:47:30.267: ISDN BR0: RX <-  RRr sapi = 0  tei = 64  nr = 1
Jan  3 14:47:34.243: ISDN BR0: TX ->  INFOc sapi = 0  tei = 64  ns = 1  nr = 0
                              i = 0x08007B3A03313233
Jan  3 14:47:34.267: ISDN BR0: RX <-  RRr sapi = 0  tei = 64  nr = 2
Jan  3 14:47:43.815: ISDN BR0: RX <-  RRp sapi = 0  tei = 64 nr = 2
Jan  3 14:47:43.819: ISDN BR0: TX ->  RRf sapi = 0  tei = 64  nr = 0
Jan  3 14:47:53.819: ISDN BR0: TX ->  RRp sapi = 0  tei = 64 nr = 0
```

The first seven lines of this example indicate a Layer 2 link establishment.

The following lines indicate the message exchanges between the data link layer entity on the local router (user side) and the assignment source point (ASP) on the network side during the TEI assignment procedure. This assumes that the link is down and no TEI currently exists.

```
Jan  3 14:47:30.171: ISDN BR0: TX ->  IDREQ  ri = 31815  ai = 127
Jan  3 14:47:30.219: ISDN BR0: RX <-  IDASSN  ri = 31815  ai = 64
```

At 14:47:30.171, the local router data link layer entity sent an Identity Request message to the network data link layer entity to request a TEI value that can be used in subsequent communication between the peer data link layer entities. The request includes a randomly generated reference number (31815) to differentiate among user devices that request automatic TEI assignment and an action indicator of 127 to indicate that the ASP can assign any TEI value available. The ISDN user interface on the router uses automatic TEI assignment.

At 14:47:30.219, the network data link entity responds to the Identity Request message with an Identity Assigned message. The response includes the reference number (31815) previously sent in the request and TEI value (64) assigned by the ASP.

The following lines indicate the message exchanges between the layer management entity on the network and the layer management entity on the local router (user side) during the TEI check procedure:

```
Jan  3 14:47:30.227: ISDN BR0: RX <-  IDCKRQ  ri = 0  ai = 127
Jan  3 14:47:30.235: ISDN BR0: TX ->  IDCKRP  ri = 16568  ai = 64
```

At 14:47:30.227, the layer management entity on the network sends the Identity Check Request message to the layer management entity on the local router to check whether a TEI is in use. The message includes a reference number that is always 0 and the TEI value to check. In this case, an ai value of 127 indicates that all TEI values should be checked. At 14:47:30.227, the layer management entity on the local router responds with an Identity Check Response message indicating that TEI value 64 is currently in use.

The following lines indicate the messages exchanged between the data link layer entity on the local router (user side) and the data link layer on the network side to place the network side into modulo 128 multiple frame acknowledged operation. Note that the data link layer entity on the network side also can initiate the exchange.

```
Jan  3 14:47:30.223: ISDN BR0: TX ->  SABMEp sapi = 0  tei = 64
Jan  3 14:47:30.239: ISDN BR0: RX <-  UAf sapi = 0  tei = 64
```

At 14:47:30.223, the data link layer entity on the local router sends the SABME command with a SAPI of 0 (call control procedure) for TEI 64. At 14:47:30.239, the first opportunity, the data link layer entity on the network responds with a UA response. This response indicates acceptance of the command. The data link layer entity sending the SABME command may need to send it more than once before receiving a UA response.

The following lines indicate the status of the data link layer entities. Both are ready to receive I frames.

```
Jan  3 14:47:43.815: ISDN BR0: RX <-  RRp sapi = 0  tei = 64 nr = 2
Jan  3 14:47:43.819: ISDN BR0: TX ->  RRf sapi = 0  tei = 64  nr = 0
```

These I-frames are typically exchanged every 10 seconds (T203 timer).

The following is sample output from the **debug isdn q921** command for an incoming call. It is an incoming SETUP message that assumes that the Layer 2 link is already established to the other side.

```
Router# debug isdn q921
Jan  3 14:49:22.507: ISDN BR0: TX ->  RRp sapi = 0  tei = 64 nr = 0
Jan  3 14:49:22.523: ISDN BR0: RX <-  RRf sapi = 0  tei = 64  nr = 2
Jan  3 14:49:32.527: ISDN BR0: TX ->  RRp sapi = 0  tei = 64 nr = 0
Jan  3 14:49:32.543: ISDN BR0: RX <-  RRf sapi = 0  tei = 64  nr = 2
Jan  3 14:49:42.067: ISDN BR0: RX <-  RRp sapi = 0  tei = 64 nr = 2
Jan  3 14:49:42.071: ISDN BR0: TX ->  RRf sapi = 0  tei = 64  nr = 0
Jan  3 14:49:47.307: ISDN BR0: RX <-  UI sapi = 0  tei = 127
                              i = 0x08011F05040288901801897006C13631383836
%LINK-3-UPDOWN: Interface BRI0:1, changed state to up
Jan  3 14:49:47.347: ISDN BR0: TX ->  INFOc sapi = 0  tei = 64  ns = 2  nr = 0
                              i = 0x08019F07180189
Jan  3 14:49:47.367: ISDN BR0: RX <-  RRr sapi = 0  tei = 64  nr = 3
Jan  3 14:49:47.383: ISDN BR0: RX <-  INFOc sapi = 0  tei = 64  ns = 0  nr = 3
                              i = 0x08011F0F180189
Jan  3 14:49:47.391: ISDN BR0: TX ->  RRr sapi = 0  tei = 64  nr = 1
%LINEPROTO-5-UPDOWN: Line protocol on Interface BRI0:1, changed state to up
```

The table below describes the significant fields shown in the display.

Table 88: debug isdn q921 Field Descriptions

Field	Description
Jan 3 14:49:47.391	Indicates the date and time at which the frame was sent from or received by the data link layer entity on the router. The time is maintained by an internal clock.
TX	Indicates that this frame is being sent from the ISDN interface on the local router (user side).
RX	Indicates that this frame is being received by the ISDN interface on the local router from the peer (network side).
IDREQ	Indicates the Identity Request message type sent from the local router to the network (ASP) during the automatic TEI assignment procedure. This message is sent in a UI command frame. The SAPI value for this message type is always 63 (indicating that it is a Layer 2 management procedure) but it is not displayed. The TEI value for this message type is 127 (indicating that it is a broadcast operation).
ri = 31815	Indicates the Reference number used to differentiate between user devices requesting TEI assignment. This value is a randomly generated number from 0 to 65535. The same ri value sent in the IDREQ message should be returned in the corresponding IDASSN message. Note that a Reference number of 0 indicates that the message is sent from the network side management layer entity and a reference number has not been generated.
ai = 127	Indicates the Action indicator used to request that the ASP assign any TEI value. It is always 127 for the broadcast TEI. Note that in some message types, such as IDREM, a specific TEI value is indicated.
IDREM	Indicates the Identity Remove message type sent from the ASP to the user side layer management entity during the TEI removal procedure. This message is sent in a UI command frame. The message includes a reference number that is always 0, because it is not responding to a request from the local router. The ASP sends the Identity Remove message twice to avoid message loss.

Field	Description
IDASSN	Indicates the Identity Assigned message type sent from the ISDN service provider on the network to the local router during the automatic TEI assignment procedure. This message is sent in a UI command frame. The SAPI value for this message type is always 63 (indicating that it is a Layer 2 management procedure). The TEI value for this message type is 127 (indicating it is a broadcast operation).
ai = 64	Indicates the TEI value automatically assigned by the ASP. This TEI value is used by data link layer entities on the local router in subsequent communication with the network. The valid values are in the range from 64 to 126.
SABME	Indicates the set asynchronous balanced mode extended command. This command places the recipient into modulo 128 multiple frame acknowledged operation. This command also indicates that all exception conditions have been cleared. The SABME command is sent once a second for N200 times (typically three times) until its acceptance is confirmed with a UA response. For a list and brief description of other commands and responses that can be exchanged between the data link layer entities on the local router and the network, see ITU-T Recommendation Q.921.
sapi = 0	Identifies the service access point at which the data link layer entity provides services to Layer 3 or to the management layer. A SAPI with the value 0 indicates it is a call control procedure. Note that the Layer 2 management procedures such as TEI assignment, TEI removal, and TEI checking, which are tracked with the **debug isdn q921**command, do not display the corresponding SAPI value; it is implicit. If the SAPI value were displayed, it would be 63.
tei = 64	Indicates the TEI value automatically assigned by the ASP. This TEI value will be used by data link layer entities on the local router in subsequent communication with the network. The valid values are in the range from 64 to 126.

Field	Description
IDCKRQ	Indicates the Identity Check Request message type sent from the ISDN service provider on the network to the local router during the TEI check procedure. This message is sent in a UI command frame. The ri field is always 0. The ai field for this message contains either a specific TEI value for the local router to check or 127, which indicates that the local router should check all TEI values. For a list and brief description of other message types that can be exchanged between the local router and the ISDN service provider on the network, see Appendix B, "ISDN Switch Types, Codes, and Values."
IDCKRP	Indicates the Identity Check Response message type sent from the local router to the ISDN service provider on the network during the TEI check procedure. This message is sent in a UI command frame in response to the IDCKRQ message. The ri field is a randomly generated number from 0 to 65535. The ai field for this message contains the specific TEI value that has been checked.
UAf	Confirms that the network side has accepted the SABME command previously sent by the local router. The final bit is set to 1.
INFOc	Indicates that this is an Information command. It is used to transfer sequentially numbered frames containing information fields that are provided by Layer 3. The information is transferred across a data-link connection.
INFORMATION pd = 8 callref = (null)	Indicates the information fields provided by Layer 3. The information is sent one frame at a time. If multiple frames need to be sent, several Information commands are sent. The pd value is the protocol discriminator. The value 8 indicates it is call control information. The call reference number is always null for SPID information.
SPID information i = 0x34313539303338333636303031	Indicates the SPID. The local router sends this information to the ISDN switch to indicate the services to which it subscribes. SPIDs are assigned by the service provider and are usually 10-digit telephone numbers followed by optional numbers. Currently, only the DMS-100 switch supports SPIDs, one for each B channel. If SPID information is sent to a switch type other than DMS-100, an error may be displayed in the debug information.

Field	Description
ns = 0	Indicates the send sequence number of sent I frames.
nr = 0	Indicates the expected send sequence number of the next received I frame. At time of transmission, this value should be equal to the value of ns. The value of nr is used to determine whether frames need to be re-sent for recovery.
RRr	Indicates the Receive Ready response for unacknowledged information transfer. The RRr is a response to an INFOc.
RRp	Indicates the Receive Ready command with the poll bit set. The data link layer entity on the user side uses the poll bit in the frame to solicit a response from the peer on the network side.
RRf	Indicates the Receive Ready response with the final bit set. The data link layer entity on the network side uses the final bit in the frame to indicate a response to the poll.
sapi	Indicates the service access point identifier. The SAPI is the point at which data link services are provided to a network layer or management entity. Currently, this field can have the value 0 (for call control procedure) or 63 (for Layer 2 management procedures).
tei	Indicates the terminal endpoint identifier (TEI) that has been assigned automatically by the assignment source point (ASP) (also called the layer management entity on the network side). The valid range is from 64 to 126. The value 127 indicates a broadcast.

Related Commands

Command	Description
debug isdn event	Displays ISDN events occurring on the user side (on the router) of the ISDN interface.
debug isdn q931	Displays information about call setup and teardown of ISDN network connections (Layer 3) between the local router (user side) and the network.
service timestamps debug datetime msec	Includes the time with each debug message.

debug isdn q931

To display information about call setup and teardown of ISDN network connections (Layer 3) between the local router (user side) and the network, use the **debug isdn q931** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug isdn q931 [**asn1**| **detail**| **interface** [**bri** *number*]]

no debug isdn q931

Syntax Description

asn1	(Optional) Displays ISDN Q.931 Abstract Syntax Notation number one (ASN.1) details.
detail	(Optional) Displays ISDN Q.931 packet details.
interface	(Optional) Specifies an interface for debugging.
bri *number*	(Optional) Specifies the BRI interface and selects the interface number. Valid values are from 0 to 6.

Command Modes

Privileged EXEC

Command History

Release	Modification
10.0	The **debug isdn** command was introduced.
12.3(11)T	This command was enhanced to display the contents of the Facility Information Element (IE) in textual format.
12.3(14)T	The **asn1**, **detail**, **interface**, and **bri** *number* keywords and argument were added.
12.4(6)T	This command was enhanced to display reports about SAPI 0 procedures that accept X.25 calls on the BRI D channel.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

The ISDN network layer interface provided by the router conforms to the user interface specification defined by ITU-T recommendation Q.931 and to other specifications (for example, switch type VN4). The router only tracks activities that occur on the user side, not on the network side, of the network connection. The **debug isdn q931** command output is limited to commands and responses exchanged during peer-to-peer communication carried over the D channel. This debug information does not include data sent over B channels, which are also part of the router's ISDN interface. The peers (network layers) communicate with each other via an ISDN switch over the D channel.

A router can be the calling or the called party of the ISDN Q.931 network connection call setup and teardown procedures. If the router is the calling party, the command displays information about an outgoing call. If the router is the called party, the command displays information about an incoming call.

This command decodes parameters of the Facility IE and displays them as text, with parameter values as they are applicable and relevant to the operation. In addition, the ASN.1 encoded Notification structure of the Notification-Indicator IE is also decoded.

You can use the **debug isdn q931** command with the **debug isdn event** and the **debug isdn q921** commands at the same time. The displays will be intermingled. Use the **service timestamps debug datetime msec** global configuration command to include the time with each message.

Examples

The following is sample output from the **debug isdn q931** command of a call setup procedure for an outgoing call:

```
Router# debug isdn q931
TX -> SETUP pd = 8 callref = 0x04
 Bearer Capability i = 0x8890
 Channel ID i = 0x83
 Called Party Number i = 0x80, '415555121202'
RX <- CALL_PROC pd = 8 callref = 0x84
 Channel ID i = 0x89
RX <- CONNECT pd = 8 callref = 0x84
TX -> CONNECT_ACK pd = 8 callref = 0x04....
Success rate is 0 percent (0/5)
```

The following is sample output from the **debug isdn q931** command of a call setup procedure for an incoming call:

```
Router# debug isdn q931
RX <- SETUP pd = 8 callref = 0x06
 Bearer Capability i = 0x8890
 Channel ID i = 0x89
 Calling Party Number i = 0x0083, '81012345678902'
TX -> CONNECT pd = 8 callref = 0x86
RX <- CONNECT_ACK pd = 8 callref = 0x06
```

The following is sample output from the **debug isdn q931** command that shows the contents of the Facility IE. The following example uses the supplementary service Malicious Call Identification (MCID). In this service, the router sends out the Facility IE.

```
Router# debug isdn q931
Sep 20 04:09:38.335 UTC: ISDN Se7/1:23 Q931: TX -> DISCONNECT pd = 8  callref = 0x0007
        Cause i = 0x8290 - Normal call clearing
        Facility i = 0x91A106020107020103
                Protocol Profile = Remote Operations Protocol
                0xA106020107020103
                Component = Invoke component
                        Invoke Id = 7 <MCID>
                        Operation = MCIDRequest
```

The following is sample output from the **debug isdn q931** command of a call teardown procedure from the network:

```
Router# debug isdn q931
RX <- DISCONNECT pd = 8 callref = 0x84
 Cause i = 0x8790
 Looking Shift to Codeset 6
 Codeset 6 IE 0x1 1 0x82 '10'
TX -> RELEASE pd = 8 callref = 0x04
```

```
 Cause i = 0x8090
RX <- RELEASE_COMP pd = 8 callref = 0x84
```

The following example shows how to turn on the **debug isdn q931 asn1** capability and how to use the **show debug** command to display the results of the debug:

```
Router# debug isdn q931 asn1
debug isdn asn1 is ON.
Router# show debug

The following ISDN debugs are enabled on all DSLs:
debug isdn error is ON.
debug isdn event is ON.
debug isdn q931 is ON.
debug isdn asn1 is ON.
DEBUGS with ASN1 enabled:
ice call = 0x1
00:08:49: Sub Msg = CDAPI_MSG_SUBTYPE_TBCT_REQ
00:08:49: Call Type = VOICE
00:08:49: B Channel = 0
00:08:49: Cause = 0
00:08:49: ISDN ASN1: isdnAsn1Component
00:08:49: ISDN ASN1: isdnAsn1Invoke
00:08:49: ISDN ASN1: isdnAsn1InvTBCT
00:08:49: ISDN ASN1: op Invoke TBCT
00:08:49: ISDN Se0:23 Q931: TX -> FACILITY pd = 8 callref = 0x8001
Facility i = 0x91A11102010506072A8648CE1500083003020101
*Jun 15 06:27:51.547: %ISDN-6-CONNECT: Interface Serial0:0 is now connected to 1 11111
00:08:51: ISDN Se0:23 Q931: RX <- FACILITY pd = 8 callref = 0x01
Facility i = 0x91A203020105A11302010180010506072A8648CE15000A81020164
00:08:51: ISDN ASN1: isdnAsn1Component
00:08:51: ISDN ASN1: isdnAsn1Res
00:08:51: ISDN ASN1: isdnAsn1ResTbct
```

The table below describes the significant fields shown in the displays, in alphabetical order.

Table 89: debug isdn q931 Field Descriptions

Field	Description
Bearer Capability	Indicates the requested bearer service to be provided by the network.
CALL_PROC	Indicates the CALL PROCEEDING message; the requested call setup has begun, and no more call setup information will be accepted.
Called Party Number	Identifies the called party. This field is present only in outgoing SETUP messages. Note that this field can be replaced by the Keypad facility field. This field uses the IA5 character set.
Calling Party Number	Identifies the origin of the call. This field is present only in incoming SETUP messages. This field uses the IA5 character set.

Field	Description
callref	Indicates the call reference number in hexadecimal notation. The value of this field indicates the number of calls made from either the router (outgoing calls) or the network (incoming calls). Note that the originator of the SETUP message sets the high-order bit of the call reference number to 0. The destination of the connection sets the high-order bit to 1 in subsequent call control messages, such as the CONNECT message. For example, callref = 0x04 in the request becomes callref = 0x84 in the response.
Cause	Indicates the cause of the disconnect.
Channel ID	Indicates the channel identifier. The value 83 indicates any channel, 89 indicates the B1 channel, and 8A indicates the B2 channel. For more information about the channel identifier, see ITU-T Recommendation Q.931.
Codeset 6 IE 0x1 i = 0x82, ‘10’	Indicates charging information. This information is specific to the NTT switch type and may not be sent by other switch types.
CONNECT	Indicates that the called user has accepted the call.
CONNECT_ACK	Indicates that the calling user acknowledges the called user’s acceptance of the call.
DISCONNECT	Indicates either that the user side has requested the network to clear an end-to-end connection or that the network has cleared the end-to-end connection.
i =	Indicates the information element identifier. The value depends on the field with which the identifier is associated. See the ITU-T Q.931 specification for details about the possible values associated with each field for which this identifier is relevant.
Looking Shift to Codeset 6	Indicates that the next information elements will be interpreted according to information element identifiers assigned in codeset 6. Codeset 6 means that the information elements are specific to the local network.

Field	Description
pd	Indicates the protocol discriminator that distinguishes messages for call control over the user-network ISDN interface from other ITU-T-defined messages, including other Q.931 messages. The protocol discriminator is 8 for call control messages, such as SETUP. For basic-1tr6, the protocol discriminator is 65.
Protocol Profile	Remote operations protocol, which contains networking extensions for other services. This profile determines which protocol should be used to decode the rest of a Facility IE message. A Facility IE can contain multiple components. Each component displays a hexadecimal code followed by the code contents in text. In the example that included encoded ISDN Facility IE message output, 0xA106020107020103 is the hexadecimal code and represents the Facility IE Component, Invoke Id, and Operation. The Operation portion of the IE corresponds to the supplementary service that the component represents.
RELEASE	Indicates that the sending equipment will release the channel and call reference. The recipient of this message should prepare to release the call reference and channel.
RELEASE_COMP	Indicates that the sending equipment has received a RELEASE message and has now released the call reference and channel.
RX <-	Indicates that this message is being received by the user side of the ISDN interface from the network side.
SETUP	Indicates that the SETUP message type has been sent to initiate call establishment between peer network layers. This message can be sent from either the local router or the network.
TX ->	Indicates that this message is being sent from the local router (user side) to the network side of the ISDN interface.

Text in bold in the following example indicates the acceptance of an incoming X.25 call on the ISDN D channel, per ITU Q.931 SAPI value 0 procedures:

```
Router# debug isdn q931
```

```
*Sep 28 12:34:29.739: ISDN BR1/1 Q931: RX <- SETUP pd = 8  callref = 0x5C (re-assembled)
        Bearer Capability i = 0x88C0C2E6
                Standard = CCITT
                Transfer Capability = Unrestricted Digital
                Transfer Mode = Packet
                Transfer Rate = Packet - not specified
                User Info L2 Protocol = Recommendation Q921/I.441
                User Info L3 Protocol = Recommendation X.25, Packet Layer

        Channel ID i = 0x8C
                Exclusive, No B-channel

        Information Rate i = 0x8888
        Packet Layer Binary Params i = 0x80
        Packet Layer Window Size i = 0x8282
        Packet Size i = 0x8888
        Calling Party Number i = 0x0083, '144014384106'
                Plan:Unknown, Type:Unknown
        User-User i = 0x02CC000000
```

The command output is intermingled with information from the **debug isdn events** command; see the description for the **debug isdn events** command to understand significant fields displayed in this report.

Related Commands

Command	Description
debug isdn events	Displays ISDN events occurring on the router (user side) of the ISDN interface.
debug isdn q921	Displays Layer 2 access procedures that are taking place at the router on the D channel of the ISDN interface.
service timestamps	Configure a time stamp on debugging or system logging messages.

debug isdn tgrm

To view ISDN trunk group resource manager information, use the **debug isdn tgrm** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug isdn tgrm

no debug isdn tgrm

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(11)T	This command was introduced.

Usage Guidelines

Disable console logging and use buffered logging before using the **debug isdn tgrm** command. Using the **debug isdn tgrm** command generates a large volume of debugs, which can affect router performance.

Examples

Sample output from the **debug isdn tgrm** command is shown below.

The output shows that the channel used (bchan) is 1, service state is 0 (in-service), call_state is 2 (busy), “false busy” is 0, and DSL is 2. The output also shows that the B channel is 1, the channel is available, and the call state is transitioned from 0 (idle) to 2 (busy).

The last two lines of output shows that bchan is 1, call state is 1 (busy), call type is 2 (voice), and call direction is 1 (incoming).

```
00:26:31:ISDN:get_tgrm_avail_state:idb 0x64229380 bchan 1 service_state 0 call_state 2 false
 busy 0x0 dsl 2
00:26:31:ISDN:update_tgrm_call_status:idb 0x64229380 bchan 1 availability state 1 call
state(prev,new) (0,2), dsl 2
00:26:31:ISDN:Calling TGRM with tgrm_call_isdn_update:idb 0x64229380 bchan 1 call state 1
call type 2 call dir 1
```

The table below provides an alphabetical listing of the fields shown in the **debug isdn tgrm** command output and a description of each field.

Table 90: debug isdn tgrm Field Descriptions

Field	Description
availability state	Indicates whether the channel is available: 0 = Not available 1 = Available
bchan	Bearer channel used for this call.
call dir	Direction of the call: 0 = Incoming 1 = Outgoing
call_state	State of the call. It has different values depending on whether it is from ISDN perspective or TGRM perspective. When printed from get_tgrm_avail_state(), it is the state value from ISDN perspective: 0 = Idle 1 = Negotiate 2 = Busy 3 = Reserved 4 = Restart pending 5 = Maintenance pend 6 = Reassigned When printed from tgrm_call_isdn_update(), it is the state value from TGRM perspective: 0 = Idle 1 = Busy 2 = Pending 3 = Reject
call state (prev, new)	Indicates the state transition of the call. The state values are as shown in call_state from the ISDN perspective.
call type	Type of call: 0 = Invalid 1 = Data 2 = Voice 3 = Modem 4 = None
dsl	Internal interface identifier.
false busy	Bit map of all the channels on the interface indicating their soft busy status.
idb	Address of the interface descriptor block (IDB) for the interface.
service_state	Service state: 0 = In-service 1 = Maintenance 2 = Out of service

Related Commands

Command	Description
show trunk group	Displays the configuration of the trunk group.

Command	Description
translation-profile (voice service POTS)	Assigns a translation profile to the interface.
trunk-group (interface)	Assigns a trunk group to the interface.

debug isis adj packets

To display information on all a djacency-related activity such as hello packets sent and received and Intermediate System-to-Intermediate System (IS-IS) adjacencies going up and down, use the **debug isis adj packets** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug isis adj packets [*interface*]

no debug isis adj packets [*interface*]

Syntax Description

interface	(Optional) Interface or subinterface name.

Command Modes

Privileged EXEC

Examples

The following is sample output from the **debug isis adj packets** command:

```
Router# debug isis adj packets
ISIS-Adj: Rec L1 IIH from 0000.0c00.40af (Ethernet0), cir type 3, cir id BBBB.BBBB.BBBB.01
ISIS-Adj: Rec L2 IIH from 0000.0c00.40af (Ethernet0), cir type 3, cir id BBBB.BBBB.BBBB.01
ISIS-Adj: Rec L1 IIH from 0000.0c00.0c36 (Ethernet1), cir type 3, cir id CCCC.CCCC.CCCC.03
ISIS-Adj: Area mismatch, level 1 IIH on Ethernet1
ISIS-Adj: Sending L1 IIH on Ethernet1
ISIS-Adj: Sending L2 IIH on Ethernet1
ISIS-Adj: Rec L2 IIH from 0000.0c00.0c36 (Ethernet1), cir type 3, cir id BBBB.BBBB.BBBB.03
```

The following line indicates that the router received an IS-IS hello packet (IIH) on Ethernet interface 0 from the Level 1 router (L1) at MAC address 0000.0c00.40af. The circuit type is the interface type:

```
1--Level 1 only; 2--Level 2 only; 3--Level 1/2
```

The circuit ID is what the neighbor interprets as the designated router for the interface.

```
ISIS-Adj: Rec L1 IIH from 0000.0c00.40af (Ethernet0), cir type 3, cir id BBBB.BBBB.BBBB.01
```

The following line indicates that the router (configured as a Level 1 router) received on Ethernet interface 1 is an IS-IS hello packet from a Level 1 router in another area, thereby declaring an area mismatch:

```
ISIS-Adj: Area mismatch, level 1 IIH on Ethernet1
```

The following lines indicates that the router (configured as a Level 1/Level 2 router) sent on Ethernet interface 1 is a Level 1 IS-IS hello packet, and then a Level 2 IS-IS packet:

```
ISIS-Adj: Sending L1 IIH on Ethernet1
ISIS-Adj: Sending L2 IIH on Ethernet1
```

debug isis authentication

To enable debugging of Intermediate System-to-Intermediate System (IS-IS) authentication, use the **debug isis authentication**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug isis authentication information

no debug isis authentication information

Syntax Description

information	Required keyword that specifies IS-IS authentication information.

Command Default

No default behavior or values.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(21)ST	This command was introduced.
12.2(13)T	This command was integrated into Cisco IOS Release 12.2(13)T.
12.2(14)S	This command was integrated into Cisco IOS Release 12.2(14)S.

Examples

The following is sample output from the **debug isis authentication**command with the **information** keyword:

```
Router# debug isis authentication information
3d03h:ISIS-AuthInfo:No auth TLV found in received packet
3d03h:ISIS-AuthInfo:No auth TLV found in received packet
```

The sample output indicates that the router has been running for 3 days and 3 hours. Debugging output is about IS-IS authentication information. The local router is configured for authentication, but it received a packet that does not contain authentication data; the remote router does not have authentication configured.

debug isis ipv6 rib

To display debugging information for Integrated Intermediate System-to-Intermediate System (IS-IS) IPv6 Version 6 routes in the global or local Routing Information Base (RIB), use the **debug isis rib**command in privileged EXEC mode. To disable the debugging of IS-IS IPv6 routes, use the **no** form of this command.

debug isis ipv6 rib [**global**| **local** [**access-list-number**| **terse**]]

no debug isis ipv6 rib [**global**| **local**]

Syntax Description

global	(Optional) Displays debugging information for IS-IS IP Version 4 routes in the global RIB.
local	(Optional) Displays debugging information for IS-IS IP Version 4 routes in the IS-IS local RIB.
access-list-number	(Optional) Number of an access list. This is a decimal number from 100 to 199 or from 2000 to 2699.
terse	(Optional) Will not display debug information if the IS-IS IP Version 4 IS-IS local RIB has not changed.

Command Default

Debugging of IS-IS IPv6 routes is disabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
Cisco IOS XE Release 3.6S	This command was introduced.

Usage Guidelines

Examples

The following is sample output from the **debug isis ipv6 rib**command shows shows an IPv6 prefix tag.

The table below describes the significant fields shown in the display.

Table 91: debug isis ipv6 rib Field Descriptions

Field	Description

Related Commands

Command	Description
isis ipv6 tag	Configures an administrative tag value to be associated with an IPv6 address prefix.

debug isis mpls traffic-eng advertisements

To print information about traffic engineering advertisements in Intermediate System-to-Intermediate System (IS-IS) link-state advertisement (LSA) messages, use the **debug isis mpls traffic-eng advertisements** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug isis mpls traffic-eng advertisements

no debug isis mpls traffic-eng advertisements

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(5)ST	This command was introduced.
12.1(3)T	This command was integrated into Cisco IOS Release 12.1(3)T.
12.0(22)S	This command was integrated into Cisco IOS Release 12.0(22)S.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Examples

In the following example, information about traffic engineering advertisements is printed in IS-IS LSA messages:

```
Router# debug isis mpls traffic-eng advertisements
System ID:Router1.00
  Router ID:10.106.0.6
  Link Count:1
    Link[1]
      Neighbor System ID:Router2.00 (P2P link)
      Interface IP address:10.42.0.6
      Neighbor IP Address:10.42.0.10
      Admin. Weight:10
      Physical BW:155520000 bits/sec
      Reservable BW:5000000 bits/sec
      BW unreserved[0]:2000000 bits/sec, BW unreserved[1]:100000 bits/sec
      BW unreserved[2]:100000 bits/sec, BW unreserved[3]:100000 bits/sec
      BW unreserved[4]:100000 bits/sec, BW unreserved[5]:100000 bits/sec
      BW unreserved[6]:100000 bits/sec, BW unreserved[7]:0 bits/sec
      Affinity Bits:0x00000000
```

The table below describes the significant fields shown in the display.

Table 92: debug isis mpls traffic-eng advertisements Field Descriptions

Field	Description
System ID	Identification value for the local system in the area.
Router ID	Multiprotocol Label Switching traffic engineering router ID.
Link Count	Number of links that MPLS traffic engineering advertised.
Neighbor System ID	Identification value for the remote system in an area.
Interface IP address	IPv4 address of the interface.
Neighbor IP Address	IPv4 address of the neighbor.
Admin. Weight	Administrative weight associated with this link.
Physical BW	Bandwidth capacity of the link (in bits per second).
Reservable BW	Amount of reservable bandwidth on this link.
BW unreserved	Amount of bandwidth that is available for reservation.
Affinity Bits	Attribute flags of the link that are being flooded.

debug isis mpls traffic-eng events

To print information about traffic engineering-related Intermediate System-to-Intermediate System (IS-IS) events, use the **debug isis mpls traffic-eng events** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug isis mpls traffic-eng events

no debug isis mpls traffic-eng events

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(5)ST	This command was introduced.
12.1(3)T	This command was integrated into Cisco IOS Release 12.1(3)T.
12.0(22)S	This command was integrated into Cisco IOS Release 12.0(22)S.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Examples

In the following example, information is printed about traffic engineering-related IS-IS events:

```
Router# debug isis mpls traffic-eng events
ISIS-RRR:Send MPLS TE Et4/0/1 Router1.02 adjacency down:address 0.0.0.0
ISIS-RRR:Found interface address 10.1.0.6 Router1.02, building subtlv... 58 bytes
ISIS-RRR:Found interface address 10.42.0.6 Router2.00, building subtlv... 64 bytes
ISIS-RRR:Interface address 0.0.0.0 Router1.00 not found, not building subtlv
ISIS-RRR:LSP Router1.02 changed from 0x606BCD30
ISIS-RRR:Mark LSP Router1.02 changed because TLV contents different, code 16
ISIS-RRR:Received 1 MPLS TE links flood info for system id Router1.00
```

debug isis nsf

To display information about the Intermediate System-to-Intermediate System (IS-IS) state during a Cisco nonstop forwarding (NSF) restart, use the **d ebug isis nsf** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug isis nsf [detail]

no debug isis nsf [detail]

Syntax Description

detail	(Optional) Provides detailed debugging information.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(22)S	This command was introduced.
12.2(18)S	This command was integrated into Cisco IOS Release 12.2(18)S.
12.2(20)S	Support for the Cisco 7304 router was added.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.

Usage Guidelines

Use the **debug isis nsf**command to display basic information about the IS-IS state during an NSF restart. Use the **debug isis nsf detail**command to display additional IS-IS state detail during an NSF restart.

Examples

The following example displays IS-IS state information during an NSF restart:

```
router# debug isis nsf
IS-IS NSF events debugging is on
```

The following example displays detailed IS-IS state information during an NSF restart:

```
router# debug isis nsf detail
IS-IS NSF events (detailed) debugging is on
router#
Jan 24 20:04:54.090:%CLNS-5-ADJCHANGE:ISIS:Adjacency to gsr1 (GigabitEthernet2/0/0) Up,
Standby adjacency
Jan 24 20:04:54.090:ISIS-NSF:ADJ:000C.0000.0000 (Gi2/0/0), type 8/1, cnt 0/1, ht 10 (NEW)
Jan 24 20:04:54.142:ISIS-NSF:Rcv LSP - L2 000B.0000.0000.00-00, seq 251, csum B0DC, ht 120,
 len 123 (local)
```

```
Jan 24 20:04:55.510:ISIS-NSF:Rcv LSP - L1 000B.0000.0000.00-00, seq 23E, csum D20D, ht 120,
 len 100 (local)
Jan 24 20:04:56.494:ISIS-NSF:ADJ:000C.0000.0000 (Gi2/0/0), type 8/0, cnt 0/1, ht 30
Jan 24 20:04:56.502:ISIS-NSF:Rcv LSP - L1 000B.0000.0000.01-00, seq 21C, csum 413, ht 120,
 len 58 (local)
Jan 24 20:04:58.230:ISIS-NSF:Rcv LSP - L2 000C.0000.0000.00-00, seq 11A, csum E197, ht 1194,
 len 88 (Gi2/0/0)
Jan 24 20:05:00.554:ISIS-NSF:Rcv LSP - L1 000B.0000.0000.00-00, seq 23F, csum 1527, ht 120,
 len 111 (local)
```

Related Commands

Command	Description
nsf (IS-IS)	Configures NSF operations for IS-IS.
nsf interface wait	Specifies how long an NSF restart will wait for all interfaces with IS-IS adjacencies to come up before completing the restart.
nsf interval	Specifies the minimum time between NSF restart attempts.
nsf t3	Specifies the methodology used to determine how long IETF NSF will wait for the LSP database to synchronize before generating overloaded link state information for itself and flooding that information out to its neighbors.
show clns neighbors	Displays both ES and IS neighbors.
show isis nsf	Displays current state information regarding IS-IS NSF.

debug isis rib

To display debugging information for Integrated Intermediate System-to-Intermediate System (IS-IS) IP Version 4 routes in the global or local Routing Information Base (RIB), use the **debug isis rib**command in privileged EXEC mode. To disable the debugging of IS-IS IP Version 4 routes, use the **no** form of this command.

debug isis rib [**global**| **local** [**access-list-number**| **terse**]]

no debug isis rib [**global**| **local**]

Syntax Description

global	(Optional) Displays debugging information for IS-IS IP Version 4 routes in the global RIB.
local	(Optional) Displays debugging information for IS-IS IP Version 4 routes in the IS-IS local RIB.
access-list-number	(Optional) Number of an access list. This is a decimal number from 100 to 199 or from 2000 to 2699.
terse	(Optional) Will not display debug information if the IS-IS IP Version 4 IS-IS local RIB has not changed.

Command Default

Debugging of IS-IS IP Version 4 routes is disabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(26)S	This command was introduced.
12.3(4)T	This command was integrated into Cisco IOS Release 12.3(4)T.
12.2(25)S	This command was integrated into Cisco IOS Release 12.2(25)S.
12.2(18)SXE	This command was integrated into Cisco IOS Release 12.2(18)SXE.
12.2(27)SBC	This command was integrated into Cisco IOS Release 12.2(27)SBC.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

Use the **debug isis rib** command to verify if an IP prefix has been installed or removed. To monitor updates from the IS-IS database to the IS-IS local RIB, use the **local** keyword, and to monitor updates from the IS-IS database to the global RIB, use the **global** keyword.

It is highly recommended that you limit the debugging output to information specific to the IP prefix that is associated with a specific access list by entering the *accest-list-number* argument.

Examples

The following is sample output from the **debug isis rib**command after the **ip route priority high**command was used to give high priority to IS-IS IP prefixes for the configured access list access-list1. The debug output shows that the route 10.1.1.0/24 has been removed from the IS-IS local RIB.

```
Router# show running-config| include access-list 1
accest-list 1 permit 10.1.1.0 0.0.0.255
! access-list 1 is configured
Router# debug isis rib local terse 1
00:07:07: ISIS-LR: 10.1.1.0/24 aged out in LSP[10/(7->8)]
! The route 10.1.1.0/24 is removed from the IS-IS local RIB LSP[10/(7->8)].
00:07:07: ISIS-LR: rem path: [115/80/20] via 10.2.2.2(Et2) from 10.22.22.22 tg 0 LSP[10/7]
 from active chain (add to deleted chain)
!The remote path [115/80/20] is removed from the active chain.
00:07:07: ISIS-LR: Enqueued to updateQ[2] for 10.1.1.0/24
!Q[2] is marked to be the update
00:07:07: ISIS-LR: rem path: [115/80/20] via 10.2.2.2(Et2) from 10.22.22.22 tg 0 LSP[10/7]
 from deleted chain
00:07:07: ISIS-LR: Rem RT 10.1.1.0/24
!The remote route [115/80/20] is removed from the deleted chain
```

The table below describes the significant fields shown in the display.

Table 93: debug isis rib Field Descriptions

Field	Description
ISIS-LR	IS-IS local route debugger.
10.1.1.0/24	IP prefix.
rem path:	Indicates the removal or insertion of a routing path--in this instance, it is a removal.
[115/80/20]	Administrative instance/type/metric for the routing path that has been removed or inserted.
via 10.2.2.2(Et2)	IP address of the next hop of the router, in this instance, Ethernet2.
from 10.22.22.22	IP address to advertise the route path.
tg 0	Priority of the IP prefix. All prefixes have a tag 0 priority unless otherwise configured.

Related Commands

Command	Description
ip route priority high	Assigns a high priority to an IS-IS IP prefix.
show isis rib	Displays paths for routes in the IP Version 4 IS-IS local RIB.

debug isis rib redistribution

To debug the events that update the Intermediate System-to-Intermediate System (IS-IS) redistribution cache, use the **debug isis rib redistribution**command in privileged EXEC mode. To turn off debugging, use the **no** form of this command.

debug isis rib redistribution [**level-1**| **level-2**] [*access-list*]

no debug isis rib redistribution [**level-1**| **level-2**] [*access-list*]

Syntax Description

level-1	(Optional) Displays debug information for level 1 redistribution cache.
level-2	(Optional) Displays debug information for level 2 redistribution cache.
access-list	(Optional) An access list number from 1 to 199 or from 1300 to 2699.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(27)S	This command was introduced.
12.3(7)T	This command was integrated into Cisco IOS Release 12.3(7)T.
12.2(25)S	This command was integrated into Cisco IOS Release 12.2(25)S.
12.2(18)SXE	This command was integrated into Cisco IOS Release 12.2(18)SXE.
12.2(27)SBC	This command was integrated into Cisco IOS Release 12.2(27)SBC.

Usage Guidelines

We recommend that you use this command only when a Cisco Technical Assistance Center representative requests you to do so to gather information for a troubleshooting purpose.

Examples

In the following example, the **debug isis rib redistribution** command is used to display information about events that update the IS-IS redistribution cache. The output is self-explanatory.

```
Router# debug isis rib redistribution level-1 123
IS-IS IPv4 redistribution RIB debugging is on for access list 123 for L1
Router# router isis
Router(config-router)# redistribute connected level-1
```

```
Router(config)# access-list 123 permit ip 10.0.0.0 0.255.255.255 any
Router(config)# interface Loopback123
Router(config-if)# ip address 10.123.123.3 255.255.255.255
Nov 25 00:33:46.532: ISIS-RR: 10.123.123.3/32: Up event, from 0x607CAF60
Nov 25 00:33:46.532: ISIS-RR:   looking at L1 redist RIB
Nov 25 00:33:46.532: ISIS-RR:     redistributed to ISIS
Nov 25 00:33:46.532: ISIS-RR:   added    10.123.123.3/32 to L1 redist RIB: [Connected/0]
tag 0 external
Nov 25 00:33:47.532: ISIS-RR: Scanning L1 redist RIB
Nov 25 00:33:47.532: ISIS-RR:   adv 10.123.123.3/32 as L1 redist route
Nov 25 00:33:47.532: ISIS-RR: End of scanningL1 redist RIB
```

The following line indicates that the connected route 10.123.123.3/32 was added to the IS-IS level 1 local redistribution cache with cost 0, metric type external, and administrative tag of 0:

```
Nov 25 00:33:46.532: ISIS-RR:   added    10.123.123.3/32 to L1 redist RIB: [Connected/0]
tag 0 external
```

The following line indicates that the redistributed route 10.123.123.3/32 was advertised in an IS-IS link-state packet (LSP) as a level 1 redistributed route:

```
Nov 25 00:33:47.532: ISIS-RR:   adv 10.123.123.3/32 as L1 redist rout
```

Related Commands

Command	Description
clear isis rib redistribution	Clears some or all prefixes in the local redistribution cache.
show isis rib redistribution	Displays the prefixes in the IS-IS redistribution cache.

debug isis spf statistics

To display statistical information about building routes between intermediate systems (ISs), use the **debug isis spf statistics** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug isis spf statistics

no debug isis spf statistics

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

The Intermediate System-to-Intermediate System (IS-IS) Interdomain Routing Protocol (IDRP) provides routing between ISs by flooding the network with link-state information. IS-IS provides routing at two levels, intra-area (Level 1) and intra-domain (Level 2). Level 1 routing allows Level 1 ISs to communicate with other Level 1 ISs in the same area. Level 2 routing allows Level 2 ISs to build an interdomain backbone between Level 1 areas by traversing only Level 2 ISs. Level 1 ISs only need to know the path to the nearest Level 2 IS in order to take advantage of the interdomain backbone created by the Level 2 ISs.

The IS-IS protocol uses the shortest-path first (SPF) routing algorithm to build Level 1 and Level 2 routes. The **debug isis spf statistics** command provides information for determining the time required to place a Level 1 IS or Level 2 IS on the shortest path tree (SPT) using the IS-IS protocol.

Note The SPF algorithm is also called the Dijkstra algorithm, after the creator of the algorithm.

Examples

The following is sample output from the **debug isis spf statistics** command:

```
Router# debug isis spf statistics
ISIS-Stats: Compute L1 SPT, Timestamp 2780.328 seconds
ISIS-Stats: Complete L1 SPT, Compute time 0.004, 1 nodes on SPT
ISIS-Stats: Compute L2 SPT, Timestamp 2780.3336 seconds
ISIS-Stats: Complete L2 SPT, Compute time 0.056, 12 nodes on SPT
```

The table below describes the significant fields shown in the display.

Table 94: debug isis spf statistics Field Descriptions

Field	Description
Compute L1 SPT	Indicates that Level 1 ISs are to be added to a Level 1 area.

Field	Description
Timestamp	Indicates the time at which the SPF algorithm was applied. The time is expressed as the number of seconds elapsed since the system was up and configured.
Complete L1 SPT	Indicates that the algorithm has completed for Level 1 routing.
Compute time	Indicates the time required to place the ISs on the SPT.
nodes on SPT	Indicates the number of ISs that have been added.
Compute L2 SPT	Indicates that Level 2 ISs are to be added to the domain.
Complete L2 SPT	Indicates that the algorithm has completed for Level 2 routing.

The following lines show the statistical information available for Level 1 ISs:

```
ISIS-Stats: Compute L1 SPT, Timestamp 2780.328 seconds
ISIS-Stats: Complete L1 SPT, Compute time 0.004, 1 nodes on SPT
```

The output indicates that the SPF algorithm was applied 2780.328 seconds after the system was up and configured. Given the existing intra-area topology, 4 milliseconds were required to place one Level 1 IS on the SPT.

The following lines show the statistical information available for Level 2 ISs:

```
ISIS-Stats: Compute L2 SPT, Timestamp 2780.3336 seconds
ISIS-Stats: Complete L2 SPT, Compute time 0.056, 12 nodes on SPT
```

This output indicates that the SPF algorithm was applied 2780.3336 seconds after the system was up and configured. Given the existing intradomain topology, 56 milliseconds were required to place 12 Level 2 ISs on the SPT.

debug isis spf-events

To display a log of significant events during an Intermediate System-to-Intermediate System (IS-IS) shortest-path first (SPF) computation, use the **debug isis spf-events**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug isis spf-events

no debug isis spf-events

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0	This command was introduced.
12.2(15)T	Support for IPv6 was added.
12.2(18)S	Support for IPv6 was added.
12.0(26)S	Support for IPv6 was added.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.
Cisco IOS XE Release 2.6	This command was introduced on Cisco ASR 1000 series routers.

Usage Guidelines

This command displays information about significant events that occur during SPF-related processing.

Examples

The following example displays significant events during an IS-IS SPF computation:

```
Router# debug isis spf-events
ISIS-Spf:  Compute L2 IPv6 SPT
ISIS-Spf: Move 0000.0000.1111.00-00 to PATHS, metric 0
ISIS-Spf: Add 0000.0000.2222.01-00 to TENT, metric 10
ISIS-Spf: Move 0000.0000.2222.01-00 to PATHS, metric 10
ISIS-Spf: considering adj to 0000.0000.2222 (Ethernet3/1) metric 10, level 2, circuit 3,
adj 3
ISIS-Spf:   (accepted)
ISIS-Spf: Add 0000.0000.2222.00-00 to TENT, metric 10
ISIS-Spf:   Next hop 0000.0000.2222 (Ethernet3/1)
ISIS-Spf: Move 0000.0000.2222.00-00 to PATHS, metric 10
ISIS-Spf: Add 0000.0000.2222.02-00 to TENT, metric 20
```

```
ISIS-Spf:   Next hop 0000.0000.2222 (Ethernet3/1)
ISIS-Spf: Move 0000.0000.2222.02-00 to PATHS, metric 20
ISIS-Spf: Add 0000.0000.3333.00-00 to TENT, metric 20
ISIS-Spf:   Next hop 0000.0000.2222 (Ethernet3/1)
ISIS-Spf: Move 0000.0000.3333.00-00 to PATHS, metric 20
```

debug isis update-packets

To display various sequence number protocol data units (PDUs) and link-state packets that are detected by a router, use the **debug isis update-packets** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug isis update-packets

no debug isis update-packets

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Examples

This router has been configured for IS-IS routing. The following is sample output from thee **debug isis update-packets** command:

```
Router# debug isis update-packets
ISIS-Update: Sending L1 CSNP on Ethernet0
ISIS-Update: Sending L2 CSNP on Ethernet0
ISIS-Update: Updating L2 LSP
ISIS-Update: Delete link 888.8800.0181.00 from L2 LSP 1600.8906.4022.00-00, seq E
ISIS-Update: Updating L1 LSP
ISIS-Update: Sending L1 CSNP on Ethernet0
ISIS-Update: Sending L2 CSNP on Ethernet0
ISIS-Update: Add link 8888.8800.0181.00 to L2 LSP 1600.8906.4022.00-00, new seq 10,
 len 91
ISIS-Update: Sending L2 LSP 1600.8906.4022.00-00, seq 10, ht 1198 on Tunnel0
ISIS-Update: Sending L2 CSNP on Tunnel0
ISIS-Update: Updating L2 LSP
ISIS-Update: Rate limiting L2 LSP 1600.8906.4022.00-00, seq 11 (Tunnel0)
ISIS-Update: Updating L1 LSP
ISIS-Update: Rec L2 LSP 888.8800.0181.00.00-00 (Tunnel0)
ISIS-Update: PSNP entry 1600.8906.4022.00-00, seq 10, ht 1196
```

The following lines indicate that the router has sent a periodic Level 1 and Level 2 complete sequence number PDU on Ethernet interface 0:

```
ISIS-Update: Sending L1 CSNP on Ethernet0
ISIS-Update: Sending L2 CSNP on Ethernet0
```

The following lines indicate that the network service access point (NSAP) identified as 8888.8800.0181.00 was deleted from the Level 2 LSP 1600.8906.4022.00-00. The sequence number associated with this LSP is 0xE.

```
ISIS-Update: Updating L2 LSP
ISIS-Update: Delete link 888.8800.0181.00 from L2 LSP 1600.8906.4022.00-00, seq E
```

The following lines indicate that the NSAP identified as 8888.8800.0181.00 was added to the Level 2 LSP 1600.8906.4022.00-00. The new sequence number associated with this LSP is 0x10.

```
ISIS-Update: Updating L1 LSP
ISIS-Update: Sending L1 CSNP on Ethernet0
ISIS-Update: Sending L2 CSNP on Ethernet0
ISIS-Update: Add link 8888.8800.0181.00 to L2 LSP 1600.8906.4022.00-00, new seq 10,
 len 91
```

The following line indicates that the router sent Level 2 LSP 1600.8906.4022.00-00 with sequence number 0x10 on tunnel 0 interface:

```
ISIS-Update: Sending L2 LSP 1600.8906.4022.00-00, seq 10, ht 1198 on Tunnel0
```

The following lines indicates that a Level 2 LSP could not be transmitted because it was recently sent:

```
ISIS-Update: Sending L2 CSNP on Tunnel0
ISIS-Update: Updating L2 LSP
ISIS-Update: Rate limiting L2 LSP 1600.8906.4022.00-00, seq 11 (Tunnel0)
```

The following lines indicate that a Level 2 partial sequence number PDU (PSNP) has been received on tunnel 0 interface:

```
ISIS-Update: Updating L1 LSP
ISIS-Update: Rec L2 PSNP from 8888.8800.0181.00 (Tunnel0)
```

The following line indicates that a Level 2 PSNP with an entry for Level 2 LSP 1600.8906.4022.00-00 has been received. This output is an acknowledgment that a previously sent LSP was received without an error.

```
ISIS-Update: PSNP entry 1600.8906.4022.00-00, seq 10, ht 1196
```

debug iua as

To display debugging messages for the IDSN User Adaptation Layer (IUA) application server (AS), use the **debug iua as** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug iua as {**user**| **state**} {**all**| **name** *as-name*}

no debug iua as

Syntax Description

user	Displays information about the use of application programming interfaces (APIs) and events between the ISDN layer and IUA.
state	Displays information about AS state transitions.
all	Enables debug for all the configured ASs.
name *as-name*	Defines the name of the AS.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(4)T	This command was introduced.
12.2(15)T	This command was integrated into Cisco IOS Release 12.2(15)T on the Cisco 2420, Cisco 2600 series, Cisco 3600 series, and Cisco 3700 series; and Cisco AS5300, Cisco AS5350, Cisco AS5400, and Cisco AS5850 network access server (NAS) platforms.

Examples

The following example shows debugging output when an ISDN backhaul connection is initially established. The output shows that state debugging is turned on for all ASs and that the AS is active.

```
Router# debug iua as state all
```

```
IUA :state debug turned ON for ALL AS
00:11:52:IUA:AS as1 number of ASPs up is 1
00:11:57:IUA:AS as1 xsition AS-Up -->  AS-Active, cause - ASP asp1
```

Related Commands

Command	Description
debug iua asp	Displays debugging messages for the IUA ASP.

debug iua asp

To display debugging messages for the IDSN User Adaptation Layer (IUA) application server process (ASP), use the **debug iua asp** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug iua asp {**pak| peer-msg| sctp-sig| state**} {**all| name** *asp-name*}

no debug iua asp

Syntax Description

pak	Displays information about all packets.
peer-msg	Displays information about IUA peer-to-peer messages.
sctp-sig	Displays information about the signals being sent by the Stream Control Transmission Protocol (SCTP) layer.
state	Displays information about ASP state transition.
all	Enables debugging output for all configured ASPs.
name *asp-name*	Defines the name of the ASP.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(4)T	This command was introduced.
12.2(15)T	This command was integrated into Cisco IOS Release 12.2(15)T on the Cisco 2420, Cisco 2600 series, Cisco 3600 series, and Cisco 3700 series; and Cisco AS5300, Cisco AS5350, Cisco AS5400, and Cisco AS5850 network access server (NAS) platforms.

Examples

The following example shows debugging output when an ISDN backhaul connection is initially established. The output shows that peer message debugging is turned on for all ASPs and that the ASP is active.

```
Router# debug iua asp peer-msg all
```

```
IUA :peer message debug turned ON for ALL ASPs
Router#
00:04:58:IUA :recieved ASP_UP message on ASP asp1
00:04:58:IUA:ASP asp1 xsition ASP-Down -->  ASP-Up , cause -  rcv peer
msg
ASP-UP
00:04:58:IUA:sending ACK of type 0x304 to asp asp1
00:05:03:IUA:recv ASP_ACTIVE message for ASP asp1
00:05:03:IUA:ASP asp1 xsition ASP-Up -->  ASP-Active, cause - rcv peer
msg
ASP-Active
```

Related Commands

Command	Description
debug iua as	Displays debugging messages for the IUA AS.

debug kerberos

To display information associated with the Kerberos Authentication Subsystem, use the **debug kerberos**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug kerberos

no debug kerberos

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

Kerberos is a security system that authenticates users and services without passing a cleartext password over the network. Cisco supports Kerberos under the authentication, authorization, and accounting (AAA) security system.

Use the **debug aaa authentication** command to get a high-level view of login activity. When Kerberos is used on the router, you can use the **debug kerberos**command for more detailed debugging information.

Examples

The following is part of the sample output from the **debug aaa authentication** command for a Kerberos login attempt that failed. The information indicates that Kerberos is the authentication method used.

```
Router# debug aaa authentication
AAA/AUTHEN/START (116852612): Method=KRB5
AAA/AUTHEN (116852612): status = GETUSER
AAA/AUTHEN/CONT (116852612): continue_login
AAA/AUTHEN (116852612): status = GETUSER
AAA/AUTHEN (116852612): Method=KRB5
AAA/AUTHEN (116852612): status = GETPASS
AAA/AUTHEN/CONT (116852612): continue_login
AAA/AUTHEN (116852612): status = GETPASS
AAA/AUTHEN (116852612): Method=KRB5
AAA/AUTHEN (116852612): password incorrect
AAA/AUTHEN (116852612): status = FAIL
```

The following is sample output from the **debug kerberos** command for a login attempt that was successful. The information indicates that the router sent a request to the key distribution center (KDC) and received a valid credential.

```
Router# debug kerberos
Kerberos: Requesting TGT with expiration date of 820911631
Kerberos: Sent TGT request to KDC
Kerberos: Received TGT reply from KDC
Kerberos: Received valid credential with endtime of 820911631
```

The following is sample output from the **debug kerberos** command for a login attempt that failed. The information indicates that the router sent a request to the KDC and received a reply, but the reply did not contain a valid credential.

```
Router# debug kerberos
Kerberos: Requesting TGT with expiration date of 820911731
Kerberos: Sent TGT request to KDC
Kerberos: Received TGT reply from KDC
```

```
Kerberos: Received invalid credential.
AAA/AUTHEN (425003829): password incorrect
```

The following output shows other failure messages you might see that indicate a configuration problem. The first message indicates that the router failed to find the default Kerberos realm, therefore the process failed to build a message to send to the KDC. The second message indicates that the router failed to retrieve its own IP address. The third message indicates that the router failed to retrieve the current time. The fourth message indicates the router failed to find or create a credentials cache for a user, which is usually caused by low memory availability.

```
Router# debug kerberos
Kerberos: authentication failed when parsing name
Kerberos: authentication failed while getting my address
Kerberos: authentication failed while getting time of day
Kerberos: authentication failed while allocating credentials cache
```

Related Commands

Command	Description
debug aaa authentication	Displays information on accountable events as they occur.

debug kpml

To enable Keypad Markup Language (KPML) parser and builder debugs, use the **debug kpml** command to specify the debug option.

To disable KPML parser and builder debugs, use the **no** form of this command (you must enter one option).

debug kpml [**all**| **parser**| **builder**| **error**]

no debug kpml [**all**| **parser**| **builder**| **error**]

Syntax Description

all	Enables all kpml debug tracing.
parser	Enables kpml parser tracing.
builder	Enables kpml builder tracing.
error	Enables kpml error tracing.

Command Default

no debug kpml all

Command Modes

Privileged EXEC mode

Command History

Release	Modification
12.4(9)T	This command was introduced.

Usage Guidelines

For incoming dial peers if you configure multiple DTMF negotiation methods, the first configure value takes precedence, then the second, then the third.

For incoming dial peers, the first out-of-band negotiation method takes precedence over other DTMF negotiation methods, except when rtp-nte has precedence; in this case, sip-kpml takes precedence over other out-of-band negotiation methods.

For incoming dial peers, if both sip-kpml and rtp-nte notification mechanisms are enabled and negotiated, the gateway relies on RFC 2833 notification to receive digits and a SUBSCRIBE for KPML is not initiated.

SIP KPML support complies to the IEFT draft “draft-ietf-sipping-kpml-04.txt” with the following limitations:

- The SIP gateway always initiates SUBSCRIBE in the context of an established INVITE dialog. The gateway supports receiving SUBSCRIBE in the context of an established INVITE dialog, as well as out-of-call context requests with a leg parameter in the Event header. If the request code does not match an existing INVITE dialog, the gateway sends a NOTIFY with KPML status-code 481 and sets Subscription-State to terminated.

- The gateway does not support the Globally Routable User Agent (GRUU) requirement. The Contact header in the INVITE/200 OK message generates locally from the gateway's contact information.
- The gateway always initiates persistent subscriptions, but it receives and processes persistent and one-shot subscriptions.
- The gateway supports only single-digit reporting. There is no need for inter-digit timer support. The only regular expressions supported are those which match to a single digit. For example:
    - <regex>x</regex>--Matches to any digit 0 through 9
    - <regex>1</regex>--Matches digit 1
    - <regex>[x#*ABCD]</regex>--Matches to any digit 0 through 9, # (the pound sign), * (an asterisk), or A, B, C, or D
    - <regex>[24]</regex>--Matches digits 2 or 4
    - <regex>[2-9]</regex>--Matches on any digit 2 through 9
    - <regex>[^2-9]</regex>--Matches digits 0 or 1
- The gateway does not support long key presses. Long key presses are detected and reported as a single digit press.
- Digit suppression is not supported (*pre* tag for suppressing inband digits).
- Individual stream selection is not supported. A SUBSCRIBE request for KPML applies to all audio streams in the dialog (*stream* element and *reverse* not supported).

You can configure support only on a SIP VoIP dial peer.

Examples

The following is output from the **debug kpml** command:

```
SIP call is established.  DTMF sip-kpml was negotiated.
...
//-1/xxxxxxxxxxxx/KPML/Parser/kpml_init:
//-1/xxxxxxxxxxxx/KPML/Builder/kpml_encode: encode_data=0x64E25B48
//-1/xxxxxxxxxxxx/KPML/Builder/kpml_encode_context_create: chunk_size=2k, max_allowed=16k
//-1/xxxxxxxxxxxx/KPML/Builder/kpml_encode_context_create: context=0x6488C0AC, mp=0x6488B89C
//-1/xxxxxxxxxxxx/KPML/Builder/kpml_build_request:
//-1/xxxxxxxxxxxx/KPML/Builder/kpml_build_pattern:
//-1/xxxxxxxxxxxx/KPML/Builder/kpml_build_regex_list:
//-1/xxxxxxxxxxxx/KPML/Builder/kpml_encode: malloc xml_buf=0x645E910C, length=328
//-1/xxxxxxxxxxxx/KPML/Builder/kpml_build_request:
//-1/xxxxxxxxxxxx/KPML/Builder/kpml_build_pattern:
//-1/xxxxxxxxxxxx/KPML/Builder/kpml_build_regex_list:
//-1/xxxxxxxxxxxx/KPML/Builder/kpml_build_request: length=289, buffp=0x645E9251
//-1/xxxxxxxxxxxx/KPML/Builder/kpml_encode: rc=0, encoded str=<?xml version="1.0"
encoding="UTF-8"?><kpml-request xmlns="urn:ietf:params:xml:ns:kpml-request"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="urn:ietf:params:xml:ns:kpml-request kpml-request.xsd"
version="1.0"><pattern persist="persist"><regex
tag="dtmf">[x*#ABCD]</regex></pattern></kpml-request>
//-1/xxxxxxxxxxxx/KPML/Builder/kpml_encode_context_free:
kpml_encode_context_free:mem_mgr_mempool_free: mem_refcnt(6488B89C)=0 - mempool cleanup
/-1/xxxxxxxxxxxx/SIP/Msg/ccsipDisplayMsg:
Sent:
SUBSCRIBE sip:8888@172.18.193.250:5060 SIP/2.0
Via: SIP/2.0/UDP 172.18.193.251:5060;branch=z9hG4bKFF36
From: <sip:172.18.193.251>;tag=EA330-F6
To: <sip:8888@172.18.193.250>;tag=39497C-2EA
Call-ID: 57633F68-2BE011D6-8013D46B-B4F9B5F6@172.18.193.251
```

```
CSeq: 103 SUBSCRIBE
Max-Forwards: 70
Date: Fri, 01 Mar 2002 00:16:15 GMT
User-Agent: Cisco-SIPGateway/IOS-12.x
Event: kpml
Expires: 7200
Contact: <sip:172.18.193.251:5060>
Content-Type: application/kpml-request+xml
Content-Length: 327
<?xml version="1.0" encoding="UTF-8"?><kpml-request
xmlns="urn:ietf:params:xml:ns:kpml-request"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="urn:ietf:params:xml:ns:kpml-request kpml-request.xsd"
version="1.0"><pattern persist="persist"><regex
tag="dtmf">[x*#ABCD]</regex></pattern></kpml-request>
/-1/xxxxxxxxxxxx/SIP/Msg/ccsipDisplayMsg:
Received:
SUBSCRIBE sip:172.18.193.251:5060 SIP/2.0
Via: SIP/2.0/UDP 172.18.193.250:5060;branch=z9hG4bK5FE3
From: <sip:8888@172.18.193.250>;tag=39497C-2EA
To: <sip:172.18.193.251>;tag=EA330-F6
Call-ID: 57633F68-2BE011D6-8013D46B-B4F9B5F6@172.18.193.251
CSeq: 101 SUBSCRIBE
Max-Forwards: 70
Date: Fri, 01 Mar 2002 01:02:46 GMT
User-Agent: Cisco-SIPGateway/IOS-12.x
Event: kpml
Expires: 7200
Contact: <sip:172.18.193.250:5060>
Content-Type: application/kpml-request+xml
Content-Length: 327
<?xml version="1.0" encoding="UTF-8"?><kpml-request
xmlns="urn:ietf:params:xml:ns:kpml-request"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="urn:ietf:params:xml:ns:kpml-request kpml-request.xsd"
version="1.0"><pattern persist="persist"><regex
tag="dtmf">[x*#ABCD]</regex></pattern></kpml-request>
/-1/xxxxxxxxxxxx/KPML/Parser/kpml_init:
//-1/xxxxxxxxxxxx/KPML/Parser/kpml_decode: Parsing <?xml version="1.0"
encoding="UTF-8"?><kpml-request xmlns="urn:ietf:params:xml:ns:kpml-request"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="urn:ietf:params:xml:ns:kpml-request kpml-request.xsd"
version="1.0"><pattern persist="persist"><regex
tag="dtmf">[x*#ABCD]</regex></pattern></kpml-request>
/-1/xxxxxxxxxxxx/KPML/Parser/kpml_request_ptbuild:
//-1/xxxxxxxxxxxx/KPML/Parser/kpml_create_new_node: creating node
par/cur/child=0x00000000/0x645E910C/0x00000000  top/child=0x645E910C/0x00000000
//-1/xxxxxxxxxxxx/KPML/Parser/kpml_pattern_ptbuild:
//-1/xxxxxxxxxxxx/KPML/Parser/kpml_create_new_node: creating node
par/cur/child=0x645E910C/0x645E91E8/0x00000000  top/child=0x645E910C/0x645E91E8
//-1/xxxxxxxxxxxx/KPML/Parser/kpml_regex_ptbuild:
//-1/xxxxxxxxxxxx/KPML/Parser/kpml_create_new_node: creating node
par/cur/child=0x645E91E8/0x645E923C/0x00000000  top/child=0x645E910C/0x645E91E8
//-1/xxxxxxxxxxxx/KPML/Parser/kpml_character_data:
buf=[x*#ABCD]</regex></pattern></kpml-request>
/-1/xxxxxxxxxxxx/KPML/Parser/kpml_regex_char_data_ptbuild: char data=[x*#ABCD]
//-1/xxxxxxxxxxxx/KPML/Parser/kpml_end_element_handler: elem name=regex
//-1/xxxxxxxxxxxx/KPML/Parser/kpml_end_element_handler: elem name=pattern
//-1/xxxxxxxxxxxx/KPML/Parser/kpml_end_element_handler: elem name=kpml-request
//-1/xxxxxxxxxxxx/KPML/Parser/kpml_pattern_ptproc:
//-1/xxxxxxxxxxxx/KPML/Parser/kpml_regex_ptproc:
//-1/xxxxxxxxxxxx/KPML/Parser/kpml_decode_context_free:
kpml_decode_context_free:mem_mgr_mempool_free: mem_refcnt(6488B89C)=0 - mempool cleanup
//-1/xxxxxxxxxxxx/SIP/Msg/ccsipDisplayMsg:
Received:
SIP/2.0 200 OK
Via: SIP/2.0/UDP 172.18.193.251:5060;branch=z9hG4bKFF36
From: <sip:172.18.193.251>;tag=EA330-F6
To: <sip:8888@172.18.193.250>;tag=39497C-2EA
Date: Fri, 01 Mar 2002 01:02:51 GMT
Call-ID: 57633F68-2BE011D6-8013D46B-B4F9B5F6@172.18.193.251
CSeq: 103 SUBSCRIBE
Content-Length: 0
```

```
Contact: <sip:172.18.193.250:5060>
Expires: 7200
//-1/xxxxxxxxxxxx/SIP/Msg/ccsipDisplayMsg:
Sent:
SIP/2.0 200 OK
Via: SIP/2.0/UDP 172.18.193.250:5060;branch=z9hG4bK5FE3
From: <sip:8888@172.18.193.250>;tag=39497C-2EA
To: <sip:172.18.193.251>;tag=EA330-F6
Date: Fri, 01 Mar 2002 00:16:24 GMT
Call-ID: 57633F68-2BE011D6-8013D46B-B4F9B5F6@172.18.193.251
CSeq: 101 SUBSCRIBE
Content-Length: 0
Contact: <sip:172.18.193.251:5060>
Expires: 7200
//-1/xxxxxxxxxxxx/SIP/Msg/ccsipDisplayMsg:
Sent:
NOTIFY sip:172.18.193.250:5060 SIP/2.0
Via: SIP/2.0/UDP 172.18.193.251:5060;branch=z9hG4bK101EA4
From: <sip:172.18.193.251>;tag=EA330-F6
To: <sip:8888@172.18.193.250>;tag=39497C-2EA
Call-ID: 57633F68-2BE011D6-8013D46B-B4F9B5F6@172.18.193.251
CSeq: 104 NOTIFY
Max-Forwards: 70
Date: Fri, 01 Mar 2002 00:16:24 GMT
User-Agent: Cisco-SIPGateway/IOS-12.x
Event: kpml
Subscription-State: active
Contact: <sip:172.18.193.251:5060>
Content-Length: 0
//-1/xxxxxxxxxxxx/SIP/Msg/ccsipDisplayMsg:
Received:
NOTIFY sip:172.18.193.251:5060 SIP/2.0
Via: SIP/2.0/UDP 172.18.193.250:5060;branch=z9hG4bK6111
From: <sip:8888@172.18.193.250>;tag=39497C-2EA
To: <sip:172.18.193.251>;tag=EA330-F6
Call-ID: 57633F68-2BE011D6-8013D46B-B4F9B5F6@172.18.193.251
CSeq: 102 NOTIFY
Max-Forwards: 70
Date: Fri, 01 Mar 2002 01:02:51 GMT
User-Agent: Cisco-SIPGateway/IOS-12.x
Event: kpml
Subscription-State: active
Contact: <sip:172.18.193.250:5060>
Content-Length: 0
...
//-1/xxxxxxxxxxxx/KPML/Builder/kpml_encode: encode_data=0x64E25D00
//-1/xxxxxxxxxxxx/KPML/Builder/kpml_encode_context_create: chunk_size=2k, max_allowed=16k
//-1/xxxxxxxxxxxx/KPML/Builder/kpml_encode_context_create: context=0x64FADC10, mp=0x64AFBBE0
//-1/xxxxxxxxxxxx/KPML/Builder/kpml_build_response:
//-1/xxxxxxxxxxxx/KPML/Builder/kpml_encode: malloc xml_buf=0x645E910C, length=112
//-1/xxxxxxxxxxxx/KPML/Builder/kpml_build_response:
//-1/xxxxxxxxxxxx/KPML/Builder/kpml_build_response: length=73, buffp=0x645E917B
//-1/xxxxxxxxxxxx/KPML/Builder/kpml_encode: rc=0, encoded str=<?xml version="1.0"
encoding="UTF-8"?><kpml-response version="1.0" code="200" text="OK" digits="1" tag="dtmf"/>
//-1/xxxxxxxxxxxx/KPML/Builder/kpml_encode_context_free:
kpml_encode_context_free:mem_mgr_mempool_free: mem_refcnt(64AFBBE0)=0 - mempool cleanup
//-1/xxxxxxxxxxxx/SIP/Msg/ccsipDisplayMsg:
Sent:
NOTIFY sip:172.18.193.250:5060 SIP/2.0
Via: SIP/2.0/UDP 172.18.193.251:5060;branch=z9hG4bK1117DE
From: <sip:172.18.193.251>;tag=EA330-F6
To: <sip:8888@172.18.193.250>;tag=39497C-2EA
Call-ID: 57633F68-2BE011D6-8013D46B-B4F9B5F6@172.18.193.251
CSeq: 105 NOTIFY
Max-Forwards: 70
Date: Fri, 01 Mar 2002 00:37:33 GMT
User-Agent: Cisco-SIPGateway/IOS-12.x
Event: kpml
Subscription-State: active
Contact: <sip:172.18.193.251:5060>
Content-Type: application/kpml-response+xml
Content-Length: 113
<?xml version="1.0" encoding="UTF-8"?><kpml-response version="1.0" code="200" text="OK"
```

```
digits="1" tag="dtmf"/>
/-1/xxxxxxxxxxxx/SIP/Msg/ccsipDisplayMsg:
Received:
SIP/2.0 200 OK
Via: SIP/2.0/UDP 172.18.193.251:5060;branch=z9hG4bK1117DE
From: <sip:172.18.193.251>;tag=EA330-F6
To: <sip:8888@172.18.193.250>;tag=39497C-2EA
Date: Fri, 01 Mar 2002 01:24:08 GMT
Call-ID: 57633F68-2BE011D6-8013D46B-B4F9B5F6@172.18.193.251
CSeq: 105 NOTIFY
Content-Length: 0
//-1/xxxxxxxxxxxx/KPML/Builder/kpml_encode: encode_data=0x64E25D00
//-1/xxxxxxxxxxxx/KPML/Builder/kpml_encode_context_create: chunk_size=2k, max_allowed=16k
//-1/xxxxxxxxxxxx/KPML/Builder/kpml_encode_context_create: context=0x651E8084, mp=0x65501720
//-1/xxxxxxxxxxxx/KPML/Builder/kpml_build_response:
//-1/xxxxxxxxxxxx/KPML/Builder/kpml_encode: malloc xml_buf=0x645E910C, length=112
//-1/xxxxxxxxxxxx/KPML/Builder/kpml_build_response:
//-1/xxxxxxxxxxxx/KPML/Builder/kpml_build_response: length=73, buffp=0x645E917B
//-1/xxxxxxxxxxxx/KPML/Builder/kpml_encode: rc=0, encoded str=<?xml version="1.0"
encoding="UTF-8"?><kpml-response version="1.0" code="200" text="OK" digits="2" tag="dtmf"/>
//-1/xxxxxxxxxxxx/KPML/Builder/kpml_encode_context_free:
kpml_encode_context_free:mem_mgr_mempool_free: mem_refcnt(65501720)=0 - mempool cleanup
//-1/xxxxxxxxxxxx/KPML/Builder/kpml_encode: encode_data=0x656F9128
//-1/xxxxxxxxxxxx/KPML/Builder/kpml_encode_context_create: chunk_size=2k, max_allowed=16k
//-1/xxxxxxxxxxxx/KPML/Builder/kpml_encode_context_create: context=0x651E8084, mp=0x6488B6CC
//-1/xxxxxxxxxxxx/KPML/Builder/kpml_build_response:
//-1/xxxxxxxxxxxx/KPML/Builder/kpml_encode: malloc xml_buf=0x645E910C, length=112
//-1/xxxxxxxxxxxx/KPML/Builder/kpml_build_response:
//-1/xxxxxxxxxxxx/KPML/Builder/kpml_build_response: length=73, buffp=0x645E917B
//-1/xxxxxxxxxxxx/KPML/Builder/kpml_encode: rc=0, encoded str=<?xml version="1.0"
encoding="UTF-8"?><kpml-response version="1.0" code="200" text="OK" digits="3" tag="dtmf"/>
//-1/xxxxxxxxxxxx/KPML/Builder/kpml_encode_context_free:
kpml_encode_context_free:mem_mgr_mempool_free: mem_refcnt(6488B6CC)=0 - mempool cleanup
//-1/xxxxxxxxxxxx/SIP/Msg/ccsipDisplayMsg:
Sent:
NOTIFY sip:172.18.193.250:5060 SIP/2.0
Via: SIP/2.0/UDP 172.18.193.251:5060;branch=z9hG4bK12339
From: <sip:172.18.193.251>;tag=EA330-F6
To: <sip:8888@172.18.193.250>;tag=39497C-2EA
Call-ID: 57633F68-2BE011D6-8013D46B-B4F9B5F6@172.18.193.251
CSeq: 106 NOTIFY
Max-Forwards: 70
Date: Fri, 01 Mar 2002 00:37:44 GMT
User-Agent: Cisco-SIPGateway/IOS-12.x
Event: kpml
Subscription-State: active
Contact: <sip:172.18.193.251:5060
Content-Type: application/kpml-response+xml
Content-Length: 113
<?xml version="1.0" encoding="UTF-8"?><kpml-response version="1.0" code="200" text="OK"
digits="2" tag="dtmf"/>
/-1/xxxxxxxxxxxx/SIP/Msg/ccsipDisplayMsg:
Received:
SIP/2.0 200 OK
Via: SIP/2.0/UDP 172.18.193.251:5060;branch=z9hG4bK12339
From: <sip:172.18.193.251>;tag=EA330-F6
To: <sip:8888@172.18.193.250>;tag=39497C-2EA
Date: Fri, 01 Mar 2002 01:24:20 GMT
Call-ID: 57633F68-2BE011D6-8013D46B-B4F9B5F6@172.18.193.251
CSeq: 106 NOTIFY
Content-Length: 0
...
```

Related Commands

Command	Description
show sip-ua calls	Verifies that the DTMF method is SIP-KPML.

debug kron

To display debugging messages about Command Scheduler policies or occurrences, use the **debug kron** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug kron {**all**| **exec-cli**| **info**| **major**}

no debug kron {**all**| **exec-cli**| **info**| **major**}

Syntax Description

all	Displays all debugging output about Command Scheduler policy lists or occurrences.
exec-cli	Displays detailed debugging output about Command Scheduler policy list command-line interface (CLI) commands.
info	Displays debugging output about Command Scheduler policy lists, occurrence warnings, or progress information.
major	Displays debugging output about Command Scheduler policy list or occurrence failures.

Command Default

If no keyword is specified, all debugging messages are displayed.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.3(1)	This command was introduced.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SRC	This command was integrated into Cisco IOS Release 12.2(33)SRC.
12.2(33)SB	This command was integrated into Cisco IOS Release 12.2(33)SB.
12.2(33)SXI	This command was integrated into Cisco IOS Release 12.2(33)SXI.

Usage Guidelines

Use the **debug kron** command to display the output of a scheduled EXEC **show** command on the console.

Examples

The following example shows debugging messages for the EXEC CLI **show version** after the CLI was run at a scheduled interval:

```
Router# debug kron exec-cli

Kron cli occurrence messages debugging is on
2w6d: Call parse_cmd 'show version'
2w6d: Kron CLI return 0
'
**CLI 'show version':
Cisco Internetwork Operating System Software IOS (tm) C2600 Software (C2600-I-M
```

Related Commands

Command	Description
show kron schedule	Displays the status and schedule information for Command Scheduler occurrences.

debug l2ctrl

To enable debugging for Layer 2 Control (L2CTRL), use the **debug l2ctrl** command in privileged EXEC mode. To disable debugging for L2CTRL, use the **no** form of this command.

debug l2ctrl {**all**| **evc**| **pm**| **registry**}

no debug l2ctrl {**all**| **evc**| **pm**| **registry**}

Syntax Description

all	Displays all L2CTRL debugging messages.
evc	Displays Ethernet virtual circuit (EVC) and L2CTRL messages.
pm	Displays switch PM and L2CTRL messages.
registry	Displays L2CTRL registries.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(33)SRD	This command was introduced.

Examples

The following example shows how to enable debugging of all L2CTRL related events:

```
Router# debug l2ctrl all
```

Related Commands

Command	Description
debug ethernet l2ctrl	Enables Ethernet L2CTRL debugging messages.

debug l2fib

To enable the logging of Layer 2 Forwarding Information Base (L2FIB) debug messages, use the **debug l2fib** command in privileged EXEC mode. To disable the debugging, use the **no** form of this command.

debug l2fib {**addr** [**unicast**| **multicast**]| **all**| **bridge-domain** [**port**]| **event**| **error**| **ha**| **l2port**| **mlrib**| **olist**| **otv tunnel** {**decap**| **encap**}}

no debug l2fib {**addr** [**unicast**| **multicast**]| **all**| **bridge-domain** [**port**]| **event**| **error**| **ha**| **l2port**| **mlrib**| **olist**| **otv tunnel** {**decap**| **encap**}}

Syntax Description

addr	Enables logging of unicast or multicast object-specific debug messages.
unicast	(Optional) Enables logging of unicast object-specific debug messages.
multicast	(Optional) Enables logging of multicast object-specific debug messages.
all	Enables logging of all L2FIB debug messages.
bridge-domain	Enables logging of bridge-domain object-specific debug messages.
port	Enables logging of bridge-domain port object-specific debug messages.
event	Enables logging of event debug messages.
error	Enables logging of the error debug messages.
ha	Enables logging of high availability (HA) events.
l2port	Enables logging of Layer 2 port object-specific debug messages.
mlrib	Enables logging of Multilayer Routing Information Base (MLRIB) interactions.
olist	Enables logging of output list object-specific debug messages.
otv tunnel	Enables logging of Overlay Transport Virtualization (OTV) tunnel object-specific debug messages.
decap	Enables logging of OTV tunnel decap object-specific debug messages.
encap	Enables logging of OTV tunnel encap object-specific debug messages.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS XE Release 3.5S	This command was introduced.

Examples

The following is sample output from the **debug l2fib all** command:

```
Router# debug l2fib all
```

```
[11/11/11 19:57:17.256 169DFD 276] L2FIB-MCAST-DEBUG: l2fib_mcast_obj_handle_nh_list:
Received next hop 70.1.1.2, Intf Ov47, owner 0x400, opcode 0x0, flags 0x2, for source
120.0.7.51, group 225.1.8.51, bd 1864
[11/11/11 19:57:17.256 169DFE 276] L2FIB-MCAST-DEBUG: l2fib_mcast_obj_add_oif: Added OIF
ED 70.1.1.2, Intf Ov47, to src 120.0.7.51, grp 225.1.8.51 BD 1864.
[11/11/11 19:57:17.256 169DFF 276] L2FIB-MCAST-DEBUG: l2fib_mcast_obj_add_oif: Found existing
 Otv DG mapping 232.1.47.2, map count 2 for source 120.0.7.51, group 225.1.8.51, bridge
domain 1864, OIF 70.1.1.2, Intf Ov47.
[11/11/11 19:57:17.256 169E00 276] L2FIB-HA-DEBUG: l2fib_ha_encode_mcast_nh_tlv: Encode Otv
 ED receiver port 70.1.1.2, Intf Ov47
[11/11/11 19:57:17.256 169E01 276] L2FIB-HA-DEBUG: l2fib_mcast_obj_chkp: Sync multicast
with src 120.0.7.51, grp 225.1.8.51 BD 1864, rcvrs 1, Otv DG map 1, oper 0x4.
[11/11/11 19:57:17.256 169E02 276] L2FIB-HA-DEBUG: l2fib_ha_enqueue_message: Enqueued message
 to hold queue 0
[11/11/11 19:57:17.256 169E03 276] L2FIB-MCAST-DEBUG: l2fib_mcast_obj_handle_s_g: Received
 source 120.0.7.53, group 225.1.8.53, bridge domain 1866, next hop count 1.
[11/11/11 19:57:17.256 169E04 276] L2FIB-MCAST-DEBUG: l2fib_mcast_obj_handle_nh_list:
Received next hop 70.1.1.2, Intf Ov47, owner 0x400, opcode 0x0, flags 0x2, for source
120.0.7.53, group 225.1.8.53, bd 1866
[11/11/11 19:57:17.256 169E05 276] L2FIB-MCAST-DEBUG: l2fib_mcast_obj_add_oif: Added OIF
ED 70.1.1.2, Intf Ov47, to src 120.0.7.53, grp 225.1.8.53 BD 1866.
[11/11/11 19:57:17.256 169E06 276] L2FIB-MCAST-DEBUG: l2fib_mcast_obj_add_oif: Found existing
 Otv DG mapping 232.1.47.4, map count 2 for source 120.0.7.53, group 225.1.8.53, bridge
domain 1866, OIF 70.1.1.2, Intf Ov47.
[11/11/11 19:57:17.256 169E07 276] L2FIB-HA-DEBUG: l2fib_ha_encode_mcast_nh_tlv: Encode Otv
 ED receiver port 70.1.1.2, Intf Ov47
[11/11/11 19:57:17.256 169E08 276] L2FIB-HA-DEBUG: l2fib_mcast_obj_chkp: Sync multicast
with src 120.0.7.53, grp 225.1.8.53 BD 1866, rcvrs 1, Otv DG map 1, oper 0x4.
[11/11/11 19:57:17.256 169E09 276] L2FIB-HA-DEBUG: l2fib_ha_enqueue_message: Enqueued message
 to hold queue 0
[11/11/11 19:57:17.256 169E0A 276] L2FIB-MCAST-DEBUG: l2fib_mcast_obj_handle_s_g: Received
 source 120.0.7.58, group 225.1.8.58, bridge domain 1871, next hop count 1.
```

Related Commands

Command	Description
show l2fib	Displays information about L2FIB.

debug l2relay events

To start debugging of Layer 2 Relay events, use the **debug l2relay events** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command (SGSN D-node only).

debug l2relay events

no debug l2relay events

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.1(1)GA	This command was introduced.
12.1(3)T	This command was integrated into Cisco IOS Release 12.1(3)T.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

The SGSN module uses the proprietary Layer 2 Relay protocol in conjunction with the intra-Serving GPRS Support Node (iSGSN) protocol for communication between the SGSN-datacom (SGSN-D) and SGSN-telecom (SGSN-T) units that comprise the SGSN.

For debugging purposes, it might also be useful to trace Layer 2 Relay packets. To display information about Layer 2 Relay packets, use the **debug l2relay packets** command.

Normally you will not need to use the **debug l2relay events** or **debug l2relay packets** commands. If problems with the SGSN are encountered, Cisco technical support personnel may request that issue the command.

Caution Because the **debug l2relay events**command generates a substantial amount of output, use it only when traffic on the GPRS network is low, so other activity on the system is not adversely affected.

Examples

The following example enables the display of Layer 2 Relay events:

```
Router# debug l2relay events
```

Related Commands

Command	Description
debug 12relay packets	Displays Layer 2 Relay packets (SGSN D-node only).

debug l2relay packets

To display information about Layer 2 Relay packets, use the **debug l2relay packets** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command (SGSN D-node only).

debug l2relay packets

no debug l2relay packets

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.1(1)GA	This command was introduced.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.1(3)T	This command was integrated into Cisco IOS Release 12.1(3)T.

Usage Guidelines

Use the **debug l2relay packets** command to display information about Layer 2 Relay packets.

The SGSN module uses the proprietary Layer 2 Relay protocol in conjunction with the intra-Serving GPRS Support Node (iSGSN) protocol for communication between the SGSN-datacom (SGSN-D) and SGSN-telecom (SGSN-T) units that comprise the SGSN.

For debugging purposes, it might also be useful to trace Layer 2 Relay events. To display information about Layer 2 Relay events, use the **debug l2relay events** command.

Normally you will not need to use the **debug l2relay packets**or **debug l2relay events** command. If problems with the SGSN are encountered, Cisco technical support personnel may request that you issue the command.

Caution Because the **debug l2relay packets**command generates a significant amount of output, use it only when traffic on the GPRS network is low, so other activity on the system is not adversely affected.

Examples

The following example enables the display of Layer 2 Relay packets:

```
Router# debug l2relay packets
```

Related Commands

Command	Description
debug ip igmp	Displays Layer 2 Relay events (SGSN D-node only).

debug l2tp

To enable debugging of Layer 2 Tunneling Protocol (L2TP) information, use the **debug l2tp**command in privileged EXEC mode. To disable L2TP debugging, use the **no** form of this command.

debug l2tp {**all**| **application**| **brief**| **db** {**error**| **event**| **lookup**}| **error**| **event**| **export**| **l2tun**| **packet** {**brief**| **detail**| **error**| **event**}| **route**| **seq** [**brief**]| **snmp**| **timer**}

no debug l2tp {**all**| **application**| **brief**| **db** {**error**| **event**| **lookup**}| **error**| **event**| **export**| **l2tun**| **packet** {**brief**| **detail**| **error**| **event**}| **route**| **seq** [**brief**]| **snmp**| **timer**}

Syntax Description

all	Enables the most commonly used L2TP debugs.
application	Enables L2TP application information debugs.
brief	Enables L2TP debug information in a single line.
db	Enables L2TP database debugs.
error	Enables L2TP error debugs.
event	Enables L2TP event debugs.
lookup	Enables L2TP database lookup.
export	Enables L2TP external data and command-line interface (CLI) debugs.
l2tun	Enables Layer 2 tunnel (L2Tun) socket application programming interface (API) debugs.
packet	Enables L2TP packet information debugs.
detail	Enables L2TP packet dump details debugs.
route	Enables L2TP route watch debugs.
seq	Enables extra sequencing debugs.
brief	(Optional) Enables L2TP one-line sequencing debugs.
snmp	Enables L2TP Simple Network Management Protocol (SNMP) event debugs.
timer	Enables L2TP timer debugs.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.4(2)T	This command was introduced.
12.2(33)SRB	This command was integrated into Cisco IOS Release 12.2(33)SRB.
12.2(33)SB	This command was integrated into Cisco IOS Release 12.2(33)SB.
15.0(1)M	This command was modified. The **application** and **brief** keywords were added.
15.0(1)S	This command was modified. The **snmp** and **route** keywords were added.

Usage Guidelines

Use the **debug l2tp** command to troubleshoot L2TP operations.

Examples

The following example shows how to enable L2TP debugging:

```
Router> enable
Router# debug l2tp all
L2TP most commonly used debugs debugging is on
Router# debug l2tp application
L2TP application debugs debugging is on
Router# debug l2tp brief
L2TP brief, one line debugs debugging is on
Router# debug l2tp db lookup

L2TP database lookups debugging is on
Router# debug l2tp error
L2TP errors debugging is on
Router# debug l2tp seq
L2TP sequencing debugging is on
Router# debug l2tp snmp
L2TP SNMP events debugging is on
```

The following sample output of the **show debugging** command displays the debugs enabled for L2TP. The field descriptions are self-explanatory.

```
Router# show debugging

L2TP:
  L2TP packet events debugging is on
  L2TP packet errors debugging is on
  L2TP packet detail debugging is on
  L2TP errors debugging is on
  L2TP events debugging is on
  L2TP L2TUN socket API debugging is on
  L2TP sequencing debugging is on
  L2TP export data to applications and cli debugging is on
  L2TP route watch debugging is on
  L2TP timers debugging is on
  L2TP brief, one line debugs debugging is on
  L2TP application debugs debugging is on
```

```
L2TP database lookups debugging is on
L2TP SNMP events debugging is on
```

Related Commands

Command	Description
show debugging	Displays information about the types of debugging that are enabled for your router.

debug l2tp redundancy

To enable the display of information on Layer 2 Tunneling Protocol (L2TP) sessions that contain redundancy status, use the **debug l2tp redundancy**command in user or privileged EXEC mode. To disable this debugging, use the **no** form of this command.

debug l2tp redundancy {**cf| detail| error| event| fsm| resync| rf**}

no debug l2tp redundancy

Syntax Description

cf	Displays L2TP redundancy-facility (cf) events.
detail	Displays L2TP redundancy details.
error	Displays L2TP redundancy errors.
event	Displays L2TP redundancy events.
fsm	Displays L2TP redundancy forwarding-service manager (fsm) events.
resync	Displays L2TP redundancy resynchronizations.
rf	Displays L2TP redundancy-facility (rf) events.

Command Modes

User EXEC (>) Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS XE Release 2.2.	This command was introduced in Cisco IOS XE Release 2.2.

Usage Guidelines

Use the **debug l2tp redundancy**command in privileged EXEC mode to display a list of redundancy events and errors.

Use the **show l2tp redundancy**command in privileged EXEC mode to display information on the state of the Layer 2 Tunneling Protocol (L2TP) or a specific L2TP session redundancy data.

Examples

The following example shows how to display a debug of redundancy events during the setup and termination of an L2TP High Availability (HA) tunnel for a L2TP Network Server (LNS) active Route Processor (RP):

```
LNS1> debug
 enable
LNS1# debug
```

```
 l2tp
 redundancy
 cf
L2TP redundancy cf debugging is on
LNS1# debug
 l2tp
 redundancy
 detail
L2TP redundancy details debugging is on
LNS1# debug
 l2tp
 redundancy
 error
L2TP redundancy errors debugging is on
LNS1# debug
 l2tp
 redundancy
 event
L2TP redundancy events debugging is on
LNS1# debug
 l2tp
 redundancy
 fsm
L2TP redundancy fsm debugging is on
LNS1# debug
 l2tp
 redundancy
 resync
L2TP redundancy resync debugging is on
LNS1# debug
 l2tp
 redundancy
 rf
L2TP redundancy rf debugging is on
LNS1#
*Aug 26 18:00:00.467: %SYS-5-CONFIG_I: Configured from console by console
LNS1#
*Aug 26 18:00:45.631: L2TP tnl   01000:________: CCM initialized CCM session
*Aug 26 18:00:45.631: : L2TP HA:CC playback chkpt skipped, CC not doing HA
*Aug 26 18:00:45.711: : L2TP HA FSM:Receive proto FSM event 19
*Aug 26 18:00:45.711: : L2TP HA FSM:Receive RxSCCRQ
*Aug 26 18:00:45.711: : L2TP HA:lcm_cc alloc: l2tp_cc 070B45B8, lcm_cc 02FE55E8
*Aug 26 18:00:45.711: : L2TP HA FSM:FSM-CC ev Rx-SCCRQ
*Aug 26 18:00:45.711: : L2TP HA FSM:FSM-CC    Idle->Wt-ChkptSidRmt
*Aug 26 18:00:45.711: : L2TP HA FSM:FSM-CC do Block-Tx-AckSCCRQ
*Aug 26 18:00:45.711: : L2TP HA FSM:Checkpoint Two Cc IDs
*Aug 26 18:00:45.711: L2TP HA CF: Chkpt send: s/c id 0/52631, BothCcId, seq 0, ns/nr 0/0,
rid 51583, len 52; flush = 1, ctr 1
*Aug 26 18:00:45.711: 01000:0000CD97: L2TP HA:Enqueue peer Ns 0 to ns_q, seq 1 (q sz 0)
*Aug 26 18:00:45.711: L2TP tnl   01000:0000CD97: Encoding SCCRQ-IN CHKPT
*Aug 26 18:00:45.711: L2TP tnl   01000:0000CD97: Tx CHKPT
*Aug 26 18:00:45.739: L2TP tnl   01000:0000CD97: Encoding SCCRP-OUT CHKPT
*Aug 26 18:00:45.739: L2TP tnl   01000:0000CD97: Tx CHKPT
*Aug 26 18:00:45.739: : L2TP HA:Adjust local window size to 10
*Aug 26 18:00:45.739: 01000:0000CD97: L2TP HA FSM:Receive proto TxCM event SCCRP
*Aug 26 18:00:45.739: : L2TP HA FSM:Receive TxSCCRP
LNS1#
*Aug 26 18:00:45.739: : L2TP HA FSM:FSM-CC ev Tx-SCCRP
*Aug 26 18:00:45.739: : L2TP HA FSM:FSM-CC    Wt-ChkptSidRmt->WtCcIdRmt2
*Aug 26 18:00:45.739: : L2TP HA FSM:FSM-CC do Block-Tx-SCCRP
*Aug 26 18:00:45.739: 01000:0000CD97: L2TP HA FSM:Found blocked RxSCCRQ, seq_num 1
*Aug 26 18:00:45.739: 01000:0000CD97: L2TP HA FSM:Queued SCCRP to CC hold_q
*Aug 26 18:00:46.863: : L2TP HA FSM:CHKPT status callback: status 0, len 56
*Aug 26 18:00:46.863: : L2TP HA FSM:Context s/c id 0/52631, BothCcId, seq 1, ns/nr 0/0, rid
 51583, len 52
*Aug 26 18:00:46.863: L2TP HA CF: Rcvd status s/c id 0/52631, BothCcId, seq 1, ns/nr 0/0,
rid 51583, len 52
*Aug 26 18:00:46.863: L2TP HA CF: Rcvd status 0: len 56
*Aug 26 18:00:46.863: L2TP HA CF: Status content s/c id 0/52631, BothCcId, seq 1, ns/nr
0/0, rid 51583, len 52
*Aug 26 18:00:46.863: 01000:0000CD97: L2TP HA FSM:Recv chkpt ack: s/c id 0/52631, BothCcId,
 seq 1, ns/nr 0/0, rid 51583, len 52
```

```
*Aug 26 18:00:46.863: : L2TP HA FSM:Receive CC-ChkptAck
*Aug 26 18:00:46.863: : L2TP HA FSM:FSM-CC ev Rx-CkpACK-CcID-Rmt
*Aug 26 18:00:46.863: : L2TP HA FSM:FSM-CC    WtCcIdRmt2->Wt-RxSccn
*Aug 26 18:00:46.863: : L2TP HA FSM:FSM-CC do Allow-Tx-SCCRP2
*Aug 26 18:00:46.863: : L2TP HA FSM:Received Chkpt of local + remote CC ID
*Aug 26 18:00:46.863: 01000:0000CD97: L2TP HA:Try to remove from CC's ns_q: seq num 1
(current Ns 1)
*Aug 26 18:00:46.863: 01000:0000CD97: L2TP HA:Ns entry to remove: found (current Ns 1)
*Aug 26 18:00:46.863: 01000:0000CD97: L2TP HA:Advance peer Nr to 1 (ns_q sz 0)
*Aug 26 18:00:46.863: 01000:0000CD97: L2TP HA:CC send all unblocked if can
LNS1#
*Aug 26 18:00:46.863: 01000:0000CD97: L2TP HA:CC send one blocked CM (SCCRP): ns 0 (0), nr
 1
*Aug 26 18:00:46.863: L2TP HA CF: O SCCRP 51583/0 ns/nr 0/1
*Aug 26 18:00:47.867: 01000:0000CD97: L2TP HA FSM:Receive Cm-Ack
*Aug 26 18:00:47.867: 01000:0000CD97: L2TP HA FSM:Receive CC Cm-Ack
*Aug 26 18:00:47.867: : L2TP HA FSM:FSM-CC ev Rx-CmACK
*Aug 26 18:00:47.867: : L2TP HA FSM:FSM-CC    in Wt-RxSccn
*Aug 26 18:00:47.867: : L2TP HA FSM:FSM-CC do Ignore
*Aug 26 18:00:47.867: 01000:0000CD97: L2TP HA FSM:Ignore event
*Aug 26 18:00:47.867: 01000:0000CD97: L2TP HA FSM:Check for Ns/Nr update 1, peer 1
*Aug 26 18:00:47.867: 01000:0000CD97: L2TP HA FSM:Receive peer Ns/Nr update (1,0/1,1, int
1, rx 1, 1) (ns_q sz 0)
*Aug 26 18:00:47.867: 01000:0000CD97: L2TP HA FSM:Peer Ns 1 (1), Nr 1 (ns_q sz 0)
*Aug 26 18:00:48.087: 01000:0000CD97: L2TP HA FSM:Check for Ns/Nr update 1, peer 1
*Aug 26 18:00:48.087: 01000:0000CD97: L2TP HA FSM:Receive peer Ns/Nr update (1,0/1,1, int
1, rx 1, 1) (ns_q sz 0)
*Aug 26 18:00:48.087: 01000:0000CD97: L2TP HA FSM:Peer Ns 1 (2), Nr 1 (ns_q sz 0)
*Aug 26 18:00:48.087: : L2TP HA FSM:Receive proto FSM event 21
*Aug 26 18:00:48.087: 01000:0000CD97: L2TP HA FSM:Receive RxSCCCN
*Aug 26 18:00:48.087: : L2TP HA FSM:FSM-CC ev Rx-SCCCN
*Aug 26 18:00:48.087: : L2TP HA FSM:FSM-CC    Wt-RxSccn->WtCcsUp
*Aug 26 18:00:48.087: : L2TP HA FSM:FSM-CC do Allow-Tx-AckSCCCN
*Aug 26 18:00:48.087: 01000:0000CD97: L2TP HA FSM:Allow TxSCCCN-ACK
*Aug 26 18:00:48.087: 01000:0000CD97: L2TP HA FSM:Receive CcUp
*Aug 26 18:00:48.087: : L2TP HA FSM:FSM-CC ev Proto CcUp
*Aug 26 18:00:48.087: : L2TP HA FSM:FSM-CC    WtCcsUp->Wt-CkptCcUp
*Aug 26 18:00:48.087: : L2TP HA FSM:FSM-CC do Chkpt-CcUp2
*Aug 26 18:00:48.087: : L2TP HA FSM:Checkpoint CcUp
*Aug 26 18:00:48.087: L2TP HA CF: Chkpt send: s/c id 0/52631, CcUp, seq 0, ns/nr 1/1, rid
0, len 52; flush = 1, ctr 2
*Aug 26 18:00:48.091: L2TP tnl   01000:0000CD97: CCM added sync data
*Aug 26 18:00:48.095: 01000:0000CD97: L2TP HA FSM:Check for Ns/Nr update 2, peer 1
*Aug 26 18:00:48.095: 01000:0000CD97: L2TP HA FSM:Receive peer Ns/Nr update (2,1/1,1, int
2, rx 1, 2) (ns_q sz 0)
*Aug 26 18:00:48.095: 01000:0000CD97: L2TP HA FSM:Peer Ns 2 (3), Nr 1 (ns_q sz 0)
*Aug 26 18:00:48.095: L2TP _____:01000:000036F8: Encoding ICRQ-IN CHKPT
*Aug 26 18:00:48.095: L2TP _____:01000:000036F8: Tx CHKPT
*Aug 26 18:00:48.095: : L2TP HA FSM:Receive proto FSM event 3
*Aug 26 18:00:48.095: : L2TP HA FSM:Receive RxICRQ
*Aug 26 18:00:48.095: _____:01000:000036F8: L2TP HA FSM:  Using ICRQ FSM
*Aug 26 18:00:48.095: _____:01000:000036F8: L2TP HA FSM:FSM-Sn ev created
*Aug 26 18:00:48.095: _____:01000:000036F8: L2TP HA FSM:FSM-Sn    Init->Idle
*Aug 26 18:00:48.095: _____:01000:000036F8: L2TP HA FSM:FSM-Sn do none
*Aug 26 18:00:48.095: _____:01000:000036F8: L2TP HA FSM:FSM-Sn ev Rx-xCRQ
*Aug 26 18:00:48.095: _____:01000:000036F8: L2TP HA FSM:FSM-Sn    Idle->Wt-ChkptSidRmt
*Aug 26 18:00:48.095: _____:01000:000036F8: L2TP HA FSM:FSM-Sn do Block-Tx-AckXCRQ
*Aug 26 18:00:48.095: _____:01000:000036F8: L2TP HA FSM:Checkpoint TwoSessionIDs
*Aug 26 18:00:48.095: L2TP HA CF: Chkpt send: s/c id 14072/52631, BothSesId, seq 0, ns/nr
1/2, rid 40276, len 52; flush = 1, ctr 3
*Aug 26 18:00:48.095: _____:01000:000036F8: L2TP HA:Enqueue peer Ns 2 to ns_q, seq 3 (q sz
 0)
*Aug 26 18:00:48.131: : L2TP HA:Try to buffer sock msg type 19
*Aug 26 18:00:48.131: : L2TP HA:Buffering skipped
*Aug 26 18:00:48.131: L2TP _____:01000:000036F8: Encoding ICRP-OUT CHKPT
*Aug 26 18:00:48.131: L2TP _____:01000:000036F8: Tx CHKPT
*Aug 26 18:00:48.131: 01000:0000CD97: L2TP HA FSM:Receive proto TxCM event ICRP
*Aug 26 18:00:48.131: _____:_____:000036F8: L2TP HA FSM:Receive TxICRP
*Aug 26 18:00:48.131: _____:01000:000036F8: L2TP HA FSM:FSM-Sn ev Tx-xCRP
*Aug 26 18:00:48.131: _____:01000:000036F8: L2TP HA FSM:FSM-Sn    Wt-ChkptSidRmt->Wt-SesIdRmt2
*Aug 26 18:00:48.131: _____:01000:000036F8: L2TP HA FSM:FSM-Sn do Block-Tx-xCRP
*Aug 26 18:00:48.131: _____:01000:000036F8: L2TP HA FSM:Found blocked RxICRQ, seq_num 3
LNS1#
```

```
*Aug 26 18:00:48.131: _____:01000:000036F8: L2TP HA FSM:Queued xCRP to session hold_q
*Aug 26 18:00:48.131: : L2TP HA:Try to buffer sock msg type 23
*Aug 26 18:00:48.131: : L2TP HA:CC not in resync state, buffering skipped
*Aug 26 18:00:49.115: 01000:0000CD97: L2TP HA FSM:Check for Ns/Nr update 2, peer 1
*Aug 26 18:00:49.115: 01000:0000CD97: L2TP HA FSM:Receive peer Ns/Nr update (2,2/1,1, int
3, rx 1, 3) (ns_q sz 1)
*Aug 26 18:00:49.211: : L2TP HA FSM:CHKPT status callback: status 0, len 56
*Aug 26 18:00:49.211: : L2TP HA FSM:Context s/c id 0/52631, CcUp, seq 2, ns/nr 1/1, rid 0,
 len 52
*Aug 26 18:00:49.211: : L2TP HA FSM:CHKPT status callback: status 0, len 56
*Aug 26 18:00:49.211: : L2TP HA FSM:Context s/c id 14072/52631, BothSesId, seq 3, ns/nr
1/2, rid 40276, len 52
*Aug 26 18:00:49.211: L2TP HA CF: Rcvd status s/c id 0/52631, CcUp, seq 2, ns/nr 1/1, rid
0, len 52
*Aug 26 18:00:49.211: L2TP HA CF: Rcvd status 0: len 56
*Aug 26 18:00:49.211: L2TP HA CF: Status content s/c id 0/52631, CcUp, seq 2, ns/nr 1/1,
rid 0, len 52
*Aug 26 18:00:49.211: 01000:0000CD97: L2TP HA FSM:Recv chkpt ack: s/c id 0/52631, CcUp, seq
 2, ns/nr 1/1, rid 0, len 52
*Aug 26 18:00:49.211: : L2TP HA FSM:Receive CC-ChkptAck
*Aug 26 18:00:49.211: : L2TP HA FSM:FSM-CC ev Rx-CkpACK-CcUp
*Aug 26 18:00:49.211: : L2TP HA FSM:FSM-CC    Wt-CkptCcUp->ProcCcsUp
*Aug 26 18:00:49.211: : L2TP HA FSM:FSM-CC do Proc-ChpACK-CcUp2
*Aug 26 18:00:49.211: : L2TP HA FSM:Received chkpt ACK of CcUp
*Aug 26 18:00:49.211: L2TP HA CF: Rcvd status s/c id 14072/52631, BothSesId, seq 3, ns/nr
1/2, rid 40276, len 52
*Aug 26 18:00:49.211: L2TP HA CF: Rcvd status 0: len 56
*Aug 26 18:00:49.211: L2TP HA CF: Status content s/c id 14072/52631, BothSesId, seq 3, ns/nr
 1/2, rid 40276, len 52
*Aug 26 18:00:49.211: 01000:0000CD97: L2TP HA FSM:Recv chkpt ack: s/c id 14072/52631,
BothSesId, seq 3, ns/nr 1/2, rid 40276, len 52
*Aug 26 18:00:49.211: _____:_____:000036F8: L2TP HA FSM:Receive Session-ChkptAck
*Aug 26 18:00:49.211: _____:01000:000036F8: L2TP HA FSM:FSM-Sn ev Rx-CktACK-SesID-Rmt
*Aug 26 18:00:49.211: _____:01000:000036F8: L2TP HA FSM:FSM-Sn    Wt-SesIdRmt2->Wt-RxXccn
*Aug 26 18:00:49.211: _____:01000:000036F8: L2TP HA FSM:FSM-Sn do Allow-Tx-xCRP
*Aug 26 18:00:49.211: 01000:0000CD97: L2TP HA:Try to remove from CC's ns_q: seq num 3
(current Ns 3)
*Aug 26 18:00:49.211: _____:01000:000036F8: L2TP HA:Ns entry to remove: found (current Ns
3)
*Aug 26 18:00:49.211: _____:01000:000036F8: L2TP HA:Advance peer Nr to 3 (ns_q sz 0)
*Aug 26 18:00:49.211: _____:01000:000036F8: L2TP HA:Session send all unblocked
*Aug 26 18:00:49.211: 01000:0000CD97: L2TP HA:CC send if can (ICRP): ns 1 (1, 1), nr 3 (3)
*Aug 26 18:00:49.211: L2TP HA CF: O ICRP 51583/40276 ns/nr 1/3
*Aug 26 18:00:49.231: 01000:0000CD97: L2TP HA FSM:Receive Cm-Ack
*Aug 26 18:00:49.231: _____:_____:000036F8: L2TP HA FSM:Receive session Cm-Ack
LNS1#
*Aug 26 18:00:49.231: _____:01000:000036F8: L2TP HA FSM:FSM-Sn ev Rx-CmACK
*Aug 26 18:00:49.231: _____:01000:000036F8: L2TP HA FSM:FSM-Sn    in Wt-RxXccn
*Aug 26 18:00:49.231: _____:01000:000036F8: L2TP HA FSM:FSM-Sn do Ignore
*Aug 26 18:00:49.231: _____:01000:000036F8: L2TP HA FSM:Ignore event
*Aug 26 18:00:49.231: 01000:0000CD97: L2TP HA FSM:Check for Ns/Nr update 3, peer 2
*Aug 26 18:00:49.231: 01000:0000CD97: L2TP HA FSM:Receive peer Ns/Nr update (3,2/2,2, int
3, rx 2, 3) (ns_q sz 0)
*Aug 26 18:00:49.231: 01000:0000CD97: L2TP HA FSM:Peer Ns 3 (3), Nr 2 (ns_q sz 0)
LNS1#
*Aug 26 18:00:50.407: 01000:0000CD97: L2TP HA FSM:Check for Ns/Nr update 3, peer 2
*Aug 26 18:00:50.407: 01000:0000CD97: L2TP HA FSM:Receive peer Ns/Nr update (3,2/2,2, int
3, rx 2, 3) (ns_q sz 0)
*Aug 26 18:00:50.407: 01000:0000CD97: L2TP HA FSM:Peer Ns 3 (4), Nr 2 (ns_q sz 0)
*Aug 26 18:00:50.407: : L2TP HA FSM:Receive proto FSM event 5
*Aug 26 18:00:50.407: _____:_____:000036F8: L2TP HA FSM:Receive RxICCN
*Aug 26 18:00:50.407: _____:01000:000036F8: L2TP HA FSM:FSM-Sn ev Rx-xCCN
*Aug 26 18:00:50.407: _____:01000:000036F8: L2TP HA FSM:FSM-Sn    Wt-RxXccn->Wt-SessUp
*Aug 26 18:00:50.407: _____:01000:000036F8: L2TP HA FSM:FSM-Sn do Allow-Tx-AckXCCN
*Aug 26 18:00:50.407: _____:01000:000036F8: L2TP HA FSM:Allow TxICCN-ACK
*Aug 26 18:00:50.407: L2TP _____:01000:000036F8: Encoding ICCN-IN CHKPT
*Aug 26 18:00:50.407: L2TP _____:01000:000036F8: Tx CHKPT
*Aug 26 18:00:50.407: _____:_____:000036F8: L2TP HA FSM:Receive SessionUp
*Aug 26 18:00:50.407: _____:01000:000036F8: L2TP HA FSM:FSM-Sn ev Proto SessUp
*Aug 26 18:00:50.407: _____:01000:000036F8: L2TP HA FSM:FSM-Sn    Wt-SessUp->Wt-CkptSesUp
*Aug 26 18:00:50.407: _____:01000:000036F8: L2TP HA FSM:FSM-Sn do Chkpt-SesUp2
*Aug 26 18:00:50.407: _____:01000:000036F8: L2TP HA FSM:Checkpoint SessionUP
*Aug 26 18:00:50.407: L2TP HA CF: Chkpt send: s/c id 14072/52631, SesUp, seq 0, ns/nr 2/3,
```

```
 rid 0, len 52; flush = 1, ctr 4
*Aug 26 18:00:51.055: : L2TP HA FSM:CHKPT status callback: status 0, len 56
*Aug 26 18:00:51.055: : L2TP HA FSM:Context s/c id 14072/52631, SesUp, seq 4, ns/nr 2/3,
rid 0, len 52
*Aug 26 18:00:51.055: L2TP HA CF: Rcvd status s/c id 14072/52631, SesUp, seq 4, ns/nr 2/3,
 rid 0, len 52
*Aug 26 18:00:51.055: L2TP HA CF: Rcvd status 0: len 56
*Aug 26 18:00:51.055: L2TP HA CF: Status content s/c id 14072/52631, SesUp, seq 4, ns/nr
2/3, rid 0, len 52
*Aug 26 18:00:51.055: 01000:0000CD97: L2TP HA FSM:Recv chkpt ack: s/c id 14072/52631, SesUp,
 seq 4, ns/nr 2/3, rid 0, len 52
*Aug 26 18:00:51.055: _____:_____:000036F8: L2TP HA FSM:Receive Session-ChkptAck
*Aug 26 18:00:51.055: _____:01000:000036F8: L2TP HA FSM:FSM-Sn ev Rx-CktACK-SesUp
*Aug 26 18:00:51.055: _____:01000:000036F8: L2TP HA FSM:FSM-Sn    Wt-CkptSesUp->Proc-SessUp
*Aug 26 18:00:51.055: _____:01000:000036F8: L2TP HA FSM:FSM-Sn do Proc-ChpACK-SesUp
*Aug 26 18:00:51.055: _____:01000:000036F8: L2TP HA FSM:Received chkpt ACK of SessionUP
*Aug 26 18:00:51.347: %LINK-3-UPDOWN: Interface Virtual-Access2, changed state to up
LNS1#
*Aug 26 18:00:51.635: : L2TP HA:Try to buffer sock msg type 26
*Aug 26 18:00:51.635: : L2TP HA:CC not in resync state, buffering skipped
*Aug 26 18:00:51.659: : L2TP HA:Try to buffer sock msg type 26
*Aug 26 18:00:51.659: : L2TP HA:CC not in resync state, buffering skipped
LNS1#
*Aug 26 18:00:52.363: %LINEPROTO-5-UPDOWN: Line protocol on Interface Virtual-Access2,
changed state to up
LNS1#
LNS1# clear
 l2tp
 all
Proceed with clearing all tunnels? [confirm]
LNS1#
*Aug 26 18:01:21.271: 00001:_____:000036F8: L2TP HA FSM:Receive Session-CC-Rm
*Aug 26 18:01:21.271: 00001:_____:000036F8: L2TP HA FSM:Receive SessionRm
*Aug 26 18:01:21.271: 01000:0000CD97: L2TP HA FSM:Receive proto TxCM event StopCCN
*Aug 26 18:01:21.271: 01000:0000CD97: L2TP HA FSM:Receive TxSTOPCCN
*Aug 26 18:01:21.271: : L2TP HA FSM:FSM-CC ev Tx-STOPCCN
*Aug 26 18:01:21.271: : L2TP HA FSM:FSM-CC    ProcCcsUp->Wt-CkptCcDn
*Aug 26 18:01:21.271: : L2TP HA FSM:FSM-CC do Chkpt-CcDwn
*Aug 26 18:01:21.271: 01000:0000CD97: L2TP HA FSM:Receive TxSTOPCCN while CC up
*Aug 26 18:01:21.271: 01000:0000CD97: L2TP HA:CC ns_q cleanup: overall head Ns old/new =
4/4 (Q sz 0)
LNS1#
*Aug 26 18:01:21.271: : L2TP HA FSM:Checkpoint CCDown
*Aug 26 18:01:21.271: L2TP HA CF: Chkpt send: s/c id 0/52631, CcDwn, seq 0, ns/nr 2/3, rid
 0, len 52; flush = 1, ctr 5
*Aug 26 18:01:21.271: 01000:0000CD97: L2TP HA FSM:Queued STOPCCN to cc hold_q
*Aug 26 18:01:21.295: : L2TP HA:Try to buffer sock msg type 22
*Aug 26 18:01:21.295: : L2TP HA:Buffering skipped
*Aug 26 18:01:22.423: : L2TP HA FSM:CHKPT status callback: status 0, len 56
*Aug 26 18:01:22.423: : L2TP HA FSM:Context s/c id 0/52631, CcDwn, seq 5, ns/nr 2/3, rid
0, len 52
*Aug 26 18:01:22.423: L2TP HA CF: Rcvd status s/c id 0/52631, CcDwn, seq 5, ns/nr 2/3, rid
 0, len 52
*Aug 26 18:01:22.423: L2TP HA CF: Rcvd status 0: len 56
*Aug 26 18:01:22.423: L2TP HA CF: Status content s/c id 0/52631, CcDwn, seq 5, ns/nr 2/3,
rid 0, len 52
*Aug 26 18:01:22.423: 01000:0000CD97: L2TP HA FSM:Recv chkpt ack: s/c id 0/52631, CcDwn,
seq 5, ns/nr 2/3, rid 0, len 52
*Aug 26 18:01:22.423: : L2TP HA FSM:Receive CC-ChkptAck
*Aug 26 18:01:22.423: : L2TP HA FSM:FSM-CC ev Rx-CkpACK-CcDwn
*Aug 26 18:01:22.423: : L2TP HA FSM:FSM-CC    Wt-CkptCcDn->Wt-RxStopAck
*Aug 26 18:01:22.423: : L2TP HA FSM:FSM-CC do Allow-Tx-STOPCCN4
*Aug 26 18:01:22.423: : L2TP HA FSM:Received Chkpt of CC removal
*Aug 26 18:01:22.423: 01000:0000CD97: L2TP HA:Try to remove from CC's ns_q: seq num 5
(current Ns 4)
*Aug 26 18:01:22.423: 01000:0000CD97: L2TP HA:Ns entry to remove: not found (current Ns 4)
*Aug 26 18:01:22.423: 01000:0000CD97: L2TP HA:CC send all unblocked if can
*Aug 26 18:01:22.423: 01000:0000CD97: L2TP HA:CC send one blocked CM (SCCRP): ns 2 (2), nr
 4
*Aug 26 18:01:22.451: 01000:0000CD97: L2TP HA FSM:Receive Cm-Ack
*Aug 26 18:01:22.451: 01000:0000CD97: L2TP HA FSM:Receive CC Cm-Ack
*Aug 26 18:01:22.451: : L2TP HA FSM:FSM-CC ev Rx-CmACK
*Aug 26 18:01:22.451: : L2TP HA FSM:FSM-CC    Wt-RxStopAck->Wt-CkptCcRm
```

```
*Aug 26 18:01:22.451: : L2TP HA FSM:FSM-CC do ChkptCcRm3
*Aug 26 18:01:22.451: : L2TP HA FSM:Received STOPCCN-ACK while waiting for it, checkpoint
CCRm and remove cc
*Aug 26 18:01:22.451: 01000:0000CD97: L2TP HA:CC ns_q cleanup: overall head Ns old/new =
4/4 (Q sz 0)
*Aug 26 18:01:22.451: 01000:0000CD97: L2TP HA FSM:Checkpoint CcRm
*Aug 26 18:01:22.451: L2TP HA CF: Chkpt send: s/c id 0/52631, CcRm, seq 0, ns/nr 3/3, rid
0, len 52; flush = 1, ctr 6
*Aug 26 18:01:22.451: 01000:0000CD97: L2TP HA FSM:Check for Ns/Nr update 4, peer 3
*Aug 26 18:01:22.451: 01000:0000CD97: L2TP HA FSM:Receive peer Ns/Nr update (4,3/3,3, int
4, rx 3, 4) (ns_q sz 0)
*Aug 26 18:01:22.451: 01000:0000CD97: L2TP HA FSM:Peer Ns 4 (4), Nr 3 (ns_q sz 0)
*Aug 26 18:01:22.451: 01000:0000CD97: L2TP HA FSM:Receive CC-Rm
*Aug 26 18:01:22.451: : L2TP HA FSM:FSM-CC ev Proto CcRm
*Aug 26 18:01:22.451: : L2TP HA FSM:FSM-CC    Wt-CkptCcRm->End
*Aug 26 18:01:22.451: : L2TP HA FSM:FSM-CC do RmCc3
*Aug 26 18:01:22.451: 01000:0000CD97: L2TP HA FSM:CC destruction after Tx/Rx StopCCN
LNS1#
*Aug 26 18:01:22.451: 01000:0000CD97: L2TP HA:CC ns_q cleanup: overall head Ns old/new =
4/4 (Q sz 0)
*Aug 26 18:01:22.451: : L2TP HA FSM:Checkpoint CCRm
*Aug 26 18:01:22.451: L2TP HA CF: Chkpt send: s/c id 0/52631, CcRm, seq 0, ns/nr 3/3, rid
0, len 52; flush = 1, ctr 7
*Aug 26 18:01:22.451: : L2TP HA:lcm_cc free: l2tp_cc 070B45B8, lcm_cc 02FE55E8
*Aug 26 18:01:22.451: L2TP tnl   _____:________: CCM setting state to DOWN
*Aug 26 18:01:23.571: : L2TP HA FSM:CHKPT status callback: status 0, len 56
*Aug 26 18:01:23.571: : L2TP HA FSM:Context s/c id 0/52631, CcRm, seq 6, ns/nr 3/3, rid 0,
 len 52
*Aug 26 18:01:23.571: : L2TP HA FSM:CHKPT status callback: status 0, len 56
*Aug 26 18:01:23.571: : L2TP HA FSM:Context s/c id 0/52631, CcRm, seq 7, ns/nr 3/3, rid 0,
 len 52
*Aug 26 18:01:23.571: L2TP HA CF: Rcvd status s/c id 0/52631, CcRm, seq 6, ns/nr 3/3, rid
0, len 52
*Aug 26 18:01:23.571: L2TP HA CF: Rcvd status 0: len 56
*Aug 26 18:01:23.571: L2TP HA CF: Status content s/c id 0/52631, CcRm, seq 6, ns/nr 3/3,
rid 0, len 52
*Aug 26 18:01:23.571: : L2TP HA FSM:Ignore chkpt ACK: CC not found.
LNS1#
*Aug 26 18:01:23.571: L2TP HA CF: Rcvd status s/c id 0/52631, CcRm, seq 7, ns/nr 3/3, rid
0, len 52
*Aug 26 18:01:23.571: L2TP HA CF: Rcvd status 0: len 56
*Aug 26 18:01:23.571: L2TP HA CF: Status content s/c id 0/52631, CcRm, seq 7, ns/nr 3/3,
rid 0, len 52
*Aug 26 18:01:23.571: : L2TP HA FSM:Ignore chkpt ACK: CC not found.
LNS1#
*Aug 26 18:01:35.771: %REDUNDANCY-3-STANDBY_LOST: Standby processor fault
(PEER_DOWN_INTERRUPT)
```

The table below describes the significant fields shown in the **debug l2tp redundancy** command output.

Table 95: debug l2tp redundancy Command Field Descriptions

Field	Description
cf	Number of L2TP checkpointing-facility events (cf-events).
error	Number of L2TP checkpointing errors.
event	Number of L2TP checkpointing events.
fsm	Number of L2TP checkpointing fsm events.
resync	Number of L2TP checkpointing resynchronized events.

Field	Description
rf	Number of L2TP checkpointing redundancy-facility events (rf-events).

Related Commands

Command	Description
debug vpdn redundancy	Displays information about VPDN sessions that have redundancy events and errors.
l2tp sso enable	Enables L2TP HA.
l2tp tunnel resync	Specifies the number of packets sent before waiting for an acknowledgment message.
show l2tp redundancy	Displays L2TP sessions containing redundancy data.
show vpdn redundancy	Displays VPDN sessions containing redundancy data.
sso enable	Enables L2TP HA session for VPDN groups.

debug l2vpn acircuit

To debug errors and events that occur on the Layer 2 VPN (L2VPN) attachment circuits (the circuits between the provider edge [PE] and customer edge [CE] devices), use the **debug l2vpn acircuit** command in privileged EXEC mode. To disable debugging, use the **no** form of this command.

debug l2vpn acircuit {**error**| **event**| **event-trace** *number* [**preserve**]}

no debug l2vpn acircuit {**error**| **event**| **event-trace** *number* [**preserve**]}

Syntax Description

error	Displays errors that occur in attachment circuits.
event	Displays events that occur in attachment circuits.
event-trace	Displays event trace logs.
number	Number of event trace logs to be stored per context.
preserve	Specifies that the event trace logs should not be removed.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS XE Release 3.7S	This command was introduced as part of the Multiprotocol Label Switching (MPLS)-based L2VPN command modifications for cross-OS support. This command will replace the **debug acircuit** command in future releases.
15.3(1)S	This command was integrated in Cisco IOS Release 15.3(1)S.

Usage Guidelines

Use the **debug l2vpn acircuit** command to identify provisioning events, setup failures, circuit up and down events, and configuration failures on attachment circuits.

An attachment circuit connects a PE device to a CE device. A device can have many attachment circuits. The attachment circuit manager controls all attachment circuits from one central location. Therefore, when you enable debug messages for an attachment circuit, you receive information about all attachment circuits.

Examples

The following is sample output from the **debug acircuit event** command when an interface is enabled:

```
Device# debug l2vpn acircuit event
```

```
*Jan 28 15:19:03.070: ACLIB: ac_cstate() Handling circuit UP for interface Se2/0
```

```
*Jan 28 15:19:03.070: ACLIB [10.0.1.1, 200]: pthru_intf_handle_circuit_up() calling
acmgr_circuit_up
*Jan 28 15:19:03.070: ACLIB [10.0.1.1, 200]: Setting new AC state to Ac-Connecting
*Jan 28 15:19:03.070: ACMGR: Receive <Circuit Up> msg
*Jan 28 15:19:03.070: Se2/0 ACMGR: circuit up event, SIP state chg down to connecting,
action is service request
*Jan 28 15:19:03.070: Se2/0 ACMGR: Sent a sip service request
*Jan 28 15:19:03.070: ACLIB [10.0.1.1, 200]: AC updating switch context.
*Jan 28 15:19:03.070: Se2/0 ACMGR: Rcv SIP msg: resp connect forwarded, hdl 9500001D,
l2ss_hdl 700001E
*Jan 28 15:19:03.070: Se2/0 ACMGR: service connected event, SIP state chg connecting to
connected, action is respond forwarded
*Jan 28 15:19:03.070: ACLIB: pthru_intf_response hdl is 9500001D, response is 1
*Jan 28 15:19:03.070: ACLIB [10.0.1.1, 200]: Setting new AC state to Ac-Connected
```

The following is sample output from the **debug l2vpn acircuit event** command when an interface is disabled:

```
Device# debug l2vpn acircuit event

*Jan 28 15:25:57.014: ACLIB: SW AC interface INTF-DOWN for interface Se2/0
*Jan 28 15:25:57.014: ACLIB [10.0.1.1, 200]: Setting new AC state to Ac-Idle
*Jan 28 15:25:57.014: ACLIB: SW AC interface INTF-DOWN for interface Se2/0
*Jan 28 15:25:57.014: Se2/0 ACMGR: Receive <Circuit Down> msg
*Jan 28 15:25:57.014: Se2/0 ACMGR: circuit down event, SIP state chg connected to end,
action is service disconnect
*Jan 28 15:25:57.014: Se2/0 ACMGR: Sent a sip service disconnect
*Jan 28 15:25:57.014: ACLIB [10.0.1.1, 200]: AC deleting switch context.
*Jan 28 15:25:59.014: %LINK-5-CHANGED: Interface Serial2/0, changed state to
administratively down
*Jan 28 15:25:59.014: ACLIB: ac_cstate() Handling circuit DOWN for interface Se2/0
*Jan 28 15:26:00.014:%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial2/0, changed
state to down
```

The following example shows output from the **debug l2vpn acircuit** command for an xconnect session on a GigabitEthernet interface:

```
Device# debug l2vpn acircuit

23:28:35: ACLIB [10.0.3.201, 5]: SW AC interface UP for GigabitEthernet interface GE2/1/1
23:28:35: ACLIB [10.0.3.201, 5]: pthru_intf_handle_circuit_up() calling acmgr_circuit_up
23:28:35: ACLIB [10.0.3.201, 5]: Setting new AC state to Ac-Connecting
23:28:35: ACLIB [10.0.3.201, 5]: SW AC interface UP for GigabitEthernet interface GE2/1/1
23:28:35: ACLIB [10.0.3.201, 5]: pthru_intf_handle_circuit_up() ignoring up event. Already
 connected or connecting.
23:28:35: ACMGR: Receive <Circuit Up> msg
23:28:35: GE2/1/1 ACMGR: circuit up event, SIP state chg down to connecting, action is
service request
23:28:35: GE2/1/1 ACMGR: Sent a sip service request
23:28:37: %LINK-3-UPDOWN: Interface GigabitEthernet2/1/1, changed state to up
23:28:38: %LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet2/1/1, changed
state to up
23:28:53: GE2/1/1 ACMGR: Rcv SIP msg: resp connect forwarded, hdl D6000002, sss_hdl 9E00000F

23:28:53: GE2/1/1 ACMGR: service connected event, SIP state chg connecting to connected,
action is respond forwarded
23:28:53: ACLIB: pthru_intf_response hdl is D6000002, response is 1
23:28:53: ACLIB [10.0.3.201, 5]: Setting new AC state to Ac-Connected
```

The command output is self-explanatory.

Related Commands

Command	Description
debug acircuit	Debugs errors and events that occur on the attachment circuits.

Command	Description
debug vpdn	Debugs errors and events relating to L2TP configuration and the surrounding Layer 2 tunneling infrastructure.
debug xconnect	Debugs errors and events related to an xconnect configuration.

debug l2vpn atom checkpoint

To enable the debugging of Any Transport over MPLS (AToM) events when AToM is configured for nonstop forwarding/stateful switchover (NSF/SSO) and graceful restart, use the **debug l2vpn atom checkpoint** command in privileged EXEC mode. To disable the debugging of these messages, use the **no** form of this command.

debug l2vpn atom checkpoint

no debug l2vpn atom checkpoint

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging of the AToM NSF/SSO and graceful restart is disabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS XE Release 3.7S	This command was introduced as part of the Multiprotocol Label Switching (MPLS)-based Layer 2 VPN (L2VPN) command modifications for cross-OS support. This command will replace the **debug mpls l2transport checkpoint** command in future releases.
15.3(1)S	This command was integrated in Cisco IOS Release 15.3(1)S.

Usage Guidelines

Debug commands use a significant amount of CPU time and can affect system performance.

Examples

In the following example, the output shows that NSF/SSO and graceful restart synchronize the data between the active and backup Route Processors (RP) after an AToM virtual circuit (VC) is created. (Both **debug l2vpn atom checkpoint** and **debug l2vpn acircuit checkpoint** commands are enabled in this example.)

The **debug l2vpn atom checkpoint** command is enabled on the active RP:

```
Device# debug l2vpn atom checkpoint
Device# debug l2vpn acircuit checkpoint

AToM HA:
  AToM checkpointing events and errors debugging is on
AC HA:
  Attachment Circuit Checkpoint debugging is on

AToM HA [10.55.55.2, 1002]: Build provision msg, SSM sw/seg 8192/8194 [0x2000/0x2002] PW
id 9216 [0x2400] local label 21
AC HA: Dynamic Sync. Event:4 Sw:8192[2000] Se:16385[4001]
AToM HA: CF sync send complete
AC HA CF: Sync send complete. Code:0
```

On the standby RP, the following messages indicate that it receives checkpointing data:

```
AC HA [10.55.55.2, 1002]: Add to WaitQ. Flags:1
AToM HA [105.55.55.2, 1002]: Received 32-byte provision version 1 CF message
AC HA CF: ClientId:89, Entity:0 Length:40
AToM HA [10.55.55.2, 1002]: Process chkpt msg provision [1], ver 1
AToM HA [10.55.55.2, 1002]: Reserved SSM sw/seg 8192/8194 [0x2000/0x2002] PW id 9216 [0x2400]
AC HA: Process Msg:35586. Ptr:44CBFD90. Val:0
AC HA: Sync. Event:4 CktType:4 Sw:8192[2000] Se:16385[4001]
AC HA [10.55.55.2, 1002]: Remove from WaitQ. Flags:1[OK][OK]
```

During a switchover from the active to the backup RP, the following debug messages are displayed:

```
%HA-5-MODE: Operating mode is hsa, configured mode is sso.
AC HA RF: CId:83, Seq:710, Sta:RF_STATUS_OPER_REDUNDANCY_MODE_CHANGE, Opr:5, St:STANDBY
HOT, PSt:ACTIVE
AToM HA: CID 84, Seq 715, Status RF_STATUS_OPER_REDUNDANCY_MODE_CHANGE, Op 5, State STANDBY
 HOT, Peer ACTIVE
AC HA RF: CId:83, Seq:710, Sta:RF_STATUS_PEER_PRESENCE, Opr:0, St:STANDBY HOT, PSt:ACTIVE
AToM HA: CID 84, Seq 715, Status RF_STATUS_PEER_PRESENCE, Op 0, State STANDBY HOT, Peer
ACTIVE
AC HA RF: CId:83, Seq:710, Sta:RF_STATUS_PEER_COMM, Opr:0, St:STANDBY HOT, PSt:DISABLED
AToM HA: CID 84, Seq 715, Status RF_STATUS_PEER_COMM, Op 0, State STANDBY HOT, Peer DISABLED
%HA-2-CUTOVER_NOTICE: Cutover initiated. Cease all console activity until system restarts.
%HA-2-CUTOVER_NOTICE: Do not add/remove RSPs or line cards until switchover completes.
%HA-2-CUTOVER_NOTICE: Deinitializing subsystems...
%OIR-6-REMCARD: Card removed from slot 4, interfaces disabled
%OIR-6-REMCARD: Card removed from slot 5, interfaces disabled
%OIR-6-REMCARD: Card removed from slot 9, interfaces disabled
%HA-2-CUTOVER_NOTICE: Reinitializing subsystems...
%HA-2-CUTOVER_NOTICE: System preparing to restart...
%HA-5-NOTICE: Resuming initialization...
AC HA RF: CId:83, Seq:710, Sta:RF_STATUS_REDUNDANCY_MODE_CHANGE, Opr:7, St:STANDBY HOT,
PSt:DISABLED
.
.
.
%LDP-5-GR: LDP restarting gracefully.  Preserving forwarding state for 250 seconds.
AC HA RF: CId:83, Seq:710, Sta:RF_PROG_ACTIVE, Opr:0, St:ACTIVE, PSt:DISABLED
AToM HA: CID 84, Seq 715, Event RF_PROG_ACTIVE, Op 0, State ACTIVE, Peer DISABLED
AC HA: Process Msg:35588. Ptr:0. Val:0
AC HA: Switchover: Standby->Active
AC HA RF: Reconciling
```

Related Commands

Command	Description
debug l2vpn acircuit checkpoint	Enables the debugging of AToM attachment circuit events when AToM is configured for NSF/SSO and graceful restart.
debug mpls l2transport checkpoint	Enables the debugging of AToM events when AToM is configured for NSF/SSO and graceful restart.

debug l2vpn atom event-trace

To enable debugging of event trace information for Layer 2 VPN (L2VPN) Any Transport over MPLS (AToM), use the **debugl2vpn atom event-trace** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug l2vpn atom {**event-trace** *number* [**preserve**]}

no debug l2vpn atom {**event-trace** *number* [**preserve**]}

Syntax Description

number	Number of event trace logs to be stored per context.
preserve	(Optional) Specifies that the event trace logs should not be removed.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS XE Release 3.7S	This command was introduced as part of the Multiprotocol Label Switching (MPLS)-based L2VPN command modifications for cross-OS support. This command will replace the **debug mpls l2transport event-trace** command in future releases.
15.3(1)S	This command was integrated in Cisco IOS Release 15.3(1)S.

Usage Guidelines

The **debug l2vpn atom event-trace** command does not produce any output of its own. Instead, it affects the size of the event-trace buffer for Any Transport over MPLS (AToM) events.

Examples

The following is sample output from the **debug l2vpn atom event-trace** command:

```
Device# debug l2vpn atom event-trace
AToM LDP event-trace debugging is on
```

Related Commands

Command	Description
debug mpls l2transport event-trace	Enables debugging of event trace information for MPLS Layer 2 transport events.

debug l2vpn atom fast-failure-detect

To enable the debugging of Layer 2 VPN (L2VPN) Any Transport over MPLS (AToM) fast failure detection, use the **debug l2vpn atom fast-failure-detect** command in privileged EXEC mode. To disable the debugging, use the **no** form of this command.

debug l2vpn atom fast-failure-detect

no debug l2vpn atom fast-failure-detect

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging of L2VPN fast failure detection is disabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS XE Release 3.7S	This command was introduced as part of the Multiprotocol Label Switching (MPLS)-based L2VPN command modifications for cross-OS support. This command will replace the **debug mpls l2transport fast-failure-detect** command in future releases.
15.3(1)S	This command was integrated in Cisco IOS Release 15.3(1)S.

Examples

The following example shows how to enable L2VPN AToM fast failure detection:

```
Device# debug l2vpn atom fast-failure-detect

========= Line Card (Slot 3) =========
AToM fast failure detect debugging is on

00:03:28: AToM FFD[10.1.1.2]: Sending type: BFD, adjacency: DOWN, local: 10.1.1.1
00:03:28: AToM FFD[10.1.1.2]: ADJ_DOWN, local: 10.1.1.1
00:03:28: AToM FFD[10.1.1.2, 100]: ADJ_DOWN
```

Related Commands

Command	Description
debug mpls l2transport fast-failure-detection	Enables the debugging of fast failure detection.

debug l2vpn atom signaling

To enable debugging of Layer 2 VPN (L2VPN) Any Transport over MPLS (AToM) signaling protocol information, use the **debug l2vpn atom signaling** command in privileged EXEC mode. To disable debugging, use the **no** form of this command.

debug l2vpn atom signaling {**event**| **message**| **fsm**}

no debug l2vpn atom signaling {**event**| **message**| **fsm**}

Syntax Description

event	Enables debugging of protocol events.
message	Enables debugging of protocol messages.
fsm	Enables debugging of finite state machine (FSM).

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS XE Release 3.7S	This command was introduced as part of the Multiprotocol Label Switching (MPLS)-based L2VPN command modifications for cross-OS support. This command will replace the **debug mpls l2transport signaling** command in future releases.
15.3(1)S	This command was integrated in Cisco IOS Release 15.3(1)S.

Examples

The following is sample output from the **debug l2vpn atom signaling** command:

```
Device# debug l2vpn atom signaling event
AToM LDP event debugging is on

Device# debug l2vpn atom signaling message
AToM LDP message debugging is on

AToM:
  AToM LDP event debugging is on
  AToM LDP message debugging is on
*Mar 24 23:10:55.611: AToM LDP [10.9.9.9]: Allocate LDP instance
*Mar 24 23:10:55.611: AToM LDP [10.9.9.9]: Opening session, 1 clients
*Mar 24 23:10:56.063: %SYS-5-CONFIG_I: Configured from console by console
*Mar 24 23:10:56.583: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial3/0, changed
state to up
*Mar 24 23:11:00.539: AToM LDP [10.9.9.9]: Session is up
*Mar 24 23:11:00.539: AToM LDP [10.9.9.9]: Peer address change, add 10.1.1.100
*Mar 24 23:11:00.539: AToM LDP [10.9.9.9]: Peer address change, add 10.1.1.6
*Mar 24 23:11:00.539: AToM LDP [10.9.9.9]: Peer address change, add 10.9.9.9
*Mar 24 23:11:00.539: AToM LDP [10.9.9.9]: Peer address change, add 10.1.1.6
```

```
*Mar 24 23:11:00.539: AToM LDP [10.9.9.9]: Sending label mapping msg
vc type 7, cbit 1, vc id 50, group id 6, vc label 21, status 0, mtu 1500
*Mar 24 23:11:00.539: AToM LDP [10.9.9.9]: Received label mapping msg, id 113
vc type 7, cbit 1, vc id 50, group id 6, vc label 21, status 0, mtu 1500
```

Related Commands

Command	Description
debug mpls l2transport signaling	Displays information about the AToM signaling protocol.

debug l2vpn atom static-oam

To enable the debugging of messages related to static operations administrative and management (OAM), use the **debug l2vpn atom static-oam** command in privileged EXEC mode. To disable the debugging of these messages, use the **no** form of this command.

debug l2vpn atom static-oam {**elog**| **error**| **event**| **fsm**}

no debug l2vpn atom static-oam [**elog**| **error**| **event**| **fsm**]

Syntax Description

elog	Displays logging messages for static pseudowire OAM.
error	Displays error messages for static pseudowire OAM.
event	Displays event messages for static pseudowire OAM.
fsm	Displays finite state machine (FSM) messages for static pseudowire OAM.

Command Default

Display of static pseudowire messages is not disabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS XE Release 3.7S	This command was introduced as part of the Multiprotocol Label Switching (MPLS)-based Layer 2 VPN (L2VPN) command modifications for cross-OS support. This command will replace the **debug mpls l2transport static-oam** command in future releases.
15.3(1)S	This command was integrated in Cisco IOS Release 15.3(1)S.

Usage Guidelines

The **debug l2vpn atom static-oam error** does not produce any output of its own. Instead, it affects the size of the event-trace buffer for Any Transport over MPLS (AToM) events.

Examples

The following example enables the display of error messages for static pseudowire OAM:

```
Device# debug l2vpn atom static-oam error
Static PW OAM events debugging is on
```

Related Commands

Command	Description
debug mpls l2transport static-oam	Enables the debugging of messages related to static pseudowire operations OAM.
show l2vpn atom static-oam	Displays the status of static pseudowires.

debug l2vpn atom vc

To enable debugging of status of the Layer 2 VPN (L2VPN) Any Transport over MPLS (AToM) virtual circuits (VCs), use the **debug l2vpn atom vc** command in privileged EXEC mode. To disable debugging, use the **no** form of this command.

debug l2vpn atom vc {**event**| **fsm**| **ldp**| **subscriber**| **status** {**event**| **fsm**}}

no debug mpls l2transport vc {**event**| **fsm**| **ldp**| **subscriber**| **status** {**event**| **fsm**}}

Syntax Description

event	Displays AToM event messages about VCs.
fsm	Displays debug information related to the finite state machine (FSM).
ldp	Displays debug information related to the Label Distribution Protocol (LDP).
subscriber	Displays debug information related to the L2VPN subscriber.
status	Displays debug information related to the status of VCs.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS XE Release 3.7S	This command was introduced as part of the Multiprotocol Label Switching (MPLS)-based L2VPN command modifications for cross-OS support. This command will replace the **debug mpls l2transport vc** command in future releases.
15.3(1)S	This command was integrated in Cisco IOS Release 15.3(1)S.

Usage Guidelines

You can issue this command from the line card or the Route Processor (RP).

Examples

The following is sample output from the **debug l2vpn atom vc event** and **debug l2vpn atom vc fsm** commands:

```
Device# debug l2vpn atom vc event
AToM vc event debugging is on

Device# debug l2vpn atom vc fsm
```

```
AToM vc fsm debugging is on

AToM:
  AToM vc event debugging is on
  AToM vc fsm debugging is on
*Mar 24 23:17:24.371: AToM MGR [10.9.9.9, 50]: Event provision, state changed from idle to
 provisioned
*Mar 24 23:17:24.371: AToM MGR [10.9.9.9, 50]: Provision vc
*Mar 24 23:17:24.371: AToM SMGR [10.9.9.9, 50]: Requesting VC create, vc_handle 61A09930
*Mar 24 23:17:24.371: AToM MGR [10.9.9.9, 50]: Event local up, state changed from provisioned
 to local standby
*Mar 24 23:17:24.371: AToM MGR [10.9.9.9, 50]: Update local vc label binding
*Mar 24 23:17:24.371: AToM SMGR [10.9.9.9, 50]: successfully processed create request
*Mar 24 23:17:24.875: %SYS-5-CONFIG_I: Configured from console by console
*Mar 24 23:17:25.131: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial3/0, changed
state to up
*Mar 24 23:17:28.567: AToM MGR [10.9.9.9, 50]: Event ldp up, state changed from local standby
 to local ready
*Mar 24 23:17:28.567: AToM MGR [10.9.9.9, 50]: Advertise local vc label binding
*Mar 24 23:17:28.567: AToM MGR [10.9.9.9, 50]: Event remote up, state changed from local
ready to establishing
*Mar 24 23:17:28.567: AToM MGR [10.9.9.9, 50]: Remote end up
*Mar 24 23:17:28.567: AToM MGR [10.9.9.9, 50]: Event remote validated, state changed from
establishing to established
*Mar 24 23:17:28.567: AToM MGR [10.9.9.9, 50]: Validate vc, activating data plane
*Mar 24 23:17:28.567: AToM SMGR [10.9.9.9, 50]: Processing imposition update, vc_handle
61A09930, update_action 3, remote_vc_label 21
*Mar 24 23:17:28.567: AToM SMGR [10.9.9.9, 50]: Imposition Programmed, Output Interface:
PO5/0
*Mar 24 23:17:28.567: AToM SMGR [10.9.9.9, 50]: Processing disposition update, vc_handle
61A09930, update_action 3, local_vc_label 22
*Mar 24 23:17:28.571: AToM SMGR: Processing TFIB event for 10.9.9.9
*Mar 24 23:17:28.571: AToM SMGR [10.9.9.9, 50]: Imposition Programmed, Output Interface:
PO5/0
```

The following is sample output of MPLS pseudowire status signaling messages from the **debug l2vpn atom vc status event** and **debug l2vpn atom vc status fsm** commands:

```
Device# debug l2vpn atom vc status event
Device# debug l2vpn atom vc status fsm

*Feb 26 14:03:42.543: AToM MGR [10.9.9.9, 100]: Receive SSS STATUS(UP)
*Feb 26 14:03:42.543: AToM MGR [10.9.9.9, 100]: AC status UP
*Feb 26 14:03:42.543: AToM MGR [10.9.9.9, 100]: S:Evt local up, LndRru->LnuRru
*Feb 26 14:03:42.543: AToM MGR [10.9.9.9, 100]: S:Evt local ready, LnuRru->LruRru
*Feb 26 14:03:42.543: AToM MGR [10.9.9.9, 100]: S:Act send label(UP)
*Feb 26 14:03:42.543: AToM MGR [10.9.9.9, 100]: Send label(UP)
*Feb 26 14:03:42.543: AToM MGR [10.9.9.9, 100]: Local AC : UP
*Feb 26 14:03:42.543: AToM MGR [10.9.9.9, 100]: Dataplane: no fault
*Feb 26 14:03:42.543: AToM MGR [10.9.9.9, 100]: Overall  : no fault
*Feb 26 14:03:42.543: AToM MGR [10.9.9.9, 100]: Remote label is ready
*Feb 26 14:03:42.543: AToM MGR [10.9.9.9, 100]: S:Evt remote ready in LruRru
*Feb 26 14:03:42.543: AToM MGR [10.9.9.9, 100]: S:Evt remote up in LruRru
*Feb 26 14:03:42.543: AToM MGR [10.9.9.9, 100]: S:Evt dataplane clear fault in LruRru
*Feb 26 14:03:42.543: AToM MGR [10.9.9.9, 100]: S:Evt dataplane clear fault in LruRru
*Feb 26 14:03:42.551: AToM MGR [10.9.9.9, 100]: S:Evt dataplane clear fault in LruRru
```

The status codes in the messages, such as S and LruRru, indicate the status of the local and remote devices. The following is the list status codes displayed in the output:

- L—local router
- R—remote router
- r or n—ready (r) or not ready (n)
- u or d—up (u) or down (d) status

The output also includes the following values:

- D—Dataplane
- S—Local shutdown

Related Commands

Command	Description
debug mpls l2transport vc	Enables debugging of the AToM VCs.

debug l2vpn atom vc vccv

To enable Layer 2 VPN (L2VPN) Any Transport over MPLS (AToM) Virtual Circuit Connection Verification (VCCV) debugging, use the **debug l2vpn atom vc vccv** command in privileged EXEC mode. To disable debugging, use the **no** form of this command.

debug l2vpn atom vc vccv [bfd] event

no debug l2vpn atom vc vccv [bfd] event

Syntax Description

bfd	(Optional) Displays event messages when Bidirectional Forwarding Detection (BFD) sessions are created, when BFD sends dataplane fault notifications to L2VPN, and when L2VPN sends the attachment circuit (AC) signaling status to BFD.
event	Displays AToM event messages about the VCCV.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS XE Release 3.7S	This command was introduced as part of the Multiprotocol Label Switching (MPLS)-based L2VPN command modifications for cross-OS support. This command will replace the **debug mpls l2transport vc vccv** command in future releases.
15.3(1)S	This command was integrated in Cisco IOS Release 15.3(1)S.

Usage Guidelines

Use this command to enable L2VPN AToM VCCV events and AToM VCCV BFD events debugging.

Examples

The following example shows how to enable MPLS L2VPN virtual circuit (VC) VCCV BFD event debugging:

```
Device# debug l2vpn atom vc vccv bfd event
AToM VCCV BFD events debugging is on

Aug 10 16:55:41.492: AToM VCCV BFD[10.1.1.2, 1234000]: ......... context not found
*Aug 10 16:55:41.492: AToM VCCV BFD[10.1.1.2, 1234000]: ......... context not found
*Aug 10 16:55:41.492: AToM VCCV BFD[10.1.1.2, 1234000]: ............ context not found
*Aug 10 16:55:41.492: AToM VCCV BFD[10.1.1.2, 1234000]: ....... context not found
*Aug 10 16:55:41.493: AToM VCCV BFD[10.1.1.2, 1234000]: .. Session create
*Aug 10 16:55:41.493: AToM VCCV BFD[10.1.1.2, 1234000]: ..  next-hop        2.1.1.2:1
*Aug 10 16:55:41.493: AToM VCCV BFD[10.1.1.2, 1234000]: ..  cc_type         1
*Aug 10 16:55:41.493: AToM VCCV BFD[10.1.1.2, 1234000]: ..  cv_type         5
*Aug 10 16:55:41.493: AToM VCCV BFD[10.1.1.2, 1234000]: ..  CC control word  enabled
```

```
*Aug 10 16:55:41.493: AToM VCCV BFD[10.1.1.2, 1234000]: ..  CV Fault Detection and Signaling
 without IP/UDP headers
*Aug 10 16:55:41.500: AToM VCCV BFD[10.1.1.2, 1234000]: .. create 00000001/2A98A72F40
*Aug 10 16:55:41.500: AToM VCCV BFD[10.1.1.2, 1234000]: .. lookup added 00000001
*Aug 10 16:55:42.315: AToM VCCV BFD[10.1.1.2, 1234000]: session 00000001 ADJ UP
*Aug 10 16:55:42.315: AToM VCCV BFD[10.1.1.2, 1234000]: inform BFD, status UP, event 1
*Aug 10 16:55:42.315: AToM VCCV BFD[10.1.1.2, 1234000]: Start VCCV BFD status timer
*Aug 10 16:55:45.374: AToM VCCV BFD[10.1.1.2, 1234000]: VCCV BFD status timer expired
*Aug 10 16:55:45.374: AToM VCCV BFD[10.1.1.2, 1234000]: session 00000001 BFD STATUS UP
```

Related Commands

Command	Description
debug mpls l2transport vc vccv	Enables AToM VCCV debugging
show mpls l2transport vc	Displays information about the status of the AToM VCs.

debug l2vpn pseudowire

To enable debugging information for Layer 2 VPN (L2VPN) pseudowire configuration, use the **debug l2vpn pseudowire** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug l2vpn pseudowire {**event**| **error**}

no debug l2vpn pseudowire {**event**| **error**}

Syntax Description

event	Displays debugging information for L2VPN pseudowire events.
error	Displays debugging information for L2VPN pseudowire errors.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS XE Release 3.7S	This command was introduced as part of the Multiprotocol Label Switching (MPLS)-based L2VPN command modifications for cross-OS support.
15.3(1)S	This command was integrated into Cisco IOS release 15.3(1)S.

Examples

The following is sample output from the **debug l2vpn pseudowire event** command:

```
Device# debug l2vpn pseudowire event
L2VPN pseudowire events debugging is on.

*Aug 10 17:52:22.896: Pseudowire[ps1]: PW interface not yet in a L2VPN service, ignore
[no]shutdown
*Aug 10 17:52:25.851: Pseudowire[pw1]: Pseudowire interface: peer id 10.0.0.0 not configured

*Aug 10 17:52:25.851: Pseudowire[pw1]: Pseudowire interface config still incomplete, skip
update to xconnect db
*Aug 10 17:52:33.727: PWCFG WAVL Event: Updating pwid: 10002 peer: 10.1.1.2 vcid: 1234000
*Aug 10 17:52:33.727: PWCFG WAVL Event: pwid: 10002 alloc peer 10.1.1.2 vcid: 1234000
*Aug 10 17:52:33.727: Pseudowire[pw1]: Pseudowire interface not yet associated with a L2VPN
 service
```

debug l2vpn vfi

To enable debugging layer 2 VPN (L2VPN) virtual forwarding instance (VFI) events and errors, use the **debug l2vpn vfi** command in privileged EXEC mode. To disable debugging of VFI events and errors, use the **no** form of this command.

debug l2vpn vfi [**fsm**] {**error**| **event**}

no debug l2vpn vfi [**fsm**] {**error**| **event**}

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS XE Release 3.7S	This command was introduced. This command will replace the **debug vfi** command in future releases.
15.3(1)S	This command was integrated in Cisco IOS Release 15.3(1)S.

Examples

The following is sample output from the **debug l2vpn vfi** command:

```
Device# debug l2vpn vfi
```

Related Commands

Command	Description
debug vfi	Enables debugging VFI events and errors.

debug l2vpn xconnect

To enable the debugging information about a Layer 2 VPN (L2VPN) xconnect configuration, use the **debug l2vpn xconnect** command in privileged EXEC mode. To disable debugging, use the **no** form of this command.

debug l2vpn xconnect {**error**| **event** [**detail**]| **initialization**| **internal**| **monitor**}

no debug l2vpn xconnect {**error**| **event** [**detail**]| **initialization**| **internal**| **monitor**}

Syntax Description

error	Displays errors related to an xconnect configuration.
event	Displays events related to an xconnect configuration.
detail	(Optional) Displays the xconnect detailed debugging information.
initialization	Displays information about xconnect initialization events.
internal	Displays information about xconnect internal events.
monitor	Displays debugging information about xconnect peer monitoring debugs.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS XE Release 3.7S	This command was introduced as part of the Multiprotocol Label Switching (MPLS)-based L2VPN command modifications for cross-OS support. This command will replace the **debug xconnect** command in future releases.
15.3(1)S	This command was integrated in Cisco IOS Release 15.3(1)S.

Examples

The following is sample output from the **debug l2vpn xconnect** command for an xconnect session on a Gigabit Ethernet interface:

```
Device# debug l2vpn xconnect event
```

```
00:01:16: XC AUTH [Gi2/1/1, 5]: Event: start xconnect authorization, state changed from
IDLE to AUTHORIZING
00:01:16: XC AUTH [Gi2/1/1, 5]: Event: found xconnect authorization, state changed from
AUTHORIZING to DONE
00:01:16: XC AUTH [Gi2/1/1, 5]: Event: free xconnect authorization request, state changed
from DONE to END
```

Related Commands

Command	Description
debug xconnect	Enables the debugging information for an xconnect configuration.

debug l3-mgr tunnel

To enable debugging for interface, tunnel, or VLAN events for the Layer 3 manager infrastructure on RP of Cisco 7600 routers, use the **debug l3-mgr tunnel** command. To disable the debugging, use the **no** form of the command.

debug l3-mgr tunnel

no debug l3-mgr tunnel

Syntax Description

l3-mgr	Displays debugging output for the Layer 3 manager infrastructure on Cisco 7600 routers.
tunnel	Displays all tunnel related reserved VLAN events.

Command Default

None

Command Modes

Privileged EXEC

Command History

Release	Modification
15.3(2)S	This command was introduced on Cisco 7600 series routers.

Usage Guidelines

Use the debug command only to troubleshoot specific problems, or during troubleshooting sessions with Cisco technical support staff.

Examples

The following shows sample output for the **debug l3-mgr tunnel** command:

```
CE1#debug l3-mgr tunnel
l3 mgr tunnel debugging is on
*Mar  1 09:50:53.431 IST: l3mgr_tunnel_checking_src_address:
Tunnel[Tunnel56] src[64003801] tbl_id[0] state changed to DOWN
*Mar  1 09:50:53.431 IST: l3mgr_tunnel_checking_src_address:
Checked Tunnel[Tunnel110] src[64006E01] if_up[UP] tbl_id[0]
*Mar  1 09:50:53.431 IST: l3mgr_tunnel_checking_src_address:
Checked Tunnel[Tunnel109] src[64006D01] if_up[UP] tbl_id[0]
*Mar  1 09:50:53.431 IST: l3mgr_tunnel_checking_src_address:
Checked Tunnel[Tunnel108] src[64006C01] if_up[UP] tbl_id[0]
*Mar  1 09:50:53.4no sh31 IST: l3mgr_tunnel_checking_src_address:
Checked Tunnel[Tunnel107] src[64006B01] if_up[UP] tbl_id[0]
*Mar  1 09:50:53.431 IST: l3mgr_tunnel_checking_src_address:
Checked Tunnel[Tunnel106] src[64006A01] if_up[UP] tbl_id[0]
*Mar  1 09:50:53.431 IST: l3mgr_tunnel_checking_src_address:
Checked Tunnel[Tunnel105] src[64006901] if_up[UP] tbl_id[0]
*Mar  1 09:50:53.431 IST: l3mgr_tunnel_checking_src_address:
Checked Tunnel[Tunnel104] src[64006801] if_up[UP] tbl_id[0]
```

```
*Mar  1 09:50:53.431 IST: l3mgr_tunnel_checking_src_address:
Checked Tunnel[Tunnel103] src[64006701] if_up[UP] tbl_id[0]
*Mar  1 09:50:53.431 IST: l3mgr_tunnel_checking_src_address:
Checked Tunnel[Tunnel102] src[64006601] if_up[UP] tbl_id[0]
*Mar  1 09:50:53.435 IST: l3mgr_tunnel_checking_src_address:
Checked Tunnel[Tunnel101] src[64006501] if_up[UP] tbl_id[0]
*Mar  1 09:50:53.435 IST: l3mgr_tunnel_checking_src_address:
Checked Tunnel[Tunnel100] src[64006401] if_up[UP] tbl_id[0]
```

debug l4f

To enable troubleshooting for Layer 4 Forwarding (L4F) flows, use the **debug l4f** command in privileged EXEC mode. To disable the troubleshooting, use the **no** form of this command.

debug l4f {**api**| **flow-db**| **flows**| **packet** {**all**| **detail**| **injection**| **interception**| **proxying**| **spoofing**}| **test-app**| **trace-db-api**| **trace-db-flow**| **trace-engine**}

no debug l4f {**api**| **flow-db**| **flows**| **packet** {**all**| **detail**| **injection**| **interception**| **proxying**| **spoofing**}| **test-app**| **trace-db-api**| **trace-db-flow**| **trace-engine**}

Syntax Description

api	Toggles L4F API debugging.
flow-db	Toggles L4F flow database debugging.
flows	Toggles L4F flows debugging.
packet	Toggles L4F packet debugging.
all	Toggles all L4F packet debugging.
detail	Toggles L4F packet detail debugging.
injection	Toggles L4F packet injection debugging.
interception	Toggles L4F packet interception debugging.
proxying	Toggles L4F packet proxying debugging.
spoofing	Toggles L4F packet spoofing debugging.
test-app	Toggles L4F test application debugging.
trace-db-api	Toggles L4F database API debugging.
trace-db-flow	Toggles L4F database flow debugging.
trace-engine	Toggles L4F API tracing debugging.

Command Default

L4F debugging is off.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
15.1(2)T	This command was introduced.

Usage Guidelines

Use this command to enable debugging for Layer 4 forwarding flows.

Examples

The following example shows how to enable debugging for L4F packets:

```
Router# debug l4f packet all
```

Related Commands

Command	Description
show l4f	Displays the flow database for L4F.

debug lacp

To enable debugging of all Link Aggregation Control Protocol (LACP) activity, use the **debug lacp**command in privileged EXEC mode. To disable LACP debugging, use the **no** form of this command.

debug lacp [**all**| **event**| **fsm**| **misc**| **multi-chassis** [**all**| **database**| **lacp-mgr**| **redundancy-group**| **user-interface**]| **packet**]

no debug lacp [**all**| **event**| **fsm**| **misc**| **multi-chassis** [**all**| **database**| **lacp-mgr**| **redundancy-group**| **user-interface**]| **packet**]

Syntax Description

all	(Optional) Activates debugging for all LACP operations.
event	(Optional) Activates debugging of events that occur within LACP.
fsm	(Optional) Activates debugging for changes within the LACP finite state machine.
misc	(Optional) Activates debugging for various operations that may be useful for monitoring the status of LACP.
multi-chassis	(Optional) Activates multi-chassis LACP (mLACP) debugging.
all	(Optional) Activates all mLACP debugging.
database	(Optional) Activates mLACP database debugging.
lacp-mgr	(Optional) Activates mLACP interface debugging.
redundancy-group	(Optional) Activates mLACP interchassis redundancy group debugging.
user-interface	(Optional) Activates mLACP interchassis user interface debugging.
packet	(Optional) Displays the receiving and transmitting LACP control packets.

Command Default

LACP debugging activity is disabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.1(13)EW	Support for this command was introduced on the Cisco Catalyst 4500 series switch.
12.2(31)SB2	This command was integrated into Cisco IOS Release 12.2(31)SB.
12.2(33)SRB	Support for this command on the Cisco 7600 router was integrated into Cisco IOS Release 12.2(33)SRB.
Cisco IOS XE Release 2.4	This command was integrated into Cisco IOS XE Release 2.4.
12.2(33)SRE	This command was modified. The following keywords were added: **multi-chassis**, **all**, **database**, **lacp-mgr**, **redundancy-group**, and **user-interface**.

Usage Guidelines

This command is useful for troubleshooting problems with LACP.

Examples

The following sample output from the **debug lacp all** command shows LACP activity on a port-channel member link Gigabit Ethernet 5/0/0:

```
Router# debug lacp all
Link Aggregation Control Protocol all debugging is on
Router1#
*Aug 20 17:21:51.685: LACP :lacp_bugpak: Receive LACP-PDU packet via Gi5/0/0
*Aug 20 17:21:51.685: LACP : packet size: 124
*Aug 20 17:21:51.685: LACP: pdu: subtype: 1, version: 1
*Aug 20 17:21:51.685: LACP: Act: tlv:1, tlv-len:20, key:0x1, p-pri:0x8000, p:0x14,
p-state:0x3C,
s-pri:0xFFFF, s-mac:0011.2026.7300
*Aug 20 17:21:51.685: LACP: Part: tlv:2, tlv-len:20, key:0x5, p-pri:0x8000, p:0x42,
p-state:0x3D,
s-pri:0x8000, s-mac:0014.a93d.4a00
*Aug 20 17:21:51.685: LACP: col-tlv:3, col-tlv-len:16, col-max-d:0x8000
*Aug 20 17:21:51.685: LACP: term-tlv:0 termr-tlv-len:0
*Aug 20 17:21:51.685: LACP: Gi5/0/0 LACP packet received, processing
*Aug 20 17:21:51.685:     lacp_rx Gi5: during state CURRENT, got event 5(recv_lacpdu)
*Aug 20 17:21:59.869: LACP: lacp_p(Gi5/0/0) timer stopped
*Aug 20 17:21:59.869: LACP: lacp_p(Gi5/0/0) expired
*Aug 20 17:21:59.869:     lacp_ptx Gi5: during state SLOW_PERIODIC, got event 3(pt_expired)
*Aug 20 17:21:59.869: @@@ lacp_ptx Gi5: SLOW_PERIODIC -> PERIODIC_TX
*Aug 20 17:21:59.869: LACP: Gi5/0/0 lacp_action_ptx_slow_periodic_exit entered
*Aug 20 17:21:59.869: LACP: lacp_p(Gi5/0/0) timer stopped
*Aug 20 17:22:00.869: LACP: lacp_t(Gi5/0/0) timer stopped
*Aug 20 17:22:00.869: LACP: lacp_t(Gi5/0/0) expired
*Aug 20 17:22:19.089: LACP :lacp_bugpak: Receive LACP-PDU packet via Gi5/0/0
*Aug 20 17:22:19.089: LACP : packet size: 124
*Aug 20 17:22:19.089: LACP: pdu: subtype: 1, version: 1
*Aug 20 17:22:19.089: LACP: Act: tlv:1, tlv-len:20, key:0x1, p-pri:0x8000, p:0x14,
p-state:0x4,
s-pri:0xFFFF, s-mac:0011.2026.7300
*Aug 20 17:22:19.089: LACP: Part: tlv:2, tlv-len:20, key:0x5, p-pri:0x8000, p:0x42,
p-state:0x34,
s-pri:0x8000, s-mac:0014.a93d.4a00
*Aug 20 17:22:19.089: LACP: col-tlv:3, col-tlv-len:16, col-max-d:0x8000
*Aug 20 17:22:19.089: LACP: term-tlv:0 termr-tlv-len:0
*Aug 20 17:22:19.089: LACP: Gi5/0/0 LACP packet received, processing
```

```
*Aug 20 17:22:19.089:     lacp_rx Gi5: during state CURRENT, got event 5(recv_lacpdu)
*Aug 20 17:22:19.989: LACP: lacp_t(Gi5/0/0) timer stopped
*Aug 20 17:22:19.989: LACP: lacp_t(Gi5/0/0) expired
*Aug 20 17:22:19.989: LACP: timer lacp_t(Gi5/0/0) started with interval 1000.
*Aug 20 17:22:19.989: LACP: lacp_send_lacpdu: (Gi5/0/0) About to send the 110 LACPDU
*Aug 20 17:22:19.989: LACP :lacp_bugpak: Send LACP-PDU packet via Gi5/0/0
*Aug 20 17:22:19.989: LACP : packet size: 124
*Aug 20 17:22:20.957: LACP: lacp_t(Gi5/0/0) timer stopped
*Aug 20 17:22:20.957: LACP: lacp_t(Gi5/0/0) expired
*Aug 20 17:22:21.205: %LINK-3-UPDOWN: Interface GigabitEthernet5/0/0, changed state to down
*Aug 20 17:22:21.205: LACP: lacp_hw_off: Gi5/0/0 is going down
*Aug 20 17:22:21.205: LACP: if_down: Gi5/0/0
*Aug 20 17:22:21.205:     lacp_ptx Gi5: during state SLOW_PERIODIC, got event 0(no_periodic)
*Aug 20 17:22:22.089: %LINEPROTO-5-UPDOWN: Line protocol on Interface Port-channel5, changed
 state to down
*Aug 20 17:22:22.153: %C10K_ALARM-6-INFO: CLEAR CRITICAL GigE 5/0/0 Physical Port Link Down

*Aug 20 17:22:23.413: LACP: Gi5/0/0 oper-key: 0x0
*Aug 20 17:22:23.413: LACP: lacp_hw_on: Gi5/0/0 is coming up
*Aug 20 17:22:23.413:     lacp_ptx Gi5: during state NO_PERIODIC, got event 0(no_periodic)
*Aug 20 17:22:23.413: @@@ lacp_ptx Gi5: NO_PERIODIC -> NO_PERIODIC
*Aug 20 17:22:23.413: LACP: Gi5/0/0 lacp_action_ptx_no_periodic entered
*Aug 20 17:22:23.413: LACP: lacp_p(Gi5/0/0) timer stopped
*Aug 20 17:22:24.153: %LINK-3-UPDOWN: Interface GigabitEthernet5/0/0, changed state to up
*Aug 20 17:22:24.153: LACP: lacp_hw_on: Gi5/0/0 is coming up
*Aug 20 17:22:24.153:     lacp_ptx Gi5: during state FAST_PERIODIC, got event 0(no_periodic)
*Aug 20 17:22:24.153: @@@ lacp_ptx Gi5: FAST_PERIODIC -> NO_PERIODIC
*Aug 20 17:22:24.153: LACP: Gi5/0/0 lacp_action_ptx_fast_periodic_exit entered
*Aug 20 17:22:24.153: LACP: lacp_p(Gi5/0/0) timer stopped
*Aug 20 17:22:24.153: LACP:
*Aug 20 17:22:25.021: LACP: lacp_p(Gi5/0/0) timer stopped
*Aug 20 17:22:25.021: LACP: lacp_p(Gi5/0/0) expired
*Aug 20 17:22:25.021:     lacp_ptx Gi5: during state FAST_PERIODIC, got event 3(pt_expired)
*Aug 20 17:22:25.021: @@@ lacp_ptx Gi5: FAST_PERIODIC -> PERIODIC_TX
*Aug 20 17:22:25.021: LACP: Gi5/0/0 lacp_action_ptx_fast_periodic_exit entered
*Aug 20 17:22:25.021: LACP: lacp_p(Gi5/0/0) timer stopped
*Aug 20 17:22:25.917: LACP: lacp_p(Gi5/0/0) timer stopped
*Aug 20 17:22:25.917: LACP: lacp_p(Gi5/0/0) expired
*Aug 20 17:22:25.917:     lacp_ptx Gi5: during state FAST_PERIODIC, got event 3(pt_expired)
*Aug 20 17:22:25.917: @@@ lacp_ptx Gi5: FAST_PERIODIC -> PERIODIC_TX
*Aug 20 17:22:25.917: LACP: Gi5/0/0 lacp_action_ptx_fast_periodic_exit entered
*Aug 20 17:22:25.917: LACP: lacp_p(Gi5/0/0) timer stopped
Router1#
```

debug lane client

> **Note** Effective with Cisco IOS Release 15.1M, the **debug lane client** command is not available in Cisco IOS software.

To display information about a LAN Emulation Client (LEC), use the **debug lane client** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug lane client {**all**| **le-arp**| **mpoa**| **packet**| **signaling**| **state**| **topology**} [**interface** *interface*]

no debug lane client {**all**| **le-arp**| **mpoa**| **packet**| **signaling**| **state**| **topology**} [**interface** *interface*]

Syntax Description

all	Displays all debug information related to the LEC.
le-arp	Displays debug information related to the LAN Emulation (LANE) Address Resolution Protocol (ARP) table.
mpoa	Displays debug information to track the following: • MPOA specific TLV information in le-arp requests/responses • Elan-id and local segment TLV in lane control frames • When a LANE client is bound to an MPC/MPS
packet	Displays debug information about each packet.
signaling	Displays debug information related to client switched virtual circuits (SVCs).
state	Displays debug information when the state changes.
topology	Displays debug information related to the topology of the emulated LAN (ELAN).
interface *interface*	(Optional) Limits the debugging output to messages that relate to a particular interface or subinterface. If you enter this command multiple times with different interfaces, the last interface entered will be the one used to filter the messages.

Command Default

If the interface number is not specified, the default will be the number of all the **mpoa lane** clients.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.0(1)T	This command was introduced.
15.1M	This command was removed.

Usage Guidelines

The **debug lane client all** command can generate a large amount of output. Use a limiting keyword or specify a subinterface to decrease the amount of output and focus on the information you need.

Examples

The following example shows output for **debug lane client packet** and **debug lane client state** commands for an LEC joining an ELAN named elan1:

```
Router# debug lane client packet
Router# debug lane client state
```

The LEC listens for signaling calls to its ATM address (Initial State):

```
LEC ATM2/0.1: sending LISTEN
LEC ATM2/0.1:   listen on       39.020304050607080910111213.00000CA05B40.01
LEC ATM2/0.1: received LISTEN
```

The LEC calls the LAN Emulation Configuration Server (LECS) and attempts to set up the Configure Direct VC (LECS Connect Phase):

```
LEC ATM2/0.1: sending SETUP
LEC ATM2/0.1:   callid          0x6114D174
LEC ATM2/0.1:   called party    39.020304050607080910111213.00000CA05B43.00
LEC ATM2/0.1:   calling_party   39.020304050607080910111213.00000CA05B40.01
```

The LEC receives a CONNECT response from the LECS. The Configure Direct VC is established:

```
LEC ATM2/0.1: received CONNECT
LEC ATM2/0.1:   callid          0x6114D174
LEC ATM2/0.1:   vcd             148
```

The LEC sends a CONFIG REQUEST to the LECS on the Configure Direct VC (Configuration Phase):

```
LEC ATM2/0.1: sending LANE_CONFIG_REQ on VCD 148
LEC ATM2/0.1:   SRC MAC address 0000.0ca0.5b40
LEC ATM2/0.1:   SRC ATM address 39.020304050607080910111213.00000CA05B40.01
LEC ATM2/0.1:   LAN Type        2
LEC ATM2/0.1:   Frame size      2
LEC ATM2/0.1:   LAN Name        elan1
LEC ATM2/0.1:   LAN Name size   5
```

The LEC receives a CONFIG RESPONSE from the LECS on the Configure Direct VC:

```
LEC ATM2/0.1: received LANE_CONFIG_RSP on VCD 148
LEC ATM2/0.1:   SRC MAC address 0000.0ca0.5b40
LEC ATM2/0.1:   SRC ATM address 39.020304050607080910111213.00000CA05B40.01
LEC ATM2/0.1:   LAN Type        2
LEC ATM2/0.1:   Frame size      2
```

```
LEC ATM2/0.1:   LAN Name        elan1
LEC ATM2/0.1:   LAN Name size   5
```

The LEC releases the Configure Direct VC:

```
LEC ATM2/0.1: sending RELEASE
LEC ATM2/0.1:   callid          0x6114D174
LEC ATM2/0.1:   cause code      31
```

The LEC receives a RELEASE_COMPLETE from the LECS:

```
LEC ATM2/0.1: received RELEASE_COMPLETE
LEC ATM2/0.1:   callid          0x6114D174
LEC ATM2/0.1:   cause code      16
```

The LEC calls the LAN Emulation Server (LES) and attempts to set up the Control Direct VC (Join/Registration Phase):

```
LEC ATM2/0.1: sending SETUP
LEC ATM2/0.1:   callid          0x61167110
LEC ATM2/0.1:   called party    39.020304050607080910111213.00000CA05B41.01
LEC ATM2/0.1:   calling_party   39.020304050607080910111213.00000CA05B40.01
```

The LEC receives a CONNECT response from the LES. The Control Direct VC is established:

```
LEC ATM2/0.1: received CONNECT
LEC ATM2/0.1:   callid          0x61167110
LEC ATM2/0.1:   vcd             150
```

The LEC sends a JOIN REQUEST to the LES on the Control Direct VC:

```
LEC ATM2/0.1: sending LANE_JOIN_REQ on VCD 150
LEC ATM2/0.1:   Status          0
LEC ATM2/0.1:   LECID           0
LEC ATM2/0.1:   SRC MAC address 0000.0ca0.5b40
LEC ATM2/0.1:   SRC ATM address 39.020304050607080910111213.00000CA05B40.01
LEC ATM2/0.1:   LAN Type        2
LEC ATM2/0.1:   Frame size      2
LEC ATM2/0.1:   LAN Name        elan1
LEC ATM2/0.1:   LAN Name size   5
```

The LEC receives a SETUP request from the LES to set up the Control Distribute VC:

```
LEC ATM2/0.1: received SETUP
LEC ATM2/0.1:   callid          0x6114D174
LEC ATM2/0.1:   called party    39.020304050607080910111213.00000CA05B40.01
LEC ATM2/0.1:   calling_party   39.020304050607080910111213.00000CA05B41.01
```

The LEC responds to the LES call setup with a CONNECT:

```
LEC ATM2/0.1: sending CONNECT
LEC ATM2/0.1:   callid          0x6114D174
LEC ATM2/0.1:   vcd             151
```

A CONNECT_ACK is received from the ATM switch. The Control Distribute VC is established:

```
LEC ATM2/0.1: received CONNECT_ACK
The LEC receives a JOIN response from the LES on the Control Direct VC.
LEC ATM2/0.1: received LANE_JOIN_RSP on VCD 150
LEC ATM2/0.1:   Status          0
LEC ATM2/0.1:   LECID           1
LEC ATM2/0.1:   SRC MAC address 0000.0ca0.5b40
LEC ATM2/0.1:   SRC ATM address 39.020304050607080910111213.00000CA05B40.01
LEC ATM2/0.1:   LAN Type        2
LEC ATM2/0.1:   Frame size      2
LEC ATM2/0.1:   LAN Name        elan1
LEC ATM2/0.1:   LAN Name size   5
```

The LEC sends an LE ARP request to the LES to obtain the broadcast and unknown server (BUS) ATM NSAP address (BUS connect):

```
LEC ATM2/0.1: sending LANE_ARP_REQ on VCD 150
```

```
LEC ATM2/0.1:   SRC MAC address     0000.0ca0.5b40
LEC ATM2/0.1:   SRC ATM address     39.020304050607080910111213.00000CA05B40.01
LEC ATM2/0.1:   TARGET MAC address     ffff.ffff.ffff
LEC ATM2/0.1:   TARGET ATM address  00.000000000000000000000000.000000000000.00
```

The LEC receives its own LE ARP request via the LES over the Control Distribute VC:

```
LEC ATM2/0.1: received LANE_ARP_RSP on VCD 151
LEC ATM2/0.1:   SRC MAC address     0000.0ca0.5b40
LEC ATM2/0.1:   SRC ATM address     39.020304050607080910111213.00000CA05B40.01
LEC ATM2/0.1:   TARGET MAC address     ffff.ffff.ffff
LEC ATM2/0.1:   TARGET ATM address  39.020304050607080910111213.00000CA05B42.01
```

The LEC calls the BUS and attempts to set up the Multicast Send VC:

```
LEC ATM2/0.1: sending SETUP
LEC ATM2/0.1:   callid          0x6114D354
LEC ATM2/0.1:   called party    39.020304050607080910111213.00000CA05B42.01
LEC ATM2/0.1:   calling_party   39.020304050607080910111213.00000CA05B40.01
```

The LEC receives a CONNECT response from the BUS. The Multicast Send VC is established:

```
LEC ATM2/0.1: received CONNECT
LEC ATM2/0.1:   callid          0x6114D354
LEC ATM2/0.1:   vcd             153
```

The LEC receives a SETUP request from the BUS to set up the Multicast Forward VC:

```
LEC ATM2/0.1: received SETUP
LEC ATM2/0.1:   callid          0x610D4230
LEC ATM2/0.1:   called party    39.020304050607080910111213.00000CA05B40.01
LEC ATM2/0.1:   calling_party   39.020304050607080910111213.00000CA05B42.01
```

The LEC responds to the BUS call setup with a CONNECT:

```
LEC ATM2/0.1: sending CONNECT
LEC ATM2/0.1:   callid          0x610D4230
LEC ATM2/0.1:   vcd             154
```

A CONNECT_ACK is received from the ATM switch. The Multicast Forward VC is established:

```
LEC ATM2/0.1: received CONNECT_ACK
The LEC moves into the OPERATIONAL state.
%LANE-5-UPDOWN: ATM2/0.1 elan elan1: LE Client changed state to up
```

The following output is from the **show lane client** command after the LEC joins the emulated LAN as shown in the **debug lane client** output:

```
Router# show lane client
LE Client ATM2/0.1  ELAN name: elan1  Admin: up  State: operational
Client ID: 1                 LEC up for 1 minute 2 seconds
Join Attempt: 1
HW Address: 0000.0ca0.5b40   Type: token ring           Max Frame Size: 4544
Ring:1      Bridge:1        ELAN Segment ID: 2048
ATM Address: 39.020304050607080910111213.00000CA05B40.01
 VCD  rxFrames  txFrames  Type       ATM Address
   0         0         0  configure  39.020304050607080910111213.00000CA05B43.00
 142         1         2  direct     39.020304050607080910111213.00000CA05B41.01
 143         1         0  distribute 39.020304050607080910111213.00000CA05B41.01
 145         0         0  send       39.020304050607080910111213.00000CA05B42.01
 146         1         0  forward    39.020304050607080910111213.00000CA05B42.01
```

The following example shows **debug lane client** all command output when an interface with LECS, an LES/BUS, and an LEC is shut down:

```
Router# debug lane client all
LEC ATM1/0.2: received RELEASE_COMPLETE
LEC ATM1/0.2:   callid          0x60E8B474
LEC ATM1/0.2:   cause code      0
LEC ATM1/0.2: action A_PROCESS_REL_COMP
LEC ATM1/0.2: action A_TEARDOWN_LEC
LEC ATM1/0.2: sending RELEASE
```

```
LEC ATM1/0.2:   callid          0x60EB6160
LEC ATM1/0.2:   cause code      31
LEC ATM1/0.2: sending RELEASE
LEC ATM1/0.2:   callid          0x60EB7548
LEC ATM1/0.2:   cause code      31
LEC ATM1/0.2: sending RELEASE
LEC ATM1/0.2:   callid          0x60EB9E48
LEC ATM1/0.2:   cause code      31
LEC ATM1/0.2: sending CANCEL
LEC ATM1/0.2:   ATM address     47.00918100000000613E5A2F01.006070174820.02
LEC ATM1/0.2: state ACTIVE event LEC_SIG_RELEASE_COMP => TERMINATING
LEC ATM1/0.3: received RELEASE_COMPLETE
LEC ATM1/0.3:   callid          0x60E8D108
LEC ATM1/0.3:   cause code      0
LEC ATM1/0.3: action A_PROCESS_REL_COMP
LEC ATM1/0.3: action A_TEARDOWN_LEC
LEC ATM1/0.3: sending RELEASE
LEC ATM1/0.3:   callid          0x60EB66D4
LEC ATM1/0.3:   cause code      31
LEC ATM1/0.3: sending RELEASE
LEC ATM1/0.3:   callid          0x60EB7B8C
LEC ATM1/0.3:   cause code      31
LEC ATM1/0.3: sending RELEASE
LEC ATM1/0.3:   callid          0x60EBA3BC
LEC ATM1/0.3:   cause code      31
LEC ATM1/0.3: sending CANCEL
LEC ATM1/0.3:   ATM address     47.00918100000000613E5A2F01.006070174820.03
LEC ATM1/0.3: state ACTIVE event LEC_SIG_RELEASE_COMP => TERMINATING
LEC ATM1/0.2: received RELEASE_COMPLETE
LEC ATM1/0.2:   callid          0x60EB7548
LEC ATM1/0.2:   cause code      0
LEC ATM1/0.2: action A_PROCESS_TERM_REL_COMP
LEC ATM1/0.2: state TERMINATING event LEC_SIG_RELEASE_COMP => TERMINATING
LEC ATM1/0.3: received RELEASE_COMPLETE
LEC ATM1/0.3:   callid          0x60EB7B8C
LEC ATM1/0.3:   cause code      0
LEC ATM1/0.3: action A_PROCESS_TERM_REL_COMP
LEC ATM1/0.3: state TERMINATING event LEC_SIG_RELEASE_COMP => TERMINATING
LEC ATM1/0.1: received RELEASE_COMPLETE
LEC ATM1/0.1:   callid          0x60EBC458
LEC ATM1/0.1:   cause code      0
LEC ATM1/0.1: action A_PROCESS_REL_COMP
LEC ATM1/0.1: action A_TEARDOWN_LEC
LEC ATM1/0.1: sending RELEASE
LEC ATM1/0.1:   callid          0x60EBD30C
LEC ATM1/0.1:   cause code      31
LEC ATM1/0.1: sending RELEASE
LEC ATM1/0.1:   callid          0x60EBDD28
LEC ATM1/0.1:   cause code      31
LEC ATM1/0.1: sending RELEASE
LEC ATM1/0.1:   callid          0x60EBF174
LEC ATM1/0.1:   cause code      31
LEC ATM1/0.1: sending CANCEL
LEC ATM1/0.1:   ATM address     47.00918100000000613E5A2F01.006070174820.01
LEC ATM1/0.1: state ACTIVE event LEC_SIG_RELEASE_COMP => TERMINATING
LEC ATM1/0.1: received RELEASE_COMPLETE
LEC ATM1/0.1:   callid          0x60EBDD28
LEC ATM1/0.1:   cause code      0
LEC ATM1/0.1: action A_PROCESS_TERM_REL_COMP
LEC ATM1/0.1: state TERMINATING event LEC_SIG_RELEASE_COMP => TERMINATING
LEC ATM1/0.2: received RELEASE_COMPLETE
LEC ATM1/0.2:   callid          0x60EB6160
LEC ATM1/0.2:   cause code      0
LEC ATM1/0.2: action A_PROCESS_TERM_REL_COMP
LEC ATM1/0.2: state TERMINATING event LEC_SIG_RELEASE_COMP => TERMINATING
LEC ATM1/0.3: received RELEASE_COMPLETE
LEC ATM1/0.3:   callid          0x60EB66D4
LEC ATM1/0.3:   cause code      0
LEC ATM1/0.3: action A_PROCESS_TERM_REL_COMP
LEC ATM1/0.3: state TERMINATING event LEC_SIG_RELEASE_COMP => TERMINATING
LEC ATM1/0.2: received RELEASE_COMPLETE
LEC ATM1/0.2:   callid          0x60EB9E48
LEC ATM1/0.2:   cause code      0
```

```
LEC ATM1/0.2: action A_PROCESS_TERM_REL_COMP
LEC ATM1/0.2: state TERMINATING event LEC_SIG_RELEASE_COMP => IDLE
LEC ATM1/0.3: received RELEASE_COMPLETE
LEC ATM1/0.3:   callid          0x60EBA3BC
LEC ATM1/0.3:   cause code      0
LEC ATM1/0.3: action A_PROCESS_TERM_REL_COMP
LEC ATM1/0.3: state TERMINATING event LEC_SIG_RELEASE_COMP => IDLE
LEC ATM1/0.1: received RELEASE_COMPLETE
LEC ATM1/0.1:   callid          0x60EBD30C
LEC ATM1/0.1:   cause code      0
LEC ATM1/0.1: action A_PROCESS_TERM_REL_COMP
LEC ATM1/0.1: state TERMINATING event LEC_SIG_RELEASE_COMP => TERMINATING
LEC ATM1/0.1: received RELEASE_COMPLETE
LEC ATM1/0.1:   callid          0x60EBF174
LEC ATM1/0.1:   cause code      0
LEC ATM1/0.1: action A_PROCESS_TERM_REL_COMP
LEC ATM1/0.1: state TERMINATING event LEC_SIG_RELEASE_COMP => IDLE
LEC ATM1/0.2: received CANCEL
LEC ATM1/0.2: state IDLE event LEC_SIG_CANCEL => IDLE
LEC ATM1/0.3: received CANCEL
LEC ATM1/0.3: state IDLE event LEC_SIG_CANCEL => IDLE
LEC ATM1/0.1: received CANCEL
LEC ATM1/0.1: state IDLE event LEC_SIG_CANCEL => IDLE
LEC ATM1/0.1: action A_SHUTDOWN_LEC
LEC ATM1/0.1: sending CANCEL
LEC ATM1/0.1:   ATM address     47.00918100000000613E5A2F01.006070174820.01
LEC ATM1/0.1: state IDLE event LEC_LOCAL_DEACTIVATE => IDLE
LEC ATM1/0.2: action A_SHUTDOWN_LEC
LEC ATM1/0.2: sending CANCEL
LEC ATM1/0.2:   ATM address     47.00918100000000613E5A2F01.006070174820.02
LEC ATM1/0.2: state IDLE event LEC_LOCAL_DEACTIVATE => IDLE
LEC ATM1/0.3: action A_SHUTDOWN_LEC
LEC ATM1/0.3: sending CANCEL
LEC ATM1/0.3:   ATM address     47.00918100000000613E5A2F01.006070174820.03
LEC ATM1/0.3: state IDLE event LEC_LOCAL_DEACTIVATE => IDLE
```

The following output is from the **debug lane client mpoa** command when the **lane** interface is shut down:

```
Router# debug lane client mpoa
Router#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#int atm 1/1/0.1
Router(config-subif)#shutdown
Router(config-subif)#
00:23:32:%LANE-5-UPDOWN:ATM1/1/0.1 elan elan2:LE Client changed state to down
00:23:32:LEC ATM1/1/0.1:lec_inform_mpoa_state_chg:DOWN
00:23:32:LEC ATM1/1/0.1:lec_inform_mpoa_state_chg:DOWN
Router(config-subif)#
Router(config-subif)#
Router(config-subif)#
Router(config-subif)#exit
Router(config)#exit
```

The following output is from the **debug lane client mpoa** command when the **lane** interface is started (not shut down):

```
Router# debug lane client mpoa
Router#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#int atm 1/1/0.1
Router(config-subif)#
Router(config-subif)#
Router(config-subif)#no shutdown
Router(config-subif)#
00:23:39:LEC ATM1/1/0.1:lec_process_lane_tlv:msg LANE_CONFIG_RSP, num_tlvs 14
00:23:39:LEC ATM1/1/0.1:elan id from LECS set to 300
00:23:39:LEC ATM1/1/0.1:lec_process_lane_tlv:msg LANE_JOIN_RSP, num_tlvs 1
00:23:39:LEC ATM1/1/0.1:elan id from LES set to 300
00:23:39:LEC ATM1/1/0.1:lec_append_mpoa_dev_tlv:
00:23:39:LEC ATM1/1/0.1:got mpoa client addr 47.0091810000000050E2097801.0050A
29AF42D.00
00:23:39:%LANE-5-UPDOWN:ATM1/1/0.1 elan elan2:LE Client changed state to up
```

```
00:23:39:LEC ATM1/1/0.1:lec_inform_mpoa_state_chg:UP
00:25:57:LEC ATM1/1/0.1:lec_process_lane_tlv:msg LANE_ARP_REQ, num_tlvs 1
00:25:57:LEC ATM1/1/0.1:lec_process_dev_type_tlv:    lec 47.0091810000000050E
2097801.00500B306440.02
    type MPS, mpc 00.000000000000000000000000.000000000000.00
    mps 47.0091810000000050E2097801.00500B306444.00, num_mps_mac 1, mac 0050.0b3
0.6440
00:25:57:LEC ATM1/1/0.1:create mpoa_lec
00:25:57:LEC ATM1/1/0.1:new mpoa_lec 0x617E3118
00:25:57:LEC ATM1/1/0.1:lec_process_dev_type_tlv:type MPS, num       _mps_mac
 1
00:2t 5:57:LEC ATM1/1/0.1:lec_add_mps:
    remote lec 47.0091810000000050E2097801.00500B306440.02
    mps 47.0091810000000050E2097801.00500B306444.00 num_mps_mac 1, mac 0050.0b30
.6440
00:25:57:LEC ATM1/1/0.1:mpoa_device_change:lec_nsap 47.0091810000000050E20978
01.00500B306440.02, appl_type 5
     mpoa_nsap 47.0091810000000050E2097801.00500B306444.00, opcode 4
00:25:57:LEC ATM1/1/0.1:lec_add_mps:add mac 0050.0b30.6440, mps_mac 0x617E372
C
00:25:57:LEC ATM1/1/0.1:mpoa_device_change:lec_nsap 47.0091810000000050E20978
01.00500B306440.02, appl_type 5
     mpoa_nsap 47.0091810000000050E2097801.00500B306444.00, opcode 5
00:25:57:LEC ATM1/1/0.1:     mps_mac 0050.0b30.6440
00:25:57:LEC ATM1/1/0.1:lec_append_mpoa_dev_tlv:
00:25:57:LEC ATM1/1/0.1:got mpoa client addr 47.0091810000000050E2097801.0050A
29AF42D.00
Router(config-subif)#exit
Router(config)#exit
```

The following output is from the **debug lane client mpoa** command when the ATM major interface is shut down:

```
Router# debug lane client mpoa
Router#
conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#int atm 1/1/0
Router(config-if)# shutdown
Router(config-if)#
00:26:28:LANE ATM1/1/0:atm hardware reset
00:26:28:%LANE-5-UPDOWN:ATM1/1/0.1 elan elan2:LE Client changed state to down
00:26:28:LEC ATM1/1/0.1:lec_inform_mpoa_state_chg:DOWN
00:26:28:LEC ATM1/1/0.1:lec_inform_mpoa_state_chg:DOWN
00:26:28:%MPOA-5-UPDOWN:MPC mpc2:state changed to down
00:26:28:LEC ATM1/1/0.1:mpoa_to_lec:appl 6, opcode 0
00:26:30:%LINK-5-CHANGED:Interface ATM1/1/0, changed state to administratively
 down
00:26:30:LANE ATM1/1/0:atm hardware reset
00:26:31:%LINEPROTO-5-UPDOWN:Line protocol on Interface ATM1/1/0, changed stat
e to down
Router(config-if)#
00:26:31:LEC ATM1/1/0.1:mpoa_to_lec:appl 6, opcode 0
00:26:32:LANE ATM1/1/0:atm hardware reset
00:26:32:LEC ATM1/1/0.1:lec_inform_mpoa_state_chg:DOWN
00:26:34:LEC ATM1/1/0.1:lec_inform_mpoa_state_chg:DOWN
Router(config-if)# exit
Router(config)#
exit
```

The following output is from the **debug lane client mpoa** command when the ATM major interface is started:

```
Router# debug lane client mpoa
Router#
conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# int atm 1/1/0
Router(config-if)# no shutdown
00:26:32:LANE ATM1/1/0:atm hardware reset
00:26:32:LEC ATM1/1/0.1:lec_inform_mpoa_state_chg:DOWN
00:26:34:%LINK-3-UPDOWN:Interface ATM1/1/0, changed state to down
00:26:34:LANE ATM1/1/0:atm hardware reset
```

```
00:26:41:%LINK-3-UPDOWN:Interface ATM1/1/0, changed state to up
00:26:42:%LINEPROTO-5-UPDOWN:Line protocol on Interface ATM1/1/0, changed stat
e to up
00:27:10:%LANE-6-INFO:ATM1/1/0:ILMI prefix add event received
00:27:10:LANE ATM1/1/0:prefix add event for 470091810000000050E2097801 ptr=0x6
17BFC0C len=13
00:27:10:    the current first prefix is now:470091810000000050E2097801
00:27:10:%ATMSSCOP-5-SSCOPINIT:- Intf :ATM1/1/0, Event :Rcv End, State :Act
ive.
00:27:10:LEC ATM1/1/0.1:mpoa_to_lec:appl 6, opcode 0
00:27:10:%LANE-3-NOREGILMI:ATM1/1/0.1 LEC cannot register 47.0091810000000050E
2097801.0050A29AF428.01 with ILMI
00:27:10:%LANE-6-INFO:ATM1/1/0:ILMI prefix add event received
00:27:10:LANE ATM1/1/0:prefix add event for 470091810000000050E2097801 ptr=0x6
17B8E6C len=13
00:27:10:    the current first prefix is now:470091810000000050E2097801
00:27:10:%LANE-5-UPDOWN:ATM1/1/0.1 elan elan2:LE Client changed state to down
00:27:10:LEC ATM1/1/0.1:lec_inform_mpoa_state_chg:DOWN
00:27:10:LEC ATM1/1/0.1:mpoa_to_lec:appl 6, opcode 0
00:27:10:%MPOA-5-UPDOWN:MPC mpc2:state changed to up
00:27:10:LEC ATM1/1/0.1:mpoa_to_lec:appl 6, opcode 1
00:27:12:LEC ATM1/1/0.1:lec_process_lane_tlv:msg LANE_CONFIG_RSP, num_tlvs 14
00:27:12:LEC ATM1/1/0.1:elan id from LECS set to 300
00:27:12:LEC ATM1/1/0.1:lec_process_lane_tlv:msg LANE_JOIN_RSP, num_tlvs 1
00:27:12:LEC ATM1/1/0.1:elan id from LES set to 300
00:27:12:LEC ATM1/1/0.1:lec_append_mpoa_dev_tlv:
00:27:12:LEC ATM1/1/0.1:got mpoa client addr 47.0091810000000050E2097801.0050A
29AF42D.00
00:27:12:%LANE-5-UPDOWN:ATM1/1/0.1 elan elan2:LE Client changed state to up
00:27:12:LEC ATM1/1/0.1:lec_inform_mpoa_state_chg:UP
Router(config-if)#exit
Router(config)#exit
```

Related Commands

Command	Description
debug modem traffic	Displays MPC debug information.
debug mpoa server	Displays information about the MPOA server.

debug lane config

> **Note** Effective with Cisco IOS Release 15.1M, the **debug lane config** command is not available in Cisco IOS software.

To display information about a LAN Emulation (LANE) configuration server, use the **debug lane config** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug lane config {**all**| **events**| **packets**}

no debug lane config {**all**| **events**| **packets**}

Syntax Description

all	Displays all debugging messages related to the LANE configuration server. The output includes both the **events** and **packets** types of output.
events	Displays only messages related to significant LANE configuration server events.
packets	Displays information on each packet sent or received by the LANE configuration server.

Command Modes

Privileged EXEC

Command History

Release	Modification
15.1M	This command was removed.

Usage Guidelines

The **debug lane config** output is intended to be used primarily by a Cisco technical support representative.

Examples

The following is sample output from the **debug lane config all** command when an interface with LECS, an LES/BUS, and an LEC is shut down:

```
Router# debug lane config all
LECS EVENT ATM1/0: processing interface down transition
LECS EVENT ATM1/0: placed de-register address 0x60E8A824
(47.00918100000000613E5A2F01.006070174823.00) request with signalling
LECS EVENT ATM1/0: ilmiDeRegisterAddress: sendSetRequestToILMI failure; interface down ?
LECS EVENT ATM1/0: placed de-register address 0x60EC4F28
(47.007900000000000000000000.00A03E000001.00) request with signalling
LECS EVENT ATM1/0: ilmiDeRegisterAddress: sendSetRequestToILMI failure; interface down ?
LECS EVENT ATM1/0: placed de-register address 0x60EC5C08
```

```
(47.00918100000000613E5A2F01.006070174823.99) request with signalling
LECS EVENT ATM1/0: ilmiDeRegisterAddress: sendSetRequestToILMI failure; interface down ?
LECS EVENT ATM1/0: tearing down all connexions
LECS EVENT ATM1/0: elan 'xxx' LES 47.00918100000000613E5A2F01.006070174821.01 callId
0x60CE0F58 deliberately being disconnected
LECS EVENT ATM1/0: sending RELEASE for call 0x60CE0F58 cause 31
LECS EVENT ATM1/0: elan 'yyy' LES 47.00918100000000613E5A2F01.006070174821.02 callId
0x60CE2104 deliberately being disconnected
LECS EVENT ATM1/0: sending RELEASE for call 0x60CE2104 cause 31
LECS EVENT ATM1/0: elan 'zzz' LES 47.00918100000000613E5A2F01.006070174821.03 callId
0x60CE2DC8 deliberately being disconnected
LECS EVENT ATM1/0: sending RELEASE for call 0x60CE2DC8 cause 31
LECS EVENT ATM1/0: All calls to/from LECSs are being released
LECS EVENT ATM1/0: placed de-register address 0x60EC4F28
(47.007900000000000000000000.00A03E000001.00) request with signalling
LECS EVENT ATM1/0: ilmiDeRegisterAddress: sendSetRequestToILMI failure; interface down ?
LECS EVENT ATM1/0: ATM_RELEASE_COMPLETE received: callId 0x60CE0F58 cause 0
LECS EVENT ATM1/0: call 0x60CE0F58 cleaned up
LECS EVENT ATM1/0: ATM_RELEASE_COMPLETE received: callId 0x60CE2104 cause 0
LECS EVENT ATM1/0: call 0x60CE2104 cleaned up
LECS EVENT ATM1/0: ATM_RELEASE_COMPLETE received: callId 0x60CE2DC8 cause 0
LECS EVENT ATM1/0: call 0x60CE2DC8 cleaned up
LECS EVENT ATM1/0: UNKNOWN/UNSET: signalling DE-registered
LECS EVENT: UNKNOWN/UNSET: signalling DE-registered
LECS EVENT ATM1/0: UNKNOWN/UNSET: signalling DE-registered
LECS EVENT ATM1/0: placed de-register address 0x60E8A824
(47.00918100000000613E5A2F01.006070174823.00) request with signalling
LECS EVENT ATM1/0: ilmiDeRegisterAddress: sendSetRequestToILMI failure; interface down ?
LECS EVENT ATM1/0: placed de-register address 0x60EC5C08
(47.00918100000000613E5A2F01.006070174823.99) request with signalling
LECS EVENT ATM1/0: ilmiDeRegisterAddress: sendSetRequestToILMI failure; interface down ?
LECS EVENT ATM1/0: tearing down all connexions
LECS EVENT ATM1/0: All calls to/from LECSs are being released
LECS EVENT: config server 56 killed
```

debug lane finder

Note Effective with Cisco IOS Release 15.1M, the **debug lane finder**command is not available in Cisco IOS software.

To display information about the finder internal state machine, use the **debug lane finder** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug lane finder

no debug lane finder

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
15.1M	This command was removed.

Usage Guidelines

The **debug lane finder** command output is intended to be used primarily by a Cisco technical support representative.

Examples

The following is sample output from the **debug lane finder** command when an interface with LECS, LES/BUS, and LEC is shut down:

```
Router# debug lane finder
LECS FINDER ATM1/0.3: user request 1819 of type GET_MASTER_LECS_ADDRESS queued up
LECS FINDER ATM1/0: finder state machine started
LECS FINDER ATM1/0: time to perform a getNext on the ILMI
LECS FINDER ATM1/0: LECS 47.00918100000000613E5A2F01.006070174823.00 deleted
LECS FINDER ATM1/0: ilmi_client_request failed, answering all users
LECS FINDER ATM1/0: answering all requests now
LECS FINDER ATM1/0: responded to user request 1819
LECS FINDER ATM1/0: number of remaining requests still to be processed: 0
LECS FINDER ATM1/0.2: user request 1820 of type GET_MASTER_LECS_ADDRESS queued up
LECS FINDER ATM1/0: finder state machine started
LECS FINDER ATM1/0: time to perform a getNext on the ILMI
LECS FINDER ATM1/0: ilmi_client_request failed, answering all users
LECS FINDER ATM1/0: answering all requests now
LECS FINDER ATM1/0: responded to user request 1820
LECS FINDER ATM1/0: number of remaining requests still to be processed: 0
LECS FINDER ATM1/0.1: user request 1821 of type GET_MASTER_LECS_ADDRESS queued up
LECS FINDER ATM1/0: finder state machine started
LECS FINDER ATM1/0: time to perform a getNext on the ILMI
LECS FINDER ATM1/0: ilmi_client_request failed, answering all users
LECS FINDER ATM1/0: answering all requests now
```

```
LECS FINDER ATM1/0: responded to user request 1821
LECS FINDER ATM1/0: number of remaining requests still to be processed: 0
```

debug lane server

Note Effective with Cisco IOS Release 15.1M, the **debug lane server**command is not available in Cisco IOS software.

To display information about a LAN Emulation (LANE) server, use the **debug lane server** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug lane server [**interface** *interface*]

no debug lane server [**interface** *interface*]

Syntax Description

interface *interface*	(Optional) Limits the debugging output to messages relating to a specific interface or subinterface. If you use this command multiple times with different interfaces, the last interface entered is the one used to filter debugging messages.

Command Modes

Privileged EXEC

Command History

Release	Modification
15.1M	This command was removed.

Usage Guidelines

The **debug lane server** command output is intended to be used primarily by a Cisco technical support representative. The **debug lane server** command can generate a substantial amount of output. Specify a subinterface to decrease the amount of output and focus on the information you need.

Examples

The following is sample output from the **debug lane server** command when an interface with LECS, LES/BUS, and LEC is shut down:

```
Router# debug lane server
LES ATM1/0.1: lsv_lecsAccessSigCB called with callId 0x60CE124C, opcode ATM_RELEASE_COMPLETE
LES ATM1/0.1: disconnected from the master LECS
LES ATM1/0.1: should have been connected, will reconnect in 3 seconds
LES ATM1/0.2: lsv_lecsAccessSigCB called with callId 0x60CE29E0, opcode ATM_RELEASE_COMPLETE
LES ATM1/0.2: disconnected from the master LECS
LES ATM1/0.2: should have been connected, will reconnect in 3 seconds
LES ATM1/0.3: lsv_lecsAccessSigCB called with callId 0x60EB1940, opcode ATM_RELEASE_COMPLETE
LES ATM1/0.3: disconnected from the master LECS
LES ATM1/0.3: should have been connected, will reconnect in 3 seconds
LES ATM1/0.2: elan yyy client 1 lost control distribute
LES ATM1/0.2: elan yyy client 1: lsv_kill_client called
```

```
LES ATM1/0.2: elan yyy client 1 state change Oper -> Term
LES ATM1/0.3: elan zzz client 1 lost control distribute
LES ATM1/0.3: elan zzz client 1: lsv_kill_client called
LES ATM1/0.3: elan zzz client 1 state change Oper -> Term
LES ATM1/0.2: elan yyy client 1 lost MC forward
LES ATM1/0.2: elan yyy client 1: lsv_kill_client called
LES ATM1/0.3: elan zzz client 1 lost MC forward
LES ATM1/0.3: elan zzz client 1: lsv_kill_client called
LES ATM1/0.1: elan xxx client 1 lost control distribute
LES ATM1/0.1: elan xxx client 1: lsv_kill_client called
LES ATM1/0.1: elan xxx client 1 state change Oper -> Term
LES ATM1/0.1: elan xxx client 1 lost MC forward
LES ATM1/0.1: elan xxx client 1: lsv_kill_client called
LES ATM1/0.2: elan yyy client 1 released control direct
LES ATM1/0.2: elan yyy client 1: lsv_kill_client called
LES ATM1/0.3: elan zzz client 1 released control direct
LES ATM1/0.3: elan zzz client 1: lsv_kill_client called
LES ATM1/0.2: elan yyy client 1 MC forward released
LES ATM1/0.2: elan yyy client 1: lsv_kill_client called
LES ATM1/0.2: elan yyy client 1: freeing client structures
LES ATM1/0.2: elan yyy client 1 unregistered 0060.7017.4820
LES ATM1/0.2: elan yyy client 1 destroyed
LES ATM1/0.3: elan zzz client 1 MC forward released
LES ATM1/0.3: elan zzz client 1: lsv_kill_client called
LES ATM1/0.3: elan zzz client 1: freeing client structures
LES ATM1/0.3: elan zzz client 1 unregistered 0060.7017.4820
LES ATM1/0.3: elan zzz client 1 destroyed
LES ATM1/0.1: elan xxx client 1 released control direct
LES ATM1/0.1: elan xxx client 1: lsv_kill_client called
LES ATM1/0.1: elan xxx client 1 MC forward released
LES ATM1/0.1: elan xxx client 1: lsv_kill_client called
LES ATM1/0.1: elan xxx client 1: freeing client structures
LES ATM1/0.1: elan xxx client 1 unregistered 0060.7017.4820
LES ATM1/0.1: elan xxx client 1 destroyed
LES ATM1/0.1: elan xxx major interface state change
LES ATM1/0.1: cleanupLecsAccess: discarding all validation requests
LES ATM1/0.1: shutting down
LES ATM1/0.1: elan xxx: lsv_kill_lesbus called
LES ATM1/0.1: elan xxx: LES/BUS state change operational -> terminating
LES ATM1/0.1: cleanupLecsAccess: discarding all validation requests
LES ATM1/0.2: elan yyy major interface state change
LES ATM1/0.2: cleanupLecsAccess: discarding all validation requests
LES ATM1/0.2: shutting down
LES ATM1/0.2: elan yyy: lsv_kill_lesbus called
LES ATM1/0.2: elan yyy: LES/BUS state change operational -> terminating
LES ATM1/0.2: cleanupLecsAccess: discarding all validation requests
LES ATM1/0.3: elan zzz major interface state change
LES ATM1/0.3: cleanupLecsAccess: discarding all validation requests
LES ATM1/0.3: shutting down
LES ATM1/0.3: elan zzz: lsv_kill_lesbus called
LES ATM1/0.3: elan zzz: LES/BUS state change operational -> terminating
LES ATM1/0.3: cleanupLecsAccess: discarding all validation requests
LES ATM1/0.1: elan xxx: lsv_kill_lesbus called
LES ATM1/0.1: cleanupLecsAccess: discarding all validation requests
LES ATM1/0.1: elan xxx: lsv_kill_lesbus called
LES ATM1/0.1: cleanupLecsAccess: discarding all validation requests
LES ATM1/0.1: elan xxx: stopped listening on addresses
LES ATM1/0.1: elan xxx: all clients killed
LES ATM1/0.1: elan xxx: multicast groups killed
LES ATM1/0.1: elan xxx: addresses de-registered from ilmi
LES ATM1/0.1: elan xxx: LES/BUS state change terminating -> down
LES ATM1/0.1: elan xxx: administratively down
LES ATM1/0.2: elan yyy: lsv_kill_lesbus called
LES ATM1/0.2: cleanupLecsAccess: discarding all validation requests
LES ATM1/0.2: elan yyy: lsv_kill_lesbus called
LES ATM1/0.2: cleanupLecsAccess: discarding all validation requests
LES ATM1/0.2: elan yyy: stopped listening on addresses
LES ATM1/0.2: elan yyy: all clients killed
LES ATM1/0.2: elan yyy: multicast groups killed
LES ATM1/0.2: elan yyy: addresses de-registered from ilmi
LES ATM1/0.2: elan yyy: LES/BUS state change terminating -> down
LES ATM1/0.2: elan yyy: administratively down
LES ATM1/0.3: elan zzz: lsv_kill_lesbus called
```

```
LES ATM1/0.3: cleanupLecsAccess: discarding all validation requests
LES ATM1/0.3: elan zzz: lsv_kill_lesbus called
LES ATM1/0.3: cleanupLecsAccess: discarding all validation requests
LES ATM1/0.3: elan zzz: stopped listening on addresses
LES ATM1/0.3: elan zzz: all clients killed
LES ATM1/0.3: elan zzz: multicast groups killed
LES ATM1/0.3: elan zzz: addresses de-registered from ilmi
LES ATM1/0.3: elan zzz: LES/BUS state change terminating -> down
LES ATM1/0.3: elan zzz: administratively down
LES ATM1/0.3: cleanupLecsAccess: discarding all validation requests
LES ATM1/0.2: cleanupLecsAccess: discarding all validation requests
LES ATM1/0.1: cleanupLecsAccess: discarding all validation requests
```

debug lane signaling

Note Effective with Cisco IOS Release 15.1M, the **debug lane signaling**command is not available in Cisco IOS software.

To display information about LANE Server (LES) and Broadcast and Unknown Server (BUS) switched virtual circuits (SVCs), use the **debug lane signaling** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug lane signaling [**interface** *interface*]

no debug lane signaling [**interface** *interface*]

Syntax Description

interface *interface*	(Optional) Limits the debugging output to messages relating to a specific interface or subinterface. If you use this command multiple times with different interfaces, the last interface entered is the one used to filter debugging messages.

Command Modes

Privileged EXEC

Command History

Release	Modification
15.1M	This command was removed.

Usage Guidelines

The **debug lane signaling** command output is intended to be used primarily by a Cisco technical support representative. The **debug lane signaling** command can generate a substantial amount of output. Specify a subinterface to decrease the amount of output and focus on the information you need.

Examples

The following is sample output from the **debug lane signaling** command when an interface with LECS, LES/BUS, and LEC is shut down:

```
Router# debug lane signaling
LANE SIG ATM1/0.2: received ATM_RELEASE_COMPLETE callid 0x60EB565C cause 0 lv 0x60E8D348
lvstate LANE_VCC_CONNECTED
LANE SIG ATM1/0.2: lane_sig_mc_release: breaking lv 0x60E8D348 from mcg 0x60E97E84
LANE SIG ATM1/0.2: timer for lv 0x60E8D348 stopped
LANE SIG ATM1/0.2: sent ATM_RELEASE request for lv 0x60E8D468 in state LANE_VCC_CONNECTED
LANE SIG ATM1/0.2: sent ATM_RELEASE request for lv 0x60E8D3D8 in state LANE_VCC_CONNECTED
LANE SIG ATM1/0.2: sent ATM_RELEASE request for lv 0x60E8D2B8 in state LANE_VCC_CONNECTED
LANE SIG ATM1/0.3: received ATM_RELEASE_COMPLETE callid 0x60EB5CA0 cause 0 lv 0x60E8BEF4
lvstate LANE_VCC_CONNECTED
LANE SIG ATM1/0.3: lane_sig_mc_release: breaking lv 0x60E8BEF4 from mcg 0x60E9A37C
```

```
LANE SIG ATM1/0.3: timer for lv 0x60E8BEF4 stopped
LANE SIG ATM1/0.3: sent ATM_RELEASE request for lv 0x60E8C014 in state LANE_VCC_CONNECTED
LANE SIG ATM1/0.3: sent ATM_RELEASE request for lv 0x60E8BF84 in state LANE_VCC_CONNECTED
LANE SIG ATM1/0.3: sent ATM_RELEASE request for lv 0x60E8BE64 in state LANE_VCC_CONNECTED
LANE SIG ATM1/0.2: received ATM_RELEASE_COMPLETE callid 0x60EB9040 cause 0 lv 0x60E8D468
lvstate LANE_VCC_DROP_SENT
LANE SIG ATM1/0.2: lane_sig_mc_release: breaking lv 0x60E8D468 from mcg 0x60E97EC8
LANE SIG ATM1/0.2: timer for lv 0x60E8D468 stopped
LANE SIG ATM1/0.3: received ATM_RELEASE_COMPLETE callid 0x60EB97D4 cause 0 lv 0x60E8C014
lvstate LANE_VCC_DROP_SENT
LANE SIG ATM1/0.3: lane_sig_mc_release: breaking lv 0x60E8C014 from mcg 0x60E9A3C0
LANE SIG ATM1/0.3: timer for lv 0x60E8C014 stopped
LANE SIG ATM1/0.1: received ATM_RELEASE_COMPLETE callid 0x60EBCEB8 cause 0 lv 0x60EBBAF0
lvstate LANE_VCC_CONNECTED
LANE SIG ATM1/0.1: lane_sig_mc_release: breaking lv 0x60EBBAF0 from mcg 0x60E8F51C
LANE SIG ATM1/0.1: timer for lv 0x60EBBAF0 stopped
LANE SIG ATM1/0.1: sent ATM_RELEASE request for lv 0x60EBBC10 in state LANE_VCC_CONNECTED
LANE SIG ATM1/0.1: sent ATM_RELEASE request for lv 0x60EBBB80 in state LANE_VCC_CONNECTED
LANE SIG ATM1/0.1: sent ATM_RELEASE request for lv 0x60EBBA60 in state LANE_VCC_CONNECTED
LANE SIG ATM1/0.1: received ATM_RELEASE_COMPLETE callid 0x60EBEB00 cause 0 lv 0x60EBBC10
lvstate LANE_VCC_DROP_SENT
LANE SIG ATM1/0.1: lane_sig_mc_release: breaking lv 0x60EBBC10 from mcg 0x60E8F560
LANE SIG ATM1/0.1: timer for lv 0x60EBBC10 stopped
LANE SIG ATM1/0.2: received ATM_RELEASE_COMPLETE callid 0x60E8B174 cause 0 lv 0x60E8D2B8
lvstate LANE_VCC_RELEASE_SENT
LANE SIG ATM1/0.2: timer for lv 0x60E8D2B8 stopped
LANE SIG ATM1/0.3: received ATM_RELEASE_COMPLETE callid 0x60E8B990 cause 0 lv 0x60E8BE64
lvstate LANE_VCC_RELEASE_SENT
LANE SIG ATM1/0.3: timer for lv 0x60E8BE64 stopped
LANE SIG ATM1/0.2: received ATM_RELEASE_COMPLETE callid 0x60EB7FE0 cause 0 lv 0x60E8D3D8
lvstate LANE_VCC_RELEASE_SENT
LANE SIG ATM1/0.2: timer for lv 0x60E8D3D8 stopped
LANE SIG ATM1/0.3: received ATM_RELEASE_COMPLETE callid 0x60EB8554 cause 0 lv 0x60E8BF84
lvstate LANE_VCC_RELEASE_SENT
LANE SIG ATM1/0.3: timer for lv 0x60E8BF84 stopped
LANE SIG ATM1/0.1: received ATM_RELEASE_COMPLETE callid 0x60EBB6D4 cause 0 lv 0x60EBBA60
lvstate LANE_VCC_RELEASE_SENT
LANE SIG ATM1/0.1: timer for lv 0x60EBBA60 stopped
LANE SIG ATM1/0.1: received ATM_RELEASE_COMPLETE callid 0x60EBE24C cause 0 lv 0x60EBBB80
lvstate LANE_VCC_RELEASE_SENT
LANE SIG ATM1/0.1: timer for lv 0x60EBBB80 stopped
LANE SIG ATM1/0.1: sent ATM_CANCEL_NSAP request for lv 0x0 in state NULL_VCC_POINTER
LANE SIG ATM1/0.1: sent ATM_CANCEL_NSAP request for lv 0x0 in state NULL_VCC_POINTER
LANE SIG ATM1/0.2: sent ATM_CANCEL_NSAP request for lv 0x0 in state NULL_VCC_POINTER
LANE SIG ATM1/0.2: sent ATM_CANCEL_NSAP request for lv 0x0 in state NULL_VCC_POINTER
LANE SIG ATM1/0.3: sent ATM_CANCEL_NSAP request for lv 0x0 in state NULL_VCC_POINTER
LANE SIG ATM1/0.3: sent ATM_CANCEL_NSAP request for lv 0x0 in state NULL_VCC_POINTER
LANE SIG ATM1/0.1: received ATM_CANCEL_NSAP for nsap
00.000000000000050000000000.000000000000.00
LANE SIG ATM1/0.1: received ATM_CANCEL_NSAP for nsap
00.000000000000050000000000.000000000000.00
LANE SIG ATM1/0.2: received ATM_CANCEL_NSAP for nsap
00.000000000000050000000000.000000000000.00
LANE SIG ATM1/0.2: received ATM_CANCEL_NSAP for nsap
00.000000000000050000000000.000000000000.00
LANE SIG ATM1/0.3: received ATM_CANCEL_NSAP for nsap
00.000000000000050000000000.000000000000.00
LANE SIG ATM1/0.3: received ATM_CANCEL_NSAP for nsap
00.000000000000050000000000.000000000000.00
```

debug lapb

To display all traffic for interfaces using Link Access Procedure, Balanced (LAPB) encapsulation, use the **debug lapb**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug lapb

no debug lapb

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
11.0	This command was introduced prior to this release.

Usage Guidelines

This command displays information on the X.25 Layer 2 protocol. It is useful to users familiar with the LAPB protocol.

You can use the **debug lapb** command to determine why X.25 interfaces or LAPB connections are going up and down. It is also useful for identifying link problems, as evidenced when the **show interfaces** EXEC command displays a high number of rejects or frame errors over the X.25 link.

The **debug lapb** command can generate debugging messages of LAPB on all interfaces configured with the **encapsulation lapb** command or when X.25 traffic is present on interfaces configured with the **encapsulation x25**command. LAPB debugging produces a substantial amount of data and makes debugging very tedious. The problem becomes more severe if the network contains a large number of X.25 interfaces. Therefore the LAPB debugs are set to be available for individual interface.

Caution Because the **debug lapb** command generates a substantial amount of output, use it when the aggregate of all LAPB traffic on X.25 and LAPB interfaces is fewer than five frames per second.

Examples

The following is sample output from the **debug lapb** command (the numbers 1 through 7 at the top of the display have been added in order to aid documentation):

```
1          2   3    4     5    6    7
Serial0: LAPB I CONNECT (5) IFRAME P 2 1
Serial0: LAPB O REJSENT (2) REJ F 3
Serial0: LAPB O REJSENT (5) IFRAME 0 3
Serial0: LAPB I REJSENT (2) REJ (C) 7
Serial0: LAPB I DISCONNECT (2) SABM P
Serial0: LAPB O CONNECT (2) UA F
```

```
Serial0: LAPB O CONNECT (5) IFRAME 0 0
Serial0: LAPB T1 CONNECT 357964 0
```

Each line of output describes a LAPB event. There are two types of LAPB events: frame events (when a frame enters or exits the LAPB) and timer events. In the sample output, the last line describes a timer event; all of the other lines describe frame events. The table below describes the first seven fields.

Table 96: debug lapb Field Descriptions

Field	Description
First field (1)	Interface type and unit number reporting the frame event.
Second field (2)	Protocol providing the information.
Third field (3)	Frame event type. Possible values are as follows: • I--Frame input • O--Frame output • T1--T1 timer expired • T3--Interface outage timer expired • T4--Idle link timer expired
Fourth field (4)	State of the protocol when the frame event occurred. Possible values are as follows: • BUSY (RNR frame received) • CONNECT • DISCONNECT • DISCSENT (disconnect sent) • ERROR (FRMR frame sent) • REJSENT (reject frame sent) • SABMSENT (SABM frame sent)
Fifth field (5)	In a frame event, this value is the size of the frame (in bytes). In a timer event, this value is the current timer value (in milliseconds).

Field	Description
Sixth field (6)	In a frame event, this value is the frame type name. Possible values for f rame type names are as follows: • DISC--Disconnect • DM--Disconnect mode • FRMR--Frame reject • IFRAME--Information frame • ILLEGAL--Illegal LAPB frame • REJ--Reject • RNR--Receiver not ready • RR--Receiver ready • SABM--Set asynchronous balanced mode • SABME--Set asynchronous balanced mode, extended • UA--Unnumbered acknowledgment In a T1 timer event, this value is the number of retransmissions already attempted.
Seventh field (7) (This field will not print if the frame control field is required to appear as either a command or a response, and that frame type is correct.)	This field is present only in frame events. It describes the frame type identified by the LAPB address and Poll/Final bit. Possible values are as follows: • (C)--Command frame • (R)--Response frame • P--Command/Poll frame • F--Response/Final frame • /ERR--Command/Response type is invalid for the control field. An ?ERR generally means that the data terminal equipment (DTE)/data communications equipment (DCE) assignments are not correct for this link. • BAD-ADDR--Address field is neither Command nor Response

A timer event displays only the first six fields of **debug lapb** command output. For frame events, however, the seventh field documents the LAPB control information present in the frame. Depending on the value of the frame type name shown in the sixth field, the seventh field may or may not appear.

After the Poll/Final indicator, depending on the frame type, three different types of LAPB control information can be printed.

For information frames, the value of the N(S) field and the N(R) field will be printed. The N(S) field of an information frame is the sequence number of that frame, so this field will rotate between 0 and 7 for (modulo 8 operation) or 0 and 127 (for modulo 128 operation) for successive outgoing information frames and (under normal circumstances) also will rotate for incoming information frame streams. The N(R) field is a "piggybacked" acknowledgment for the incoming information frame stream; it informs the other end of the link which sequence number is expected next.

RR, RNR, and REJ frames have an N(R) field, so the value of that field is printed. This field has exactly the same significance that it does in an information frame.

For the FRMR frame, the error information is decoded to display the rejected control field, V(R) and V(S) values, the Response/Command flag, and the error flags WXYZ.

In the following example, the output shows an idle link timer action (T4) where the timer expires twice on an idle link, with the value of T4 set to five seconds:

```
Serial2: LAPB T4 CONNECT 255748
Serial2: LAPB O CONNECT (2) RR P 5
Serial2: LAPB I CONNECT (2) RR F 5
Serial2: LAPB T4 CONNECT 260748
Serial2: LAPB O CONNECT (2) RR P 5
Serial2: LAPB I CONNECT (2) RR F 5
```

The next example shows an interface outage timer expiration (T3):

```
Serial2: LAPB T3 DISCONNECT 273284
```

The following example output shows an error condition when no DCE to DTE connection exists. Note that if a frame has only one valid type (for example, a SABM can only be a command frame), a received frame that has the wrong frame type will be flagged as a receive error (R/ERR in the following output). This feature makes misconfigured links (DTE-DTE or DCE-DCE) easy to spot. Other less common errors will also be highlighted, such as a too-short or too-long frame or an invalid address (neither command nor response).

```
Serial2: LAPB T1 SABMSENT 1026508 1
Serial2: LAPB O SABMSENT (2) SABM P
Serial2: LAPB I SABMSENT (2) SABM (R/ERR)
Serial2: LAPB T1 SABMSENT 1029508 2
Serial2: LAPB O SABMSENT (2) SABM P
Serial2: LAPB I SABMSENT (2) SABM (R/ERR)
```

The output in the next example shows tha the router is misconfigured and has a standard (modulo 8) interface connected to an extended (modulo 128) interface. This condition is indicated by the SABM balanced mode and SABME balanced mode extended messages appearing on the same interface.

```
Serial2: LAPB T1 SABMSENT 1428720 0
Serial2: LAPB O SABMSENT (2) SABME P
Serial2: LAPB I SABMSENT (2) SABM P
Serial2: LAPB T1 SABMSENT 1431720 1
Serial2: LAPB O SABMSENT (2) SABME P
Serial2: LAPB I SABMSENT (2) SABM P
```

The output in the next example shows that the **debug lapb**command is set for a single interface; that is, interface 0/0.

```
Serial0/0: LAPB O CONNECT (17) IFRAME 1 7
Serial0/0: LAPB I CONNECT (5) IFRAME 7 2
Serial0/0: LAPB I CONNECT (6) IFRAME 0 2
Serial0/0: LAPB O CONNECT (2) RR (R) 1
Serial0/0: LAPB O CONNECT (50) IFRAME 2 1
Serial0/0: LAPB I CONNECT (15) IFRAME 1 2
Serial0/0: LAPB O CONNECT (5) IFRAME 3 2
```

debug lapb-ta

To display debugging messages for Link Access Procedure, Balanced-Terminal Adapter (LAPB-TA), use the **debug lapb-ta** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug lapb-ta [**error**| **event**| **traffic**]

no debug lapb-ta [**error**| **event**| **traffic**]

Syntax Description

error	(Optional) Displays LAPB-TA errors.
event	(Optional) Displays LAPB-TA normal events.
traffic	(Optional) Displays LAPB-TA in/out traffic data.

Command Default

Debugging for LAPB-TA is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(4)T	This command was introduced.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Examples

The following is sample output from the **debug lapb-ta** command with the **error**, **event**, and **traffic** keywords activated:

```
Router# debug lapb-ta error
LAPB-TA error debugging is on
Router# debug lapb-ta event
LAPB-TA event debugging is on
Router# debug lapb-ta traffic
LAPB-TA traffic debugging is on
Mar  9 12:11:36.464:LAPB-TA:Autodetect trying to detect LAPB on
BR3/0:1
Mar  9 12:11:36.464:  sampled pkt: 2 bytes: 1 3F.. match
Mar  9 12:11:36.468:LAPBTA:get_ll_config:BRI3/0:1
Mar  9 12:11:36.468:LAPBTA:line 130 allocated for BR3/0:1
Mar  9 12:11:36.468:LAPBTA:process 79
Mar  9 12:11:36.468:BR3/0:1:LAPB-TA started
Mar  9 12:11:36.468:LAPBTA:service change:LAPB physical layer up,
context 6183E144 interface up, protocol down
Mar  9 12:11:36.468:LAPBTA:service change:, context 6183E144 up
Mar  9 12:11:36.468:LAPB-TA:BR3/0:1, 44 sent
2d14h:%LINEPROTO-5-UPDOWN:Line protocol on Interface BRI3/0:1, changed state to up
```

```
2d14h:%ISDN-6-CONNECT:Interface BRI3/0:1 is now connected to 60213
Mar  9 12:11:44.508:LAPB-TA:BR3/0:1, 1 rcvd
Mar  9 12:11:44.508:LAPB-TA:BR3/0:1, 3 sent
Mar  9 12:11:44.700:LAPB-TA:BR3/0:1, 1 rcvd
Mar  9 12:11:44.700:LAPB-TA:BR3/0:1, 3 sent
Mar  9 12:11:44.840:LAPB-TA:BR3/0:1, 1 rcvd
Mar  9 12:11:44.840:LAPB-TA:BR3/0:1, 14 sent
Mar  9 12:11:45.852:LAPB-TA:BR3/0:1, 1 rcvd
Mar  9 12:11:46.160:LAPB-TA:BR3/0:1, 2 rcvd
Mar  9 12:11:47.016:LAPB-TA:BR3/0:1, 1 rcvd
Mar  9 12:11:47.016:LAPB-TA:BR3/0:1, 10 sent
```

debug lat packet

To display information on all local-area transport (LAT) events, use the **debug lat packet** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug lat packet

no debug lat packet

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

For each datagram (packet) received or sent, a message is logged to the console.

Caution This command severely impacts LAT performance and is intended for troubleshooting use only.

Examples

The following is sample output from the **debug lat packet** command:

```
Router# debug lat packet
LAT: I int=Ethernet0, src=0000.0c01.0509, dst=0900.2b00.000f, type=0, M=0, R=0
LAT: I int=Ethernet0, src=0800.2b11.2d13, dst=0000.0c01.7876, type=A, M=0, R=0
LAT: O dst=0800.2b11.2d13, int=Ethernet0, type= A, M=0, R=0, len= 20, next 0 ref 1
```

The second line of output describes a packet that is input to the router. The table below describes the fields in this line.

Table 97: debug lat packet Field Descriptions

Field	Description
LAT:	Indicates that this display shows LAT debugging output.
I	Indicates that this line of output describes a packet that is input to the router (I) or output from the router (O).
int = Ethernet0	Indicates the interface on which the packet event took place.
src = 0800.2b11.2d13	Indicates the source address of the packet.
dst=0000.0c01.7876	Indicates the destination address of the packet.

Field	Description
type=A	Indicates the message type (in hexadecimal notation). Possible values are as follows: • 0 = Run Circuit • 1 = Start Circuit • 2 = Stop Circuit • A = Service Announcement • C = Command • D = Status • E = Solicit Information • F = Response Information

The third line of output describes a packet that is output from the router. The table below describes the last three fields in this line.

Table 98: debug lat packet Field Descriptions

Field	Description
len= 20	Indicates the length (in hexadecimal notation) of the packet (in bytes).
next 0	Indicates the link on the transmit queue.
ref 1	Indicates the count of packet users.

debug ldap

To enable debugging for Lightweight Directory Access Protocol (LDAP) configuration, use the **debug ldap** command in privileged EXEC mode. To disable debugging, use the no form of this command.

debug ldap {**all| error| event| legacy| packet**}

no debug ldap {**all| error| event| legacy| packet**}

Syntax Description

all	Displays all event, legacy, and packet related messages.
error	Displays error messages about the local authentication server.
event	Displays debug messages related to LDAP proxy events.
legacy	Displays legacy messages.
packet	Displays the content of the RADIUS packets that are sent and received.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
15.1(1)T	This command was introduced.

Examples

The following is sample output from the **debug ldap legacy** command:

```
Router# debug ldap legacy
put_filter "(&(objectclass=*)(cn=firewall_user))"
put_filter: AND
put_filter_list "(objectclass=*)(cn=firewall_user)"
put_filter "(objectclass=*)"
put_filter: simple
put_filter "(cn=firewall_user)"
put_filter: simple
Doing socket writeldap_result
wait4msg (timeout 0 sec, 1 usec)
ldap_select_fd_wait (select)
ldap_read_activity lc 0x6804D354
Doing socket read
LDAP-TCP:Bytes read = 1478
ldap_match_request succeeded for msgid 2 h 0
ldap_get_dn
```

```
ldap_get_dn
ldap_msgfree
ldap_result
wait4msg (timeout 0 sec, 1 usec)
ldap_read_activity lc 0x6804D354
ldap_match_request succeeded for msgid 2 h 0
changing lr 0x6774F8D4 to COMPLETE as no continuations
removing request 0x6774F8D4 from list as lm 0x681C9B78 all 0
ldap_msgfree
ldap_msgfree
ldap_parse_result
ldap_parse_result
ldap_req_encode
Doing socket writeldap_msgfree
ldap_result
wait4msg (timeout 0 sec, 1 usec)
ldap_select_fd_wait (select)
ldap_result
wait4msg (timeout 0 sec, 1 usec)
ldap_select_fd_wait (select)
ldap_read_activity lc 0x6804D354
Doing socket read
LDAP-TCP:Bytes read = 22
ldap_match_request succeeded for msgid 3 h 0
changing lr 0x6774F8D4 to COMPLETE as no continuations
removing request 0x6774F8D4 from list as lm 0x681C9B78 all 0
ldap_msgfree
ldap_msgfree
ldap_parse_result
ldap_parse_result
ldap_msgfree
ldap_result
wait4msg (timeout 0 sec, 1 usec)
ldap_select_fd_wait (select)
```

Related Commands

Command	Description
ipv4 (ldap)	Creates an IPv4 address within an LDAP server address pool
ldap server	Defines an LDAP server and enters LDAP server configuration mode.
transport port (ldap)	Configures the transport protocol for establishing a connection with the LDAP server.

debug lex rcmd

To debug LAN Extender remote commands, use the **debug lex rcmd** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug lex rcmd

no debug lex rcmd

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Examples

The following is sample output from the **debug lex rcmd** command:

```
Router# debug lex rcmd
LEX-RCMD: "shutdown" command received on unbound serial interface- Serial0
LEX-RCMD: Lex0: "inventory" command received
Rcvd rcmd: FF 03 80 41 41 13 00 1A 8A 00 00 16 01 FF 00 00
Rcvd rcmd: 00 02 00 00 07 5B CD 15 00 00 0C 01 15 26
LEX-RCMD: ACK or response received on Serial0 without a corresponding ID
LEX-RCMD: REJ received
LEX-RCMD: illegal CODE field received in header: <number>
LEX-RCMD: illegal length for Lex0: "lex input-type-list"
LEX-RCMD: Lex0 is not bound to a serial interface
LEX-RCMD: encapsulation failure
LEX-RCMD: timeout for Lex0: "lex priority-group" command
LEX-RCMD: re-transmitting Lex0: "lex priority-group" command
LEX-RCMD: lex_setup_and_send called with invalid parameter
LEX-RCMD: bind occurred on shutdown LEX interface
LEX-RCMD: Serial0- No free Lex interface found with negotiated MAC address 0000.0c00.d8db
LEX-RCMD: No active Lex interface found for unbind
```

The following output indicates that a LAN Extender remote command packet was received on a serial interface that is not bound to a LAN Extender interface:

```
LEX-RCMD: "shutdown" command received on unbound serial interface- Serial0
```

This message can occur for any of the LAN Extender remote commands. Possible causes of this message are as follows:

- FLEX state machine software error
- Serial line momentarily goes down, which is detected by the host but not by FLEX

The following output indicates that a LAN Extender remote command response has been received. The hexadecimal values are for internal use only.

```
LEX-RCMD: Lex0: "inventory" command received
Rcvd rcmd: FF 03 80 41 41 13 00 1A 8A 00 00 16 01 FF 00 00
Rcvd rcmd: 00 02 00 00 07 5B CD 15 00 00 0C 01 15 26
```

The following output indicates that when the host router originates a LAN Extender remote command to FLEX, it generates an 8-bit identifier that is used to associate a command with its corresponding response:

```
LEX-RCMD: ACK or response received on Serial0 without a corresponding ID
```

This message could be displayed for any of the following reasons:

- FLEX was busy at the time that the command arrived and could not send an immediate response. The command timed out on the host router and then FLEX finally sent the response.
- Transmission error.
- Software error.

Possible responses to Config-Request are Config-ACK, Config-NAK, and Config-Rej. The following output shows that some of the options in the Config-Request are not recognizable or are not acceptable to FLEX due to transmission errors or software errors:

```
LEX-RCMD: REJ received
```

The following output shows that a LAN Extender remote command response was received but that the CODE field in the header was incorrect:

```
LEX-RCMD: illegal CODE field received in header: <number>
```

The following output indicates that a LAN Extender remote command response was received but that it had an incorrect length field. This message can occur for any of the LAN Extender remote commands.

```
LEX-RCMD: illegal length for Lex0: "lex input-type-list"
```

The following output shows that a host router was about to send a remote command when the serial link went down:

```
LEX-RCMD: Lex0 is not bound to a serial interface
```

The following output shows that the serial encapsulation routine of the interface failed to encapsulate the remote command datagram because the LEX-NCP was not in the OPEN state. Due to the way the PPP state machine is implemented, it is normal to see a single encapsulation failure for each remote command that gets sent at bind time.

```
LEX-RCMD: encapsulation failure
```

The following output shows that the timer expired for the given remote command without having received a response from the FLEX device. This message can occur for any of the LAN Extender remote commands.

```
 LEX-RCMD: timeout for Lex0: "lex priority-group" command
```

This message could be displayed for any of the following reasons:

- FLEX too busy to respond
- Transmission failure
- Software error

The following output indicates that the host is resending the remote command after a timeout:

```
LEX-RCMD: re-transmitting Lex0: “lex priority-group” command
```

The following output indicates that an illegal parameter was passed to the lex_setup_and_send routine. This message could be displayed due to a host software error.

```
LEX-RCMD: lex_setup_and_send called with invalid parameter
```

The following output is informational and shows when a bind occurs on a shutdown interface:

```
LEX-RCMD: bind occurred on shutdown LEX interface
```

The following output shows that the LEX-NCP reached the open state and a bind operation was attempted with the FLEX's MAC address, but no free LAN Extender interfaces were found that were configured with that MAC address. This output can occur when the network administrator does not configure a LAN Extender interface with the correct MAC address.

```
LEX-RCMD: Serial0- No free Lex interface found with negotiated MAC address 0000.0c00.d8db
```

The following output shows that the serial line that was bound to the LAN Extender interface went down and the unbind routine was called, but when the list of active LAN Extender interfaces was searched, the LAN Extender interface corresponding to the serial interface was not found. This output usually occurs because of a host software error.

```
LEX-RCMD: No active Lex interface found for unbind
```

debug license

To enable controlled Cisco IOS software license debugging activity on a device, use the **debug license** command in privileged EXEC mode. To disable debugging, use the no form of this command.

debug license {**agent** {**all**| **error**}| **core** {**all**| **errors**| **events**}| **errors**| **events**| **ipc**}

no debug license {**agent** {**all**| **error**}| **core** {**all**| **errors**| **events**}| **errors**| **events**| **ipc**}

Cisco ASR 1001 Router Platforms

debug license {**core** {**all**| **errors**| **events**}| **errors**| **ipc**}

no debug license {**core** {**all**| **errors**| **events**}| **errors**| **ipc**}

Syntax Description

agent	Debugs license agent information. • **all** --Debugs all license agent messages. • **error** --Debugs only license agent error messages.
core	Debugs messages from a license core module. • **all** --Debugs all license core messages • **errors** --Debugs only license core error messages • **events** --Debugs only license core event messages.
errors	Debugs license warnings and errors.
events	Debugs license event messages.
ipc	Debugs license interprocess communication (IPC) messages.

Command Default

Debugging is disabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(35)SE2	This command was introduced.

Release	Modification
12.4(15)XZ	This command was integrated into Cisco IOS Release 12.4(15)XZ.
12.4(20)T	This command was integrated into Cisco IOS Release 12.4(20)T.
Cisco IOS XE Release 3.2S	This command was implemented on the Cisco ASR 1001 router.

Usage Guidelines

Use this command to help troubleshoot issues with licenses on a device.

On the Cisco ASR 1001 router, the output from the **debug license** command is not in standard IOS format. You must execute the **request platform software trace rotate all** privileged EXEC command to make the output in the log files in the bootflash:tracelogs directory.

Examples

The following example shows how to enable debugging for license warnings and errors on a router:

```
Router# debug license errors
```

The following example shows how to enable debugging for all license agent information on a switch:

```
Switch# debug license agent all
license agent app https[0x43FBC7C]: urlhook function
license agent app https[0x43FBC7C]: https action function
LIC_AGENT:Processing XML message
```

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP:Envelope xmlns:SOAP="http://www.w3.org/2003/05/soap-envelope">
<SOAP:Header>
<clm:Header version="1.0" xmlns:clm="http://www.cisco.com/clm">
<clm:Time>2003-04-23T20:27:19.827Z</clm:Time>
</clm:Header>
</SOAP:Header>
<SOAP:Body>
<lica:request xmlns:lica="http://www.cisco.com/clm">
<lica:installRequest>
<lica:license encoding="BASE64">
PENJU0NPX1dUX0FSVElGQUNUUyB2ZXJzaW9uPSIxLjAiPjxDSVNDT19XVF9MSUNFTlNFIHZlcnNp
b249IjEuMCI+PEZFQVRVUkVfTkFNRT5pcGJhc2U8L0ZFQVRVUkVfTkFNRT48RkVBVFVSRV9WRVJT
SU9OPjEuMDwvRkVBVFVSRV9WRVJTSU9OPjxVREk+PFBJRD5CVUxMU0VZRTI0PC9QSUQ+PFNOPkNB
VDEwMDZSMEU4PC9TTj48L1VEST48U09VUkNFPkNpc2NvIEhRPC9TT1VSQ0U+PENSRUFURV9EQVRF
PjIwMDYtMTEtMjJUMDA6MzM6NTA8L0NSRUFURV9EQVRFPjxMSUNFTlNFX0xJTkVfSEFTSCBoYXNo
QWxnbz0iU0hBMSI+NDJiNFVWWFpOd3pJK0ZNdEV6QlNZSDRWdzFFPTwvTElDRU5TRV9MSU5FX0hB
U0g+PFRZUEU+UkVHVUxBUjwvVFlQRT48TElDRU5TRV9MSU5FPjwhW0NEQVRBWzExIGlwYmFzZSAx
LjAgTE9ORyBOT1JNQUwgU1RBTkRBTE9ORSBFWENMIElORklOSVRFX0tFWVMgSU5GSU5JVEVfS0VZ
UyBORVZFUiBORVZFUiBOaUwgU0xNX0NPREUgQ0xfTkRfTENLIE5pTCAqMVZBUlY5WlJESzREOU5U
NDAwIE5pTCBOaUwgTmlMIDVfTUlOUyA8VURJPjxQSUQ+QlVMTFNFWUUyNDwvUElEPjxTTj5DQVQx
MDA2UjBFODwvU04+PC9VREk+IGUxWW8wS1U2VnJLOnBJZXRib1dJVkEyZlVaVGdieUlEaklHWERR
VXc3dkxOYWlXRzZ0dUJOMG5lTXpKaHpzQ2tMN1l3TWFxS2paem05YW5FbVJHUUVPTHlDdmRVZksw
QmNLN0pPcnZsUkw0VjMyJDxXTEM+QVFFQklRQUIvLy9GbS8vWDkybThhNb0NOZkVMSHJiVzRjWDFM
ZGNpdDNMVU5GWlV1OWppT0phcXB5Q2N6TTFpaUlKbVE3NEd5WHJFY3F2UGlBbVdTYUVtVVQ1NnJz
dGs2Z3ZtaitFUUtSZkQ5QTBpbWUxY3pyZEt4ZklMVDBMYVhUNDE2bndtZnA5MlR5YTZ2SVE0Rm5s
QmRxSjFzTXpYZVNxOFBtVmNUVTlBNG85aGlsOXZLdXI4TjlGODg1RDlHVkYwYkpIY2lUNU09PC9X
TEM+XV0+PC9MSUNFTlNFX0xJTkU+PFVTRVJfTU9ESUZJQUJMRV9DT01NRU5UIGZpZWxkUmVzdHJp
Y3Rpb25zPSJNYXggOTkgQVNDSUkgY2hhcmFjdGVycyBpbiBsZW5ndGguIj48L1VTRVJfTU9ESUZJ
QUJMRV9DT01NRU5UPjwvQ0lTQ09fV1RfTElDRU5TRT48L0NJU0NPX1dUX0FSVElGQUNUUz4=
</lica:license>
</lica:installRequest>
</lica:request>
</SOAP:Body>
</SOAP:Envelope>
```

```
LIC_AGENT: XML received opcode(1)
LIC_AGENT: License ipbase
%IOS_LICENSE_IMAGE_APPLICATION-6-LICENSE_LEVEL: Next reboot level = ipbase and License =
```

```
ipbase
LIC_AGENT: Notification Event type = 1 License Installed
LIC_AGENT: Notification Event type = 13 License Annotate
```

debug link monitor

To display the statistics of the executing process, use the **debug link monitor** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug link monitor

no debug link monitor

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(1)	This command was introduced.

Usage Guidelines

This command is used to display the statistics, which are used for debugging the status of the various conditions occurred during execution of the monitoring process.

Examples

The following example enables link monitoring statistics:

```
Router# debug link monitor
%DEBUG-ENABLED Error Rate Link Monitor
```

The following example disables link monitoring statistics:

```
Router# no debug link monitor

%DEBUG-DISABLED Error Rate Link Monitor
```

Related Commands

Command	Description
debug all	Enables debugging for link monitoring.
no debug all	Disables debugging for link monitoring.
clear counters	Clears show interface counters on all interfaces.
show link monitor debug	Show link monitor error statistics.

debug list

To filter debugging information on a per-interface or per-access list basis, use the **debug list** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug list [*list*] [*interface*]

no debug list [*list*] [*interface*]

Syntax Description

list	(Optional) An access list number in the range from 1100 to 1199.
interface	(Optional) The interface type. Allowed values are the following: • **channel** --IBM Channel interface • **ethernet** --IEEE 802.3 • **fddi** --ANSI X3T9.5 • **null** --Null interface • **serial** --Serial • **tokenring** --IEEE 802.5 • **tunnel** --Tunnel interface

Command Modes

Privileged EXEC

Usage Guidelines

The **debug list** command is used with other **debug** commands for specific protocols and interfaces to filter the amount of debug information that is displayed. In particular, this command is designed to filter specific physical unit (PU) output from bridging protocols. The **debug list** command is supported with the following commands:

- **debug arp**
- **debug llc2 errors**
- **debug llc2 packets**
- **debug llc2 state**
- **debug rif**
- **debug sdlc**
- **debug token ring**

Note All **debug** commands that support access list filtering use access lists in the range from 1100 to 1199. The access list numbers shown in the examples are merely samples of valid numbers.

Examples

To use the **debug list** command on only the first of several Logical Link Control, type 2 (LLC2) connections, use the **show llc2** command to display the active connections:

```
Router# show llc2
SdllcVirtualRing2008 DTE: 4000.2222.22c7 4000.1111.111c 04 04 state NORMAL
SdllcVirtualRing2008 DTE: 4000.2222.22c8 4000.1111.1120 04 04 state NORMAL
SdllcVirtualRing2008 DTE: 4000.2222.22c1 4000.1111.1104 04 04 state NORMAL
```

Next, configure an extended bridging access list, numbered 1103, for the connection you want to filter:

```
access-list 1103 permit 4000.1111.111c 0000.0000.0000 4000.2222.22c7 0000.0000.0000 0xC 2
eq 0x404
```

The convention for the LLC **debug list** command filtering is to use dmac = 6 bytes, smac = 6 bytes, dsap_offset = 12, and ssap_offset = 13.

Finally, you invoke the following **debug** commands:

```
Router# debug list 1103
Router# debug llc2 packet
LLC2 Packets debugging is on
for access list: 1103
```

To use the **debug list** command for Synchronous Data Link Control (SDLC) connections, with the exception of address 04, create access list 1102 to deny the specific address and permit all others:

```
access-list 1102 deny 0000.0000.0000 0000.0000.0000 0000.0000.0000 0000.0000.0000 0xC 1 eq
 0x4
access-list 1102 permit 0000.0000.0000 0000.0000.0000 0000.0000.0000 0000.0000.0000
```

The convention is to use dmac = 0.0.0, smac = 0.0.0, and sdlc_frame_offset = 12.

Invoke the following **debug** commands:

```
Router# debug list 1102
Router# debug sdlc
SDLC link debugging is on
for access list: 1102
```

To enable SDLC debugging (or debugging for any of the other supported protocols) for a specific interface rather than for all interfaces on a router, use the following commands:

```
Router# debug list serial 0
Router# debug sdlc
SDLC link debugging is on
for interface: Serial0
```

To enable Token Ring debugging between two MAC address, 0000.3018.4acd and 0000.30e0.8250, configure an extended bridging access list 1106:

```
access-list 1106 permit 0000.3018.4acd 8000.0000.0000 0000.30e0.8250 8000.0000.0000
access-list 1106 permit 0000.30e0.8250 8000.0000.0000 0000.3018.4acd 8000.0000.0000
```

Invoke the following **debug** commands:

```
Router# debug list 1106
Router# debug token ring
Token Ring Interface debugging is on
for access list: 1106
```

To enable routing information field (RIF) debugging for a single MAC address, configure an access list 1109:

```
access-list 1109 permit permit 0000.0000.0000 ffff.ffff.ffff 4000.2222.22c6 0000.0000.0000
```

Invoke the following **debug** commands:

```
Router# debug list 1109
Router# debug rif
RIF update debugging is on
for access list: 1109
```

Related Commands

Command	Description
debug llc2 errors	Displays LLC2 protocol error conditions or unexpected input.
debug llc2 packet	Displays all input and output from the LLC2 protocol stack.
debug llc2 state	Displays state transitions of the LLC2 protocol.
debug rif	Displays information on entries entering and leaving the RIF cache.
debug rtsp	Displays information on SDLC frames received and sent by any router serial interface involved in supporting SDLC end station functions.
debug token ring	Displays messages about Token Ring interface activity.

debug llc2 dynwind

To display changes to the dynamic window over Frame Relay, use the **debug llc2 dynwind**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug llc2 dynwind

no debug llc2 dynwind

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Examples

The following is sample output from the **debug llc2 dynwind** command:

```
Router# debug llc2 dynwind
LLC2/DW: BECN received! event REC_I_CMD, Window size reduced to 4
LLC2/DW: 1 consecutive I-frame(s) received without BECN
LLC2/DW: 2 consecutive I-frame(s) received without BECN
LLC2/DW: 3 consecutive I-frame(s) received without BECN
LLC2/DW: 4 consecutive I-frame(s) received without BECN
LLC2/DW: 5 consecutive I-frame(s) received without BECN
LLC2/DW: Current working window size is 5
```

In this example, the router receives a backward explicit congestion notification (BECN) and reduces the window size to 4. After receiving five consecutive I frames without a BECN, the router increases the window size to 5.

Related Commands

Command	Description
debug llc2 errors	Displays LLC2 protocol error conditions or unexpected input.
debug llc2 packet	Displays all input and output from the LLC2 protocol stack.
debug llc2 state	Displays state transitions of the LLC2 protocol.

debug llc2 errors

To display Logical Link Control, type 2 (LLC2) protocol error conditions or unexpected input, use the **debug llc2 errors**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug llc2 errors

no debug llc2 errors

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Examples

The following is sample output from the **debug llc2 errors** command from a router ignoring an incorrectly configured device:

```
Router# debug llc2 errors
LLC: admstate: 4000.1014.0001 0000.0000.0000 04 04 REC_RR_RSP
LLC: admstate: 4000.1014.0001 0000.0000.0000 04 04 REC_RR_RSP
LLC: admstate: 4000.1014.0001 0000.0000.0000 04 04 REC_RR_RSP
LLC: admstate: 4000.1014.0001 0000.0000.0000 04 04 REC_RR_RSP
LLC: admstate: 4000.1014.0001 0000.0000.0000 04 04 REC_RR_RSP
LLC: admstate: 4000.1014.0001 0000.0000.0000 04 04 REC_RR_RSP
```

Each line of output contains the remote MAC address, the local MAC address, the remote service access point (SAP), and the local SAP. In this example, the router receives unsolicited RR frames marked as responses.

Related Commands

Command	Description
debug list	Filters debugging information on a per-interface or per-access list basis.
debug llc2 dynwind	Displays changes to the dynamic window over Frame Relay.
debug llc2 packet	Displays all input and output from the LLC2 protocol stack.
debug llc2 state	Displays state transitions of the LLC2 protocol.

debug llc2 packet

To display all input and output from the Logical Link Control, type 2 (LLC2) protocol stack, use the **debug llc2 packet** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug llc2 packet

no debug llc2 packet

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

This command also displays information about some error conditions as well as internal interactions between the Common Link Services (CLS) layer and the LLC2 layer.

Examples

The following is sample output from the **debug llc2 packet** command from the router sending ping data back and forth to another router:

```
Router# debug llc2 packet
LLC: llc2_input
401E54F0:                            10400000              .@..
401E5500: 303A90CF 0006F4E1 2A200404 012B5E    0:.O..ta* ...+
LLC: i REC_RR_CMD N(R)=21 p/f=1
LLC: 0006.f4e1.2a20 0000.303a.90cf 04 04 NORMAL REC_RR_CMD (3)
LLC (rs): 0006.f4e1.2a20 0000.303a.90cf 04 04 REC_RR_CMD N(R)=42
LLC: 0006.f4e1.2a20 0000.303a.90cf 04 04 txmt RR_RSP N(R)=20 p/f=1
LLC: llc_sendframe
401E5610:              0040 0006F4E1 2A200000        .@..ta* ..
401E5620: 303A90CF 04050129 00               N  0:.O...).       2012
LLC: llc_sendframe
4022E3A0:                       0040 0006F4E1            .@..ta
4022E3B0: 2A200000 303A90CF 04042A28 2C000202  * ..0:.O..*(,...
4022E3C0: 00050B90 A02E0502 FF0003D1 004006C1  .... ......Q.@.A
4022E3D0: D7C9D5C    0.128
 C400130A C1D7D7D5 4BD5F2F0  WIUGD...AWWUKUrp
4022E3E0: F1F30000 011A6071 00010860 D7027000  qs....`q...`W.p.
4022E3F0: 00003B00 1112FF01 03000243 6973636F  ..;........Cisco
4022E400: 20494F53 69                           IOSi
LLC: 0006.f4e1.2a20 0000.303a.90cf 04 04 txmt I N(S)=21 N(R)=20 p/f=0 size=90
LLC: llc2_input
401E5620:                  10400000 303A90CF          .@..0:.O
401E5630: 0006F4E1 2A200404 282C2C00 02020004  ..ta* ..(,,.....
401E5640: 03902000 1112FF01 03000243 6973636F  .. ........Cisco
401E5650: 20494F53 A0                           IOS
LLC: i REC_I_CMD N(R)=22 N(S)=20 V(R)=20 p/f=0
LLC: 0006.f4e1.2a20 0000.303a.90cf 04 04 NORMAL REC_I_CMD (1)
LLC (rs): 0006.f4e1.2a20 0000.303a.90cf 04 04 REC_I_CMD N(S)=20 V(R)=20
LLC (rs): 0006.f4e1.2a20 0000.303a.90cf 04 04 REC_I_CMD N(R)=44
LLC: INFO: 0006.f4e1.2a20 0000.303a.90cf 04 04 v(r) 20
```

The first three lines indicate that the router has received some input from the link:

```
LLC: llc2_input
401E54F0:                            10400000              .@..
401E5500: 303A90CF 0006F4E1 2A200404 012B5E    0:.O..ta* ...+
```

The next line indicates that this input was an RR command with the poll bit set. The other router has received sequence number 21 and is waiting for the final bit.

```
LLC: i REC_RR_CMD N(R)=21 p/f=1
```

The next two lines contain the MAC addresses of the sender and receiver, and the state of the router when it received this frame:

```
LLC: 0006.f4e1.2a20 0000.303a.90cf 04 04 NORMAL REC_RR_CMD (3)
LLC (rs): 0006.f4e1.2a20 0000.303a.90cf 04 04 REC_RR_CMD N(R)=42
```

The next four lines indicate that the router is sending a response with the final bit set:

```
LLC: 0006.f4e1.2a20 0000.303a.90cf 04 04 txmt RR_RSP N(R)=20 p/f=1
LLC: llc_sendframe
401E5610:              0040 0006F4E1 2A200000        .@..ta* ..
401E5620: 303A90CF 04050129 00               N  0:.O...).       2012
```

Related Commands

Command	Description
debug list	Filters debugging information on a per-interface or per-access list basis.
debug llc2 dynwind	Displays changes to the dynamic window over Frame Relay.
debug llc2 errors	Displays LLC2 protocol error conditions or unexpected input.
debug llc2 state	Displays state transitions of the LLC2 protocol.

debug llc2 state

To display state transitions of the Logical Link Control, type 2 (LLC2) protocol, use the **debug llc2 state** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug llc2 state

no debug llc2 state

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

Refer to the ISO/IEC standard 8802-2 for definitions and explanations of **debug llc2 state** command output.

Examples

The following is sample output from the **debug llc2 state** command when a router disables and enables an interface:

```
Router# debug llc2 state
LLC (stsw): 0006.f4e1.2a20 0000.303a.90cf 04 04, NORMAL -> AWAIT (P_TIMER_EXP)
LLC(rs): 0006.f4e1.2a20 0000.303a.90cf 04 04, AWAIT -> D_CONN (P_TIMER_EXP)
LLC: cleanup 0006.f4e1.2a20 0000.303a.90cf 04 04, UNKNOWN (17)
LLC (stsw): 0006.f4e1.2a20 0000.303a.90cf 04 04, ADM -> SETUP (CONN_REQ)
LLC: normalstate: set_local_busy 0006.f4e1.2a20 0000.303a.90cf 04 04
LLC (stsw): 0006.f4e1.2a20 0000.303a.90cf 04 04, NORMAL -> BUSY (SET_LOCAL_BUSY)
LLC: Connection established: 0006.f4e1.2a20 0000.303a.90cf 04 04, success
LLC (stsw): 0006.f4e1.2a20 0000.303a.90cf 04 04, SETUP -> BUSY (SET_LOCAL_BUSY)
LLC: busystate: 0006.f4e1.2a20 0000.303a.90cf 04 04 local busy cleared
LLC (stsw): 0006.f4e1.2a20 0000.303a.90cf 04 04, BUSY -> NORMAL (CLEAR_LOCAL_BUSY)
```

Related Commands

Command	Description
debug list	Filters debugging information on a per-interface or per-access list basis.
debug llc2 dynwind	Displays changes to the dynamic window over Frame Relay.
debug llc2 errors	Displays LLC2 protocol error conditions or unexpected input.
debug llc2 packet	Displays all input and output from the LLC2 protocol stack.

debug lnm events

To display any unusual events that occur on a Token Ring network, use the **debug lnm events** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug lnm events

no debug lnm events

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

Unusual events include stations reporting errors or error thresholds being exceeded.

Examples

The following is sample output from the **debug lnm events** command:

```
Router# debug lnm events
IBMNM3: Adding 0000.3001.1166 to error list
IBMNM3: Station 0000.3001.1166 going into preweight condition
IBMNM3: Station 0000.3001.1166 going into weight condition
IBMNM3: Removing 0000.3001.1166 from error list
LANMGR0: Beaconing is present on the ring
LANMGR0: Ring is no longer beaconing
IBMNM3: Beaconing, Postmortem Started
IBMNM3: Beaconing, heard from 0000.3000.1234
IBMNM3: Beaconing, Postmortem Next Stage
IBMNM3: Beaconing, Postmortem Finished
```

The following message indicates that station 0000.3001.1166 reported errors and has been added to the list of stations reporting errors. This station is located on Ring 3.

```
IBMNM3: Adding 0000.3001.1166 to error list
```

The following message indicates that station 0000.3001.1166 has passed the “early warning” threshold for error counts:

```
IBMNM3: Station 0000.3001.1166 going into preweight condition
```

The following message indicates that station 0000.3001.1166 is experiencing a severe number of errors:

```
IBMNM3: Station 0000.3001.1166 going into weight condition
```

The following message indicates that the error counts for station 0000.3001.1166 have all decayed to zero, so this station is being removed from the list of stations that have reported errors:

```
IBMNM3: Removing 0000.3001.1166 from error list
```

The following message indicates that Ring 0 has entered failure mode. This ring number is assigned internally.

```
LANMGR0: Beaconing is present on the ring
```

The following message indicates that Ring 0 is no longer in failure mode. This ring number is assigned internally.

```
LANMGR0: Ring is no longer beaconing
```

The following message indicates that the router is beginning its attempt to determine whether any stations left the ring during the automatic recovery process for the last beaconing failure. The router attempts to contact stations that were part of the fault domain to detect whether they are still operating on the ring.

```
IBMNM3: Beaconing, Postmortem Started
```

The following message indicates that the router is attempting to determine whether any stations left the ring during the automatic recovery process for the last beaconing failure. It received a response from station 0000.3000.1234, one of the two stations in the fault domain.

```
IBMNM3: Beaconing, heard from 0000.3000.1234
```

The following message indicates that the router is attempting to determine whether any stations left the ring during the automatic recovery process for the last beaconing failure. It is initiating another attempt to contact the two stations in the fault domain.

```
IBMNM3: Beaconing, Postmortem Next Stage
```

The following message indicates that the router has attempted to determine whether any stations left the ring during the automatic recovery process for the last beaconing failure. It has successfully heard back from both stations that were part of the fault domain.

```
IBMNM3: Beaconing, Postmortem Finished
```

Explanations follow for other messages that the **debug lnm events** command can generate.

The following message indicates that the router is out of memory:

```
LANMGR: memory request failed, find_or_build_station()
```

The following message indicates that Ring 3 is experiencing a large number of errors that cannot be attributed to any individual station:

```
IBMNM3: Non-isolating error threshold exceeded
```

The following message indicates that a station (or stations) on Ring 3 is receiving frames faster than they can be processed:

```
IBMNM3: Adapters experiencing congestion
```

The following message indicates that the beaconing has lasted for over 1 minute and is considered a “permanent” error:

```
IBMNM3: Beaconing, permanent
```

The following message indicates that the beaconing lasted for less than 1 minute. The router is attempting to determine whether either station in the fault domain left the ring.

```
IBMNM: Beaconing, Destination Started
```

In the preceding line of output, the following can replace “Started”: “Next State,” “Finished,” “Timed out,” and “Cannot find station *n*.”

debug lnm llc

To display all comm unication between the router/bridge and the LAN Network Managers (LNMs) that have connections to it, use the **debug lnm llc** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug lnm llc

no debug lnm llc

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

One line is displayed for each message sent or received.

Examples

The following is sample output from the **debug lnm llc** command:

```
Router# debug lnm llc
IBMNM: Received LRM Set Reporting Point frame from 1000.5ade.0d8a.
IBMNM: found bridge: 001-2-00A, addresses: 0000.3040.a630 4000.3040.a630
IBMNM: Opening connection to 1000.5ade.0d8a on TokenRing0
IBMNM: Sending LRM LAN Manager Accepted to 1000.5ade.0d8a on link 0.
IBMNM: sending LRM New Reporting Link Established to 1000.5a79.dbf8 on link 1.
IBMNM: Determining new controlling LNM
IBMNM: Sending Report LAN Manager Control Shift to 1000.5ade.0d8a on link 0.
IBMNM: Sending Report LAN Manager Control Shift to 1000.5a79.dbf8 on link 1.
IBMNM: Bridge 001-2-00A received Request Bridge Status from 1000.5ade.0d8a.
IBMNM: Sending Report Bridge Status to 1000.5ade.0d8a on link 0.
IBMNM: Bridge 001-2-00A received Request REM Status from 1000.5ade.0d8a.
IBMNM: Sending Report REM Status to 1000.5ade.0d8a on link 0.
IBMNM: Bridge 001-2-00A received Set Bridge Parameters from 1000.5ade.0d8a.
IBMNM: Sending Bridge Parameters Set to 1000.5ade.0d8a on link 0.
IBMNM: sending Bridge Params Changed Notification to 1000.5a79.dbf8 on link 1.
IBMNM: Bridge 001-2-00A received Set REM Parameters from 1000.5ade.0d8a.
IBMNM: Sending REM Parameters Set to 1000.5ade.0d8a on link 0.
IBMNM: sending REM Parameters Changed Notification to 1000.5a79.dbf8 on link 1.
IBMNM: Bridge 001-2-00A received Set REM Parameters from 1000.5ade.0d8a.
IBMNM: Sending REM Parameters Set to 1000.5ade.0d8a on link 0.
IBMNM: sending REM Parameters Changed Notification to 1000.5a79.dbf8 on link 1.
IBMNM: Received LRM Set Reporting Point frame from 1000.5ade.0d8a.
IBMNM: found bridge: 001-1-00A, addresses: 0000.3080.2d79 4000.3080.2d7
```

As the output indicates, the **debug lnm llc** command output can vary somewhat in format.

The table below describes the significant fields shown in the display.

Table 99: debug lnm llc Field Descriptions

Field	Description
IBMNM:	Displays LLC-level debugging information.
Received	Router received a frame. The other possible value is Sending, to indicate that the router is sending a frame.

Field	Description
LRM	The function of the LLC-level software that is communicating as follows: • CRS--Configuration Report Server • LBS--LAN Bridge Server • LRM--LAN Reporting Manager • REM--Ring Error Monitor • RPS--Ring Parameter Server • RS--Ring Station
Set Reporting Point	Name of the specific frame that the router sent or received. Possible values include the following: • Bridge Counter Report • Bridge Parameters Changed Notification • Bridge Parameters Set • CRS Remove Ring Station • CRS Report NAUN Change • CRS Report Station Information • CRS Request Station Information • CRS Ring Station Removed • LRM LAN Manager Accepted • LRM Set Reporting Point • New Reporting Link Established • REM Forward MAC Frame • REM Parameters Changed Notification • REM Parameters Set • Report Bridge Status • Report LAN Manager Control Shift • Report REM Status • Request Bridge Status • Request REM Status • Set Bridge Parameters • Set REM Parameters

Field	Description
from 1000.5ade.0d8a	If the router has received the frame, this address is the source address of the frame. If the router is sending the frame, this address is the destination address of the frame.

The following message indicates that the lookup for the bridge with which the LAN Manager was requesting to communicate was successful:

```
IBMNM: found bridge: 001-2-00A, addresses: 0000.3040.a630 4000.3040.a630
```

The following message indicates that the connection is being opened:

```
IBMNM: Opening connection to 1000.5ade.0d8a on TokenRing0
```

The following message indicates that a LAN Manager has connected or disconnected from an internal bridge and that the router computes which LAN Manager is allowed to change parameters:

```
IBMNM: Determining new controlling LNM
```

The following line of output indicates which bridge in the router is the destination for the frame:

```
IBMNM: Bridge 001-2-00A received Request Bridge Status from 1000.5ade.0d8a.
```

debug lnm mac

To display all management communication between the router/bridge and all stations on the local Token Rings, use the **debug lnm mac** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug lnm mac

no debug lnm mac

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

One line is displayed for each message sent or received.

Examples

The following is sample output from the **debug lnm mac** command:

```
Router# debug lnm mac
LANMGR0: RS received request address from 4000.3040.a670.
LANMGR0: RS sending report address to 4000.3040.a670.
LANMGR0: RS received request state from 4000.3040.a670.
LANMGR0: RS sending report state to 4000.3040.a670.
LANMGR0: RS received request attachments from 4000.3040.a670.
LANMGR0: RS sending report attachments to 4000.3040.a670.
LANMGR2: RS received ring purge from 0000.3040.a630.
LANMGR2: CRS received report NAUN change from 0000.3040.a630.
LANMGR2: RS start watching ring poll.
LANMGR0: CRS received report NAUN change from 0000.3040.a630.
LANMGR0: RS start watching ring poll.
LANMGR2: REM received report soft error from 0000.3040.a630.
LANMGR0: REM received report soft error from 0000.3040.a630.
LANMGR2: RS received ring purge from 0000.3040.a630.
LANMGR2: RS received AMP from 0000.3040.a630.
LANMGR2: RS received SMP from 0000.3080.2d79.
LANMGR2: CRS received report NAUN change from 1000.5ade.0d8a.
LANMGR2: RS start watching ring poll.
LANMGR0: RS received ring purge from 0000.3040.a630.
LANMGR0: RS received AMP from 0000.3040.a630.
LANMGR0: RS received SMP from 0000.3080.2d79.
LANMGR0: CRS received report NAUN change from 1000.5ade.0d8a.
LANMGR0: RS start watching ring poll.
LANMGR2: RS received SMP from 1000.5ade.0d8a.
LANMGR2: RPS received request initialization from 1000.5ade.0d8a.
LANMGR2: RPS sending initialize station to 1000.5ade.0d8a.
```

The table below describes the significant fields shown in the display.

Table 100: debug lnm mac Field Descriptions

Field	Description
LANMGR0:	Indicates that this line of output displays MAC-level debugging information. 0 indicates the number of the Token Ring interface associated with this line of debugging output.
RS	Indicates which function of the MAC-level software is communicating as follows: • CRS--Configuration Report Server • REM--Ring Error Monitor • RPS--Ring Parameter Server • RS--Ring Station
received	Indicates that the router received a frame. The other possible value is sending, to indicate that the router is sending a frame.
request address	Indicates the name of the specific frame that the router sent or received. Possible values include the following: • AMP • initialize station • report address • report attachments • report nearest active upstream neighbor (NAUN) change • report soft error • report state • request address • request attachments • request initialization • request state • ring purge • SMP

Field	Description
from 4000.3040.a670	Indicates the source address of the frame, if the router has received the frame. If the router is sending the frame, this address is the destination address of the frame.

As the output indicates, all **debug lnm mac** command messages follow the format described in the table above except the following:

```
LANMGR2: RS start watching ring poll
LANMGR2: RS stop watching ring poll
```

These messages indicate that the router starts and stops receiving AMP and SMP frames. These frames are used to build a current picture of which stations are on the ring.

debug local-ack state

To display the new and the old state conditions whenever there is a state change in the local acknowledgment state machine, use the **debug loc al-ack state** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug local-ack state

no debug local-ack state

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Examples

The following is sample output from the **debug local-ack state** command:

```
Router# debug local-ack state
LACK_STATE: 2370300, hashp 2AE628, old state = disconn, new state = awaiting
LLC2 open to finish
LACK_STATE: 2370304, hashp 2AE628, old state = awaiting LLC2 open to finish,
new state = connected
LACK_STATE: 2373816, hashp 2AE628, old state = connected, new state = disconnected
LACK_STATE: 2489548, hashp 2AE628, old state = disconn, new state = awaiting
LLC2 open to finish
LACK_STATE: 2489548, hashp 2AE628, old state = awaiting LLC2 open to finish,
new state = connected
LACK_STATE: 2490132, hashp 2AE628, old state = connected, new state = awaiting
linkdown response
LACK_STATE: 2490140, hashp 2AE628, old state = awaiting linkdown response,
new state = disconnected
LACK_STATE: 2497640, hashp 2AE628, old state = disconn, new state = awaiting
LLC2 open to finish
LACK_STATE: 2497644, hashp 2AE628, old state = awaiting LLC2 open to finish,
new state = connected
```

The table below describes the significant fields shown in the display.

Table 101: debug local-ack state Field Descriptions

Field	Description
LACK_STATE:	Indicates that this packet describes a state change in the local acknowledgment state machine.
2370300	System clock.
hashp 2AE628	Internal control block pointer used by technical support staff for debugging purposes.

Field	Description
old state = disconn	Old state condition in the local acknowledgment state machine. Possible values include the following: • Disconn (disconnected) • awaiting LLC2 open to finish • connected • awaiting linkdown response
new state = awaiting LLC2 open to finish	New state condition in the local acknowledgment state machine. Possible values include the following: • Disconn (disconnected) • awaiting LLC2 open to finish • connected • awaiting linkdown response