

# debug qbm

To display debugging output for quality of service (QoS) bandwidth manager (QBM) options, use the **debug qbm** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

```
debug qbm {api | events}
```

```
no debug qbm {api | events}
```

## Syntax Description

<b>api</b>	Displays information about QBM client requests and notifications. See the “Usage Guidelines” section for additional information.
<b>events</b>	Displays information about QBM pool events.

## 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

Use the **debug qbm** command to troubleshoot QBM behavior.

Examples of client requests are when a client creates or destroys a bandwidth pool and when a client attempts to admit bandwidth into a pool. An example of a notification is when a client’s previously admitted bandwidth gets preempted from a pool.

## Examples

The following example shows how to enable the **debug qbm api** command:

```
Router# debug qbm api
QBM client requests and notifications debugging is on
```

The following example show how to enable the **debug qbm events** command:

```
Router# debug qbm events
QBM pool events debugging is on
```

The following example shows how to verify that QBM debugging is enabled:

```
Router# show debug
QoS Bandwidth Manager:
  QBM client requests and notifications debugging is on
  QBM pool events debugging is on
```

**Related Commands**

<b>Command</b>	<b>Description</b>
<b>show qbm client</b>	Displays registered QBM clients.
<b>show qbm pool</b>	Displays allocated QBM pools and associated objects.

# debug qlc error

To display quality link line control (QLLC) errors, use the **debug qlc error** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug qlc error**

**no debug qlc error**

---

**Syntax Description**

This command has no arguments or keywords.

---

**Command Modes**

Privileged EXEC (#)

---

**Usage Guidelines**

This command helps you track down errors in the QLLC interactions with X.25 networks. Use the **debug qlc error** command in conjunction with the **debug x25 all** command to see the connection. The data shown by this command only flows through the router on the X.25 connection. Some forms of this command can generate a substantial amount of output and network traffic.

---

**Examples**

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

```
Router# debug qlc error
```

```
%QLLC-3-GENERRMSG: qlc_close - bad qlc pointer Caller 00407116 Caller 00400BD2  
QLLC 4000.1111.0002: NO X.25 connection. Discarding XID and calling out
```

The following line indicates that the QLLC connection was closed:

```
%QLLC-3-GENERRMSG: qlc_close - bad qlc pointer Caller 00407116 Caller 00400BD2
```

The following line shows the virtual MAC address of the failed connection:

```
QLLC 4000.1111.0002: NO X.25 connection. Discarding XID and calling out
```

# debug qlc event

To enable debugging of quality link line control (QLLC) events, use the **debug qlc event** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug qlc event**

**no debug qlc event**

---

**Syntax Description**

This command has no arguments or keywords.

---

**Command Modes**

Privileged EXEC (#)

---

**Usage Guidelines**

Use the **debug qlc event** command to display primitives that might affect the state of a QLLC connection. An example of these events is the allocation of a QLLC structure for a logical channel indicator when an X.25 call has been accepted with the QLLC call user data. Other examples are the receipt and transmission of LAN explorer and exchange identification (XID) frames.

---

**Examples**

The following is sample output from the **debug qlc event** command:

```
Router# debug qlc event

QLLC: allocating new qlc lci 9
QLLC: tx POLLING TEST, da 4001.3745.1088, sa 4000.1111.0001
QLLC: rx explorer response, da 4000.1111.0001, sa c001.3745.1088, rif 08B0.1A91.1901.A040
QLLC: gen NULL XID, da c001.3745.1088, sa 4000.1111.0001, rif 0830.1A91.1901.A040, dsap 4,
ssap 4
QLLC: rx XID response, da 4000.1111.0001, sa c001.3745.1088, rif 08B0.1A91.1901.A040
```

The following line indicates that a new QLLC data structure has been allocated:

```
QLLC: allocating new qlc lci 9
```

The following lines show transmission and receipt of LAN explorer or test frames:

```
QLLC: tx POLLING TEST, da 4001.3745.1088, sa 4000.1111.0001
QLLC: rx explorer response, da 4000.1111.0001, sa c001.3745.1088, rif 08B0.1A91.1901.A040
```

The following lines show XID events:

```
QLLC: gen NULL XID, da c001.3745.1088, sa 4000.1111.0001, rif 0830.1A91.1901.A040, dsap 4,
ssap 4
QLLC: rx XID response, da 4000.1111.0001, sa c001.3745.1088, rif 08B0.1A91.1901.A040
```

# debug qlc packet

To display quality link line control (QLLC) events and QLLC data packets, use the **debug qlc packet** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug qlc packet**

**no debug qlc packet**

---

## Syntax Description

This command has no arguments or keywords.

---

## Command Modes

Privileged EXEC (#)

---

## Usage Guidelines

This command helps you to track down errors in the QLLC interactions with X.25 networks. The data shown by this command only flows through the router on the X25 connection. Use the **debug qlc packet** command in conjunction with the **debug x25 all** command to see the connection and the data that flows through the router.

---

## Examples

The following is sample output from the **debug qlc packet** command:

```
Router# debug qlc packet
```

```
14:38:05: Serial2/5 QLLC I: Data Packet.-RSP 9 bytes.  
14:38:07: Serial2/6 QLLC I: Data Packet.-RSP 112 bytes.  
14:38:07: Serial2/6 QLLC O: Data Packet. 128 bytes.  
14:38:08: Serial2/6 QLLC I: Data Packet.-RSP 9 bytes.  
14:38:08: Serial2/6 QLLC I: Data Packet.-RSP 112 bytes.  
14:38:08: Serial2/6 QLLC O: Data Packet. 128 bytes.  
14:38:08: Serial2/6 QLLC I: Data Packet.-RSP 9 bytes.  
14:38:12: Serial2/5 QLLC I: Data Packet.-RSP 112 bytes.  
14:38:12: Serial2/5 QLLC O: Data Packet. 128 bytes.
```

The following lines indicate that a packet was received on the interfaces:

```
14:38:05: Serial2/5 QLLC I: Data Packet.-RSP 9 bytes.  
14:38:07: Serial2/6 QLLC I: Data Packet.-RSP 112 bytes.
```

The following lines show that a packet was sent on the interfaces:

```
14:38:07: Serial2/6 QLLC O: Data Packet. 128 bytes.  
14:38:12: Serial2/5 QLLC O: Data Packet. 128 bytes.
```

# debug qlc state

To enable debugging of quality link line control (QLLC) events, use the **debug qlc state** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug qlc state**

**no debug qlc state**

---

**Syntax Description** This command has no arguments or keywords.

---

**Command Modes** Privileged EXEC (#)

---

**Usage Guidelines** Use the **debug qlc state** command to show when the state of a QLLC connection has changed. The typical QLLC connection goes from states ADM to SETUP to NORMAL. The NORMAL state indicates that a QLLC connection exists and is ready for data transfer.

---

**Examples** The following is sample output from the **debug qlc state** command:

```
Router# debug qlc state

Serial2 QLLC O: QSM-CMD
Serial2: X25 O D1 DATA (5) Q 8 lci 9 PS 4 PR 3
QLLC: state ADM -> SETUP
Serial2: X25 I D1 RR (3) 8 lci 9 PR 5
Serial2: X25 I D1 DATA (5) Q 8 lci 9 PS 3 PR 5
Serial2 QLLC I: QUA-RSPQLLC: addr 00, ctl 73

QLLC: qsetupstate: recvd qua rsp
QLLC: state SETUP -> NORMAL
```

The following line indicates that a QLLC connection attempt is changing state from ADM to SETUP:

```
QLLC: state ADM -> SETUP
```

The following line indicates that a QLLC connection attempt is changing state from SETUP to NORMAL:

```
QLLC: state SETUP -> NORMAL
```

# debug qlc timer

To display quality link line control (QLLC) timer events, use the **debug qlc timer** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug qlc timer**

**no debug qlc timer**

---

**Syntax Description**

This command has no arguments or keywords.

---

**Command Modes**

Privileged EXEC (#)

---

**Usage Guidelines**

The QLLC process periodically cycles and checks status of itself and its partner. If the partner is not found in the desired state, an LAPB primitive command is re-sent until the partner is in the desired state or the timer expires.

---

**Examples**

The following is sample output from the **debug qlc timer** command:

```
Router# debug qlc timer
```

```
14:27:24: Qllc timer lci 257, state ADM retry count 0 Caller 00407116 Caller 00400BD2  
14:27:34: Qllc timer lci 257, state NORMAL retry count 0  
14:27:44: Qllc timer lci 257, state NORMAL retry count 1  
14:27:54: Qllc timer lci 257, state NORMAL retry count 1
```

The following line of output shows the state of a QLLC partner on a given X.25 logical channel identifier:

```
14:27:24: Qllc timer lci 257, state ADM retry count 0 Caller 00407116 Caller 00400BD2
```

Other messages are informational and appear every ten seconds.

## debug qlc x25

To display X.25 packets that affect a quality link line control (QLLC) connection, use the **debug qlc x25** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

```
debug qlc x25
```

```
no debug qlc x25
```

---

**Syntax Description** This command has no arguments or keywords.

---

**Command Modes** Privileged EXEC (#)

---

**Usage Guidelines** This command is helpful to track down errors in the QLLC interactions with X.25 networks. Use the **debug qlc x25** command in conjunction with the **debug x25 events** or **debug x25 all** commands to see the X.25 events between the router and its partner.

---

**Examples** The following is sample output from the **debug qlc x25** command:

```
Router# debug qlc x25

15:07:23: QLLC X25 notify lci 257 event 1
15:07:23: QLLC X25 notify lci 257 event 5
15:07:34: QLLC X25 notify lci 257 event 3 Caller 00407116 Caller 00400BD2
15:07:35: QLLC X25 notify lci 257 event 4
```

[Table 293](#) describes the significant fields shown in the display.

**Table 293** *debug qlc x25 Field Descriptions*

Field	Description
15:07:23	Displays the time of day.
QLLC X25 notify 257	Indicates that this is a QLLC X25 message.
event <n>	Indicates the type of event, <i>n</i> . Values for <i>n</i> can be as follows: <ul style="list-style-type: none"> <li>1—Circuit is cleared</li> <li>2—Circuit has been reset</li> <li>3—Circuit is connected</li> <li>4—Circuit congestion has cleared</li> <li>5—Circuit has been deleted</li> </ul>

# debug qos ha

To debug quality of service (QoS) information on the networking device, use the **debug qos ha** command in privileged EXEC mode. To disable the display of debugging output, use the **no** form of this command.

**debug qos ha [detail]**

**no debug qos ha [detail]**

## Syntax Description

<b>detail</b>	(Optional) Displays detailed debug messages related to specified QoS information.
---------------	---

## 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.
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 to determine that QoS is running properly on your networking device.

## Examples

The following example enables QoS debugging:

```
Router# debug qos ha
```

# debug radius

To enable debugging for Remote Authentication Dial-In User Service (RADIUS) configuration, use the **debug radius** command in privileged EXEC mode. To disable debugging, use the **no** form of this command.

**debug radius** [**accounting** | **authentication** | **brief** | **elog** | **failover** | **retransmit** | **verbose**]

**no debug radius** [**accounting** | **authentication** | **brief** | **elog** | **failover** | **retransmit** | **verbose**]

## Syntax Description

<b>accounting</b>	(Optional) Enables debugging of RADIUS accounting collection.
<b>authentication</b>	(Optional) Enables debugging of RADIUS authentication packets.
<b>brief</b>	(Optional) Displays abbreviated debug output.
<b>elog</b>	(Optional) Enables RADIUS event logging.
<b>failover</b>	(Optional) Enables debugging of packets sent upon failover.
<b>retransmit</b>	(Optional) Enables retransmission of packets.
<b>verbose</b>	(Optional) Displays detailed debug output.

## Defaults

RADIUS event logging and debugging output in ASCII format are enabled.

## Command Modes

Privileged EXEC (#)

## Command History

Release	Modification
11.2(1)T	This command was introduced.
12.0(2)T	The <b>brief</b> keyword was added. The default output format became ASCII from hexadecimal.
12.2(11)T	The <b>verbose</b> keyword was added.
12.3(2)T	The <b>elog</b> keyword was added.
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

RADIUS is a distributed security system that secures networks against unauthorized access. Cisco supports RADIUS under the authentication, authorization, and accounting (AAA) security system. When RADIUS is used on the router, you can use the **debug radius** command to display debugging and troubleshooting information in ASCII format. Use the **debug radius brief** command for abbreviated output displaying client/server interaction and minimum packet information. Only the input and output transactions are recorded. Use the **debug radius verbose** command to include non-essential RADIUS debugs.

**Examples**

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

```

Router# debug radius

Radius protocol debugging is on
Radius packet hex dump debugging is off

Router# show debug

00:19:20: RADIUS/ENCODE(00000015):Orig. component type = AUTH_PROXY
00:19:20: RADIUS(00000015): Config NAS IP: 0.0.0.0
00:19:20: RADIUS/ENCODE(00000015): acct_session_id: 21
00:19:20: RADIUS(00000015): sending
00:19:20: RADIUS/ENCODE: Best Local IP-Address 33.0.0.2 for Radius-Server 33.2.0.1
00:19:20: RADIUS(00000015): Send Access-Request to 33.2.0.1:1645 id 1645/21, len 159
00:19:20: RADIUS: authenticator 2D 03 E5 A6 A5 30 1A 32 - F2 C5 EE E2 AC 5E 5D 22
00:19:20: RADIUS: User-Name [1] 11 "authproxy"
00:19:20: RADIUS: User-Password [2] 18 *
00:19:20: RADIUS: Service-Type [6] 6 Outbound [5]
00:19:20: RADIUS: Message-Authenticato[80] 18
00:19:20: RADIUS: 85 EF E8 43 03 88 58 63 78 D2 7B E7 26 61 D3 3C [ CXcx{&a<]
00:19:20: RADIUS: Vendor, Cisco [26] 49
00:19:20: RADIUS: Cisco AVpair [1] 43
"audit-session-id=0D00000200000013001112FD"
00:19:20: RADIUS: NAS-Port-Type [61] 6 Ethernet [15]
00:19:20: RADIUS: NAS-Port [5] 6 16480
00:19:20: RADIUS: NAS-Port-Id [87] 19 "FastEthernet1/0/3"
00:19:20: RADIUS: NAS-IP-Address [4] 6 33.0.0.2
00:19:20: RADIUS(00000015): Started 5 sec timeout
00:19:20: RADIUS: Received from id 1645/21 33.2.0.1:1645, Access-Accept, len 313
00:19:20: RADIUS: authenticator E6 6E 1D 64 5A 15 FD AE - C9 60 C0 68 F5 10 E9 B7
00:19:20: RADIUS: Filter-Id [11] 8
00:19:20: RADIUS: 31 30 30 2E 69 6E [ 100.in]
00:19:20: RADIUS: Vendor, Cisco [26] 19
00:19:20: RADIUS: Cisco AVpair [1] 13 "priv-lvl=15"
00:19:20: RADIUS: Termination-Action [29] 6 1
00:19:20: RADIUS: Vendor, Cisco [26] 45
00:19:20: RADIUS: Cisco AVpair [1] 39 "supplicant-name=Port-description test"
00:19:20: RADIUS: Vendor, Cisco [26] 38
00:19:20: RADIUS: Cisco AVpair [1] 32 "security-group-tag=2468-C0FFEE"
00:19:20: RADIUS: Vendor, Cisco [26] 33
00:19:20: RADIUS: Cisco AVpair [1] 27 "supplicant-group=engineer"
00:19:20: RADIUS: Vendor, Cisco [26] 36
00:19:20: RADIUS: Cisco AVpair [1] 30 "supplicant-group=idf_testing"
00:19:20: RADIUS: Vendor, Cisco [26] 28
00:19:20: RADIUS: Cisco AVpair [1] 22 "authz-directive=open"
00:19:20: RADIUS: Vendor, Cisco [26] 32
00:19:20: RADIUS: Cisco AVpair [1] 26 "supplicant-group=group-9"
00:19:20: RADIUS: Class [25] 30
00:19:20: RADIUS: 43 41 43 53 3A 63 2F 61 37 31 38 38 61 2F 32 31 [CACS:c/a7188a/21]
00:19:20: RADIUS: 30 30 30 30 30 32 2F 31 36 34 38 30 [ 000002/16480]
00:19:20: RADIUS: Message-Authenticato[80] 18
00:19:20: RADIUS: 24 13 29 95 A1 5E 9F D3 CB ED 78 F1 F6 62 2B E3 [ $)^xb+]
00:19:20: RADIUS(00000015): Received from id 1645/21
00:19:20: RADIUS/DECODE: parse unknown cisco vsa "supplicant-group" - IGNORE
00:19:20: RADIUS/DECODE: parse unknown cisco vsa "supplicant-group" - IGNORE
00:19:20: RADIUS/DECODE: parse unknown cisco vsa "authz-directive" - IGNORE
00:19:20: RADIUS/DECODE: parse unknown cisco vsa "supplicant-group" - IGNORE
00:19:20: RADIUS/ENCODE(00000015):Orig. component type = AUTH_PROXY
00:19:20: RADIUS(00000015): Config NAS IP: 0.0.0.0
00:19:20: RADIUS(00000015): sending
00:19:20: RADIUS/ENCODE: Best Local IP-Address 33.0.0.2 for Radius-Server 33.2.0.1
00:19:20: RADIUS(00000015): Send Accounting-Request to 33.2.0.1:1646 id 1646/1, len 204
00:19:20: RADIUS: authenticator A7 6B A0 94 F4 63 30 51 - 8A CE 8C F4 8A 8E 0B CC

```

```

00:19:20: RADIUS: Acct-Session-Id [44] 10 "00000015"
00:19:20: RADIUS: Calling-Station-Id [31] 10 "13.1.0.1"
00:19:20: RADIUS: Vendor, Cisco [26] 49

00:19:20: RADIUS: Cisco AVpair [1] 43
"audit-session-id=0D00000200000013001112FD"

```

The following is sample output from the **debug radius brief** command:

```
Router# debug radius brief
```

```

Radius protocol debugging is on
Radius packet hex dump debugging is off
Radius protocol in brief format debugging is on
00:05:21: RADIUS: Initial Transmit ISDN 0:D:23 id 6 10.0.0.1:1824, Accounting-Request, len
358
00:05:21: %ISDN-6-CONNECT: Interface Serial0:22 is now connected to 4085274206
00:05:26: RADIUS: Retransmit id 6
00:05:31: RADIUS: Tried all servers.
00:05:31: RADIUS: No valid server found. Trying any viable server
00:05:31: RADIUS: Tried all servers.
00:05:31: RADIUS: No response for id 7
00:05:31: RADIUS: Initial Transmit ISDN 0:D:23 id 8 10.0.0.0:1823, Access-Request, len 171
00:05:36: RADIUS: Retransmit id 8
00:05:36: RADIUS: Received from id 8 1.7.157.1:1823, Access-Accept, len 115
00:05:47: %ISDN-6-DISCONNECT: Interface Serial0:22 disconnected from 4085274206, call
lasted 26 seconds
00:05:47: RADIUS: Initial Transmit ISDN 0:D:23 id 9 10.0.0.1:1824, Accounting-Request, len
775
00:05:47: RADIUS: Received from id 9 1.7.157.1:1824, Accounting-response, len 20

```

The following example shows how to enable debugging of RADIUS accounting collection:

```
Router# debug radius accounting
```

```

Radius protocol debugging is on
Radius protocol brief debugging is off
Radius protocol verbose debugging is off
Radius packet hex dump debugging is off
Radius packet protocol (authentication) debugging is off
Radius packet protocol (accounting) debugging is on
Radius packet retransmission debugging is off
Radius server fail-over debugging is off
Radius elog debugging is off

```

## Related Commands

Command	Description
<b>debug aaa accounting</b>	Displays information on accountable events as they occur.
<b>debug aaa authentication</b>	Displays information on AAA/TACACS+ authentication.

# debug radius local-server

To control the display of debug messages for the local authentication server, use the **debug radius local-server** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

```
debug radius local-server {client | error | packets}
```

```
no debug radius local-server {client | error | packets}
```

## Syntax Description

<b>client</b>	Displays error messages about failed client authentications.
<b>error</b>	Displays error messages about the local authentication server.
<b>packets</b>	Displays the content of the RADIUS packets that are sent and received.

## Defaults

No default behavior or values

## Command Modes

Privileged EXEC (#)

## Command History

Release	Modification
12.2(11)JA	This command was introduced on Cisco Aironet Access Point 1200 and Cisco Aironet Access Point 1100.
12.3(11)T	This command was implemented on the following platforms: Cisco 2600XM, Cisco 2691, Cisco 2811, Cisco 2821, Cisco 2851, Cisco 3700, and Cisco 3800 series routers.
12.4(2)T	This command was integrated into Cisco IOS Release 12.4(2)T.
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 control the display of debug messages for the local authentication server.

## Examples

The following command shows how to display messages regarding failed client authentication:

```
Router# debug radius local-server client
```

## Related Commands

Command	Description
<b>clear radius local-server</b>	Clears the statistics display or unblocks a user.
<b>show radius local-server statistics</b>	Displays statistics for a local network access server.
<b>ssid</b>	Specifies up to 20 SSIDs to be used by a user group.

<b>Command</b>	<b>Description</b>
<b>user</b>	Authorizes a user to authenticate using the local authentication server.
<b>vlan</b>	Specifies a VLAN to be used by members of a user group.

# debug radius-proxy

To display debugging messages for Intelligent Service Gateway (ISG) RADIUS proxy functionality, use the **debug radius-proxy** command in privileged EXEC mode. To disable debugging, use the **no** form of this command.

```
debug radius-proxy {events | errors}
```

```
no debug radius-proxy {events | errors}
```

## Syntax Description

<b>events</b>	Displays debug messages related to ISG RADIUS proxy events.
<b>errors</b>	Displays debug messages related to ISG RADIUS proxy errors.

## Command Modes

Privileged EXEC

## Command History

Release	Modification
12.2(31)SB2	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

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, only use **debug** commands 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.

## Examples

The following example shows output for the **debug radius-proxy** command with the **events** keyword:

```
Router# debug radius-proxy events

*Nov 7 07:53:11.411: RP-EVENT: Parse Request: Username = 12345679@cisco
*Nov 7 07:53:11.411: RP-EVENT: Parse Request: Caller ID = 12345679@cisco
*Nov 7 07:53:11.411: RP-EVENT: Parse Request: NAS id = localhost
*Nov 7 07:53:11.411: RP-EVENT: Found matching context for user Caller ID:12345679@cisco
Name:aa
*Nov 7 07:53:11.411: RP-EVENT: Received event client Access-Request in state activated
*Nov 7 07:53:11.411: RP-EVENT: User Caller ID:12345679@cisco Name:12 re-authenticating
*Nov 7 07:53:11.411: RP-EVENT: Forwarding Request to method list (handle=1979711512)
*Nov 7 07:53:11.411: RP-EVENT: Sending request to server group EAP
*Nov 7 07:53:11.411: RP-EVENT: State changed activated --> wait for Access-Response
```

# debug rai

To enable debugging for Resource Allocation Indication (RAI), use the **debug rai** command in privileged EXEC mode. To disable debugging for RAI, use the **no** form of this command.

**debug rai**

**no debug rai**

**Syntax Description** This command has no arguments or keywords.

**Command Default** Debugging is disabled.

**Command Modes** Privileged EXEC (#)

## Command History

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

## Usage Guidelines

You can use the **debug rai** command along with the **debug ccsip all** command to get the complete debugging information for RAI.

## Examples

The following example shows how to enable resource allocation debugging:

```
Router# debug rai

Resource Availability debugging is on

*Dec 16 05:50:34.863: //1/rai_new_resource_index:New index created 1
*Dec 16 05:50:34.863: //1/rai_main:- event code:7
*Dec 16 05:50:34.863: //1/rai_process_new_rsc_group:New Resource Index created
*Dec 16 05:50:34.907: //1/rai_set_resource_info_config:Resource type 0
*Dec 16 05:50:34.907: //1/rai_set_resource_info_config:New system resource created
0x4961A38C
*Dec 16 05:50:34.907: //1/rai_set_resource_info_config:Resource New Config Event passed
*Dec 16 05:50:34.907: //1/rai_set_resource_info_config:Resource Type CPU Subtype 1-min-avg
Low watermark 30High watermark 50
*Dec 16 05:50:34.907: //1/rai_main:- event code:4
```

## Related Commands

Command	Description
<b>periodic-report interval</b>	Configures periodic reporting parameters for gateway resource entities.
<b>rai target</b>	Configures the SIP RAI mechanism.
<b>resource (voice)</b>	Configures parameters for monitoring resources, use the resource command in voice-class configuration mode.

<b>Command</b>	<b>Description</b>
<b>show voice class resource-group</b>	Displays the resource group configuration information for a specific resource group or all resource groups.
<b>voice class resource-group</b>	Enters voice-class configuration mode and assigns an identification tag number for a resource group.

# debug ras

To display the types and addressing of Registration, Admission and Status (RAS) messages sent and received, use the **debug ras** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug ras**

**no debug ras**

## Syntax Description

This command has arguments or keywords.

## Defaults

This command is disabled.

## Command Modes

Privileged EXEC

## Command History

Release	Modification
11.3(6)NA2	This command was introduced.
12.2(2)XB1	This command was implemented on the Cisco AS5850 universal access router.
12.2(11)T	This command was integrated into Cisco IOS Release 12.2(11)T.
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 ras** command to display the types and addressing of RAS messages sent and received. The debug output lists the message type using mnemonics defined in International Telecommunications Union-Telecommunication (ITU-T) specification H.225.

## Examples

In the following output, gateway GW13.cisco.com sends a RAS registration request (RRQ) message to gatekeeper GK15.cisco.com at IP address 10.9.53.15. GW13.cisco.com then receives a registration confirmation (RCF) message from the gatekeeper. If there is no response, it could mean that the gatekeeper is offline or improperly addressed. If you receive a reject (RRJ) message, it could mean that the gatekeeper is unable to handle another gateway or that the registration information is incorrect.

```
Router# debug ras

*Mar 13 19:53:34.231:      RASLib::ras_sendto:msg length 105 from
                        10.9.53.13:8658 to 10.9.53.15:1719
*Mar 13 19:53:34.231:      RASLib::RASSendRRQ:RRQ (seq# 36939) sent
                        to 10.9.53.15
*Mar 13 19:53:34.247:      RASLib::RASRecvData:successfully rcvd
                        message of length 105 from 10.9.53.15:1719
*Mar 13 19:53:34.251:      RASLib::RASRecvData:RCF (seq# 36939) rcvd
                        from [10.9.53.15:1719] on sock [0x6168356C]
```

# debug redundancy application group config

To display the redundancy application group configuration, use the **debug redundancy application group config** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

```
debug redundancy application group config {all | error | event | func}
```

```
no debug redundancy application group config {all | error | event | func}
```

## Syntax Description

<b>all</b>	Displays debug information about the configuration.
<b>error</b>	Displays information about the redundancy group's configuration errors.
<b>event</b>	Displays information about the redundancy group's configuration.
<b>func</b>	Displays information about the redundancy group's configuration functions entered.

## Command Modes

Privileged EXEC (#)

## Command History

Release	Modification
Cisco IOS XE Release 3.1S	This command was introduced.

## Examples

The following is sample output from the **debug redundancy application group config all** command:

```
Router# debug redundancy application group config all
```

```
RG config all debugging is on
```

## Related Commands

Command	Description
<b>debug redundancy application group media</b>	Displays the redundancy application group media information.
<b>debug redundancy application group protocol</b>	Displays the redundancy application group protocol information.
<b>debug redundancy application group rii</b>	Displays the redundancy application group RII information.
<b>debug redundancy application group transport</b>	Displays the redundancy application group transport information.
<b>debug redundancy application group vp</b>	Displays the redundancy application group VP information.

# debug redundancy application group faults

To display the redundancy application group faults, use the **debug redundancy application group faults** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug redundancy application group faults {all | error | event | fault | func}**

**no debug redundancy application group faults {all | error | event | fault | func}**

## Syntax Description

<b>all</b>	Displays fault information of a redundancy group.
<b>error</b>	Displays error information of a redundancy groups.
<b>event</b>	Displays event information of a redundancy group.
<b>fault</b>	Displays fault events information of a redundancy group.
<b>func</b>	Displays fault functions information of a redundancy group.

## Command Modes

Privileged EXEC (#)

## Command History

Release	Modification
Cisco IOS XE Release 3.1S	This command was introduced.

## Examples

The following is sample output from the **debug redundancy application group faults error** command:

```
Router# debug redundancy application group faults error
```

```
RG Faults error debugging is on
```

## Related Commands

Command	Description
<b>debug redundancy application group config</b>	Displays the redundancy application group configuration.
<b>debug redundancy application group media</b>	Displays the redundancy application group media information.
<b>debug redundancy application group protocol</b>	Displays the redundancy application group protocol information.
<b>debug redundancy application group rii</b>	Displays the redundancy application group RII information.

<b>Command</b>	<b>Description</b>
<b>debug redundancy application group transport</b>	Displays the redundancy group application group transport information.
<b>debug redundancy application group vp</b>	Displays the redundancy group application group VP information.

# debug redundancy application group media

To display the redundancy application group media information, use the **debug redundancy application group media** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug redundancy application group media** {all | error | event | nbr | packet {rx | tx} | timer }

**no debug redundancy application group media** {all | error | event | nbr | packet {rx | tx} | timer }

## Syntax Description

<b>all</b>	Displays media information of a redundancy group.
<b>error</b>	Displays media error information of a redundancy group.
<b>event</b>	Displays media events information of a redundancy group.
<b>nbr</b>	Displays media neighbor (nbr) information of a redundancy group.
<b>packet</b>	Displays media packets information of a redundancy group.
<b>rx</b>	Displays the incoming packets information.
<b>tx</b>	Displays the outgoing packets information.
<b>timer</b>	Displays information about redundancy group media timer events.

## Command Modes

Privileged EXEC (#)

## Command History

Release	Modification
Cisco IOS XE Release 3.1S	This command was introduced.

## Examples

The following is sample output from the **debug redundancy application group media timer** command:

```
Router# debug redundancy application group media timer
```

```
RG Media timer debugging is on
```

## Related Commands

Command	Description
<b>debug redundancy application group config</b>	Displays the redundancy group application configuration.
<b>debug redundancy application group protocol</b>	Displays the redundancy group application group protocol information.
<b>debug redundancy application group rii</b>	Displays the redundancy group application group RII information.

<b>Command</b>	<b>Description</b>
<b>debug redundancy application group transport</b>	Displays the redundancy group application group transport information.
<b>debug redundancy application group vp</b>	Displays the redundancy application group VP information.

# debug redundancy application group protocol

To display the redundancy application group protocol information, use the **debug redundancy application group protocol** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

```
debug redundancy application group protocol {all | detail | error | event | media | peer}
```

```
no debug redundancy application group protocol {all | detail | error | event | media | peer}
```

## Syntax Description

<b>all</b>	Displays protocol information of a redundancy group.
<b>detail</b>	Displays event details of a redundancy group.
<b>error</b>	Displays protocol error information of a redundancy group.
<b>event</b>	Displays protocol events information of a redundancy group.
<b>media</b>	Displays protocol media events information of a redundancy group.
<b>peer</b>	Displays protocol peer information of a redundancy group.

## Command Modes

Privileged EXEC (#)

## Command History

Release	Modification
Cisco IOS XE Release 3.1S	This command was introduced.

## Examples

The following is sample output from the **debug redundancy application group protocol peer** command:

```
Router# debug redundancy application group protocol peer

RG Protocol peer debugging is on
```

## Related Commands

Command	Description
<b>debug redundancy application group config</b>	Displays the redundancy group application configuration.
<b>debug redundancy application group media</b>	Displays the redundancy application group media information.
<b>debug redundancy application group protocol</b>	Displays the redundancy application group protocol information.
<b>debug redundancy application group rri</b>	Displays the redundancy application group RII information.

<b>Command</b>	<b>Description</b>
<b>debug redundancy application group transport</b>	Displays the redundancy application group transport information.
<b>debug redundancy application group vp</b>	Displays the redundancy application group VP information.

# debug redundancy application group rii

To display the redundancy application group redundancy interface identifier (RII) information, use the **debug redundancy application group rii** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

```
debug redundancy application group rii {error | event}
```

```
no debug redundancy application group rii {error | event}
```

## Syntax Description

<b>error</b>	Displays RII error information of a redundancy group.
<b>event</b>	Displays RII event information of a redundancy group.

## Command Modes

Privileged EXEC (#)

## Command History

Release	Modification
Cisco IOS XE Release 3.1S	This command was introduced.

## Examples

The following is sample output from the **debug redundancy application group rii event** command:

```
Router# debug redundancy application group rii event
RG RII events debugging is on
```

## Related Commands

Command	Description
<b>debug redundancy application group config</b>	Displays the redundancy group application configuration.
<b>debug redundancy application group media</b>	Displays the redundancy application group media information.
<b>debug redundancy application group protocol</b>	Displays the redundancy group application group protocol information.
<b>debug redundancy application group vp</b>	Displays the redundancy group application group VP information.

# debug redundancy application group transport

To display the redundancy application group transport information, use the **debug redundancy application group transport** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

```
debug redundancy application group transport {db | error | event | packet | timer | trace}
```

```
no debug redundancy application group transport {db | error | event | packet | timer | trace}
```

## Syntax Description

<b>db</b>	Displays transport information of a redundancy group.
<b>error</b>	Displays transport error information of a redundancy group.
<b>event</b>	Displays transport event information of a redundancy group.
<b>packet</b>	Displays transport packet information of a redundancy group.
<b>timer</b>	Displays transport timer information of a redundancy group.
<b>trace</b>	Displays transport trace information of a redundancy group.

## Command Modes

Privileged EXEC (#)

## Command History

Release	Modification
Cisco IOS XE Release 3.1S	This command was introduced.

## Examples

The following is sample output from the **debug redundancy application group transport trace** command:

```
Router# debug redundancy application group transport trace

RG Transport trace debugging is on
```

## Related Commands

Command	Description
<b>debug redundancy application group config</b>	Displays the redundancy group application configuration.
<b>debug redundancy application group media</b>	Displays the redundancy application group media information.
<b>debug redundancy application group protocol</b>	Displays the redundancy application group protocol information.
<b>debug redundancy application group rii</b>	Displays the redundancy application group RII information.

# debug redundancy application group vp

To display the redundancy application group virtual platform (VP) information, use the **debug redundancy application group vp** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

```
debug redundancy application group vp {error | event}
```

```
no debug redundancy application group vp {error | event}
```

## Syntax Description

<b>error</b>	Displays VP error information of a redundancy group.
<b>event</b>	Displays VP event information of a redundancy group.

## Command Modes

Privileged EXEC (#)

## Command History

Release	Modification
Cisco IOS XE Release 3.1S	This command was introduced.

## Examples

The following is sample output from the **debug redundancy application group vp event** command:

```
Router# debug redundancy application group vp event

RG VP events debugging is on
```

## Related Commands

Command	Description
<b>debug redundancy application group config</b>	Displays the redundancy group application configuration.
<b>debug redundancy application group media</b>	Displays the redundancy application group media information.
<b>debug redundancy application group protocol</b>	Displays the redundancy application group protocol information.
<b>debug redundancy application group rii</b>	Displays the redundancy application group RII information.
<b>debug redundancy application group transport</b>	Displays the redundancy application group transport information.

# debug redundancy (RP)

To enable the display of events for troubleshooting dual Route Processors (RPs), use the **debug redundancy** command in privileged EXEC mode. To disable the display of debugging output, use the **no** form of this command.

```
debug redundancy {ehsa | errors | fsm | kpa | msg | progression | status | timer}
```

```
no debug redundancy {ehsa | errors | fsm | kpa | msg | progression | status | timer}
```

## Syntax Description

<b>ehsa</b>	Displays redundancy facility (RF) enhanced high system availability (EHSA) information.
<b>errors</b>	Displays RF errors.
<b>fsm</b>	Displays RF feasible successor metrics (FSM) events.
<b>kpa</b>	Displays RF keepalive events.
<b>msg</b>	Displays RF messaging events.
<b>progression</b>	Displays RF progression events.
<b>status</b>	Displays RF status events.
<b>timer</b>	Displays RF timer events.

## Command Modes

Privileged EXEC

## Command History

Release	Modification
11.3(6)AA	This command was introduced.
12.0(15)ST	This command was introduced on Cisco 10000 series Internet routers.
12.0(22)S	This command was introduced on Cisco 7500, 10000, and 12000 series Internet routers.
12.2(18)S	This command was integrated into Cisco IOS Release 12.2(18)S on Cisco 7500 series routers.
12.2(20)S	Support was added for the Cisco 7304 router. The Cisco 7500 series router is not supported in Cisco IOS Release 12.2(20)S.
12.2(28)SB	Support for 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.
12.2SX	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 example enables debugging information for RF keepalive events:

```
Router# debug redundancy kpa
```

# debug redundancy application group config

To display the redundancy application group configuration, use the **debug redundancy application group config** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

```
debug redundancy application group config {all | error | event | func}
```

```
no debug redundancy application group config {all | error | event | func}
```

## Syntax Description

<b>all</b>	Displays debug information about configuration.
<b>error</b>	Displays information about the redundancy group's configuration errors.
<b>event</b>	Displays information about the redundancy group's configuration .
<b>func</b>	Displays information about the redundancy group's configuration functions entered.

## Command Modes

Privileged EXEC (#)

## Command History

Release	Modification
Cisco IOS XE Release 3.1S	This command was introduced.

## Examples

The following is sample output from the **debug redundancy application group config all** command:

```
Router# debug redundancy application group config all
```

```
RG config all debugging is on
```

## Related Commands

Command	Description
<b>debug redundancy application group media</b>	Displays the redundancy application group media information.
<b>debug redundancy application group protocol</b>	Displays the redundancy application group protocol information.
<b>debug redundancy application group RII</b>	Displays the redundancy application group RII information.
<b>debug redundancy application group transport</b>	Displays the redundancy application group transport information.
<b>debug redundancy application group VP</b>	Displays the redundancy application group VP information.

# debug redundancy application group faults

To display the redundancy application group faults, use the **debug redundancy application group faults** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

```
debug redundancy application group faults {all | error | event | fault | func}
```

```
no debug redundancy application group faults {all | error | event | fault | func}
```

## Syntax Description

<b>all</b>	Displays fault information of a redundancy group.
<b>error</b>	Displays error information of a redundancy groups.
<b>event</b>	Displaysevent information of a redundancy group.
<b>fault</b>	Displays fault events information of a redundancy group.
<b>func</b>	Displays fault functions information of a redundancy group.

## Command Modes

Privileged EXEC (#)

## Command History

Release	Modification
Cisco IOS XE Release 3.1S	This command was introduced.

## Examples

The following is sample output from the **debug redundancy application group faults error** command:

```
Router# debug redundancy application group faults error
```

```
RG Faults error debugging is on
```

## Related Commands

Command	Description
<b>redundancy application group config</b>	display the redundancy application group configuration.
<b>debug redundancy application group media</b>	Displays the redundancy application group media information.
<b>debug redundancy application group protocol</b>	Displays the redundancy application group protocol information.
<b>debug redundancy application group rii</b>	Displays the redundancy application group RII information.

<b>Command</b>	<b>Description</b>
<b>debug redundancy application group transport</b>	Displays the redundancy group application group transport information.
<b>debug redundancy application group vp</b>	Displays the redundancy group application group VP information.

# debug redundancy application group media

To display the redundancy application group media information, use the **debug redundancy application group media** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

```
debug redundancy application group media {all | error | event | nbr | packet {rx | tx} | timer}
```

```
no debug redundancy application group media {all | error | event | nbr | packet {rx | tx} | timer}
```

## Syntax Description

<b>all</b>	Displays media information of a redundancy group.
<b>error</b>	Displays media errors information of a redundancy group.
<b>event</b>	Displays media events information of a redundancy group.
<b>nbr</b>	Displays media neighbor (nbr) information of a redundancy group.
<b>packet</b>	Displays media packets information of a redundancy group.
<b>rx</b>	Displays the incoming packets information.
<b>tx</b>	Displays the outgoing packets information.
<b>timer</b>	Displays media timer events information about redundancy group media timer events.

## Command Modes

Privileged EXEC (#)

## Command History

Release	Modification
Cisco IOS XE Release 3.1S	This command was introduced.

## Examples

The following is sample output from the **debug redundancy application group media timer** command:

```
Router# debug redundancy application group media timer

RG Media timer debugging is on
```

## Related Commands

Command	Description
<b>redundancy application group config</b>	Displays the redundancy group application configuration.
<b>debug redundancy application group protocol</b>	Displays the redundancy group application group protocol information.
<b>debug redundancy application group rii</b>	Displays the redundancy group application group RII information.

<b>Command</b>	<b>Description</b>
<b>debug redundancy application group transport</b>	Displays the redundancy group application group transport information.
<b>debug redundancy application group vp</b>	Displays the redundancy application group VP information.
<b>redundancy application group config</b>	Displays the redundancy group application configuration.
<b>debug redundancy application group media</b>	Displays the redundancy application group media information.
<b>debug redundancy application group protocol</b>	Displays the redundancy application group protocol information.

# debug redundancy application group protocol

To display the redundancy application group protocol information, use the **debug redundancy application group protocol** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

```
debug redundancy application group protocol {all | detail | error | event | media | peer}
```

```
no debug redundancy application group protocol {all | detail | error | event | media | peer}
```

## Syntax Description

<b>all</b>	Displays protocol information of a redundancy group.
<b>detail</b>	Displays event details of a redundancy group.
<b>error</b>	Displays protocol error information of a redundancy group.
<b>event</b>	Displays protocol events information of a redundancy group.
<b>media</b>	Displays protocol media events information of a redundancy group.
<b>peer</b>	Displays protocol peer information of a redundancy group.

## Command Modes

Privileged EXEC (#)

## Command History

Release	Modification
Cisco IOS XE Release 3.1S	This command was introduced.

## Examples

The following is sample output from the **debug redundancy application group protocol peer** command:

```
Router# debug redundancy application group protocol peer

RG Protocol peer debugging is on
```

## Related Commands

Command	Description
<b>redundancy application group config</b>	Displays the redundancy group application configuration.
<b>debug redundancy application group protocol</b>	Displays the redundancy application group protocol information.
<b>debug redundancy application group rii</b>	Displays the redundancy application group RII information.
<b>debug redundancy application group transport</b>	Displays the redundancy application group transport information.

<b>Command</b>	<b>Description</b>
<b>debug redundancy application group vp</b>	Displays the redundancy application group VP information.
<b>redundancy application group config</b>	Displays the redundancy group application configuration.
<b>debug redundancy application group media</b>	Displays the redundancy application group media information.
<b>redundancy application group config</b>	Displays the redundancy application configuration.
<b>debug redundancy application group protocol</b>	Displays the redundancy application group protocol information.

# debug redundancy application group rii

To display the redundancy application group RII information, use the **debug redundancy application group rii** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

```
debug redundancy application group rii {error | event}
```

```
no debug redundancy application group rii {error | event }
```

## Syntax Description

<b>error</b>	Displays RII errors information about the redundancy group's .
<b>event</b>	Display information about the redundancy group's RII events.

## Command Modes

Privileged EXEC (#)

## Command History

Release	Modification
Cisco IOS XE Release 3.1S	This command was introduced.

## Examples

The following is sample output from the **debug redundancy application group rii event** command:

```
Router# debug redundancy application group rii event

RG RII events debugging is on
```

## Related Commands

Command	Description
<b>redundancy application group config</b>	Displays the redundancy group application configuration.
<b>debug redundancy application group protocol</b>	Displays the redundancy group application group protocol information.
<b>debug redundancy application group rii</b>	Displays the redundancy group application group RII information.
<b>debug redundancy application group vp</b>	Displays the redundancy group application group VP information.
<b>redundancy application group config</b>	Displays the redundancy group application configuration.

<b>Command</b>	<b>Description</b>
<b>debug redundancy application group media</b>	Displays the redundancy application group media information.
<b>debug redundancy application group protocol</b>	Displays the redundancy application group protocol information.

# debug redundancy application group transport

To display the redundancy application group transport information, use the **debug redundancy application group transport** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

```
debug redundancy application group transport {db | error | event | packet | timer | trace}
```

```
no debug redundancy application group transport {db | error | event | packet | timer | trace}
```

## Syntax Description

<b>db</b>	Displays transport information of a redundancy group.
<b>error</b>	Displays transport error information of a redundancy group.
<b>event</b>	Displays transport event information of a redundancy group.
<b>packet</b>	Displays transport packet information of a redundancy group.
<b>timer</b>	Displays transport timer information of a redundancy group.
<b>trace</b>	Displays transport trace information of a redundancy group.

## Command Modes

Privileged EXEC (#)

## Command History

Release	Modification
Cisco IOS XE Release 3.1S	This command was introduced.

## Examples

The following is sample output from the **debug redundancy application group transport trace** command:

```
Router# debug redundancy application group transport trace
```

```
RG Transport trace debugging is on
```

## Related Commands

Command	Description
<b>redundancy application group config</b>	Displays the redundancy group application configuration.
<b>debug redundancy application group protocol</b>	Displays the redundancy application group protocol information.
<b>debug redundancy application group rii</b>	Displays the redundancy application group RII information.
<b>debug redundancy application group transport</b>	Displays the redundancy application group transport information.

<b>Command</b>	<b>Description</b>
<b>redundancy application group config</b>	Displays the redundancy group application configuration.
<b>debug redundancy application group media</b>	Displays the redundancy application group media information.
<b>debug redundancy application group protocol</b>	Displays the redundancy application group protocol information.

# debug redundancy application group vp

To display the redundancy application group virtual platform (VP) information, use the **debug redundancy application group VP** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

```
debug redundancy application group vp {error | event}
```

```
no debug redundancy application group vp{error | event}
```

## Syntax Description

<b>error</b>	Displays VP errors information of a redundancy group.
<b>event</b>	Displays VP event information of a redundancy group.

## Command Modes

Privileged EXEC (#)

## Command History

Release	Modification
Cisco IOS XE Release 3.1S	This command was introduced.

## Examples

The following is sample output from the **debug redundancy application group vp event** command:

```
Router# debug redundancy application group vp event

RG VP events debugging is on
```

## Related Commands

Command	Description
<b>redundancy application group config</b>	Displays the redundancy group application configuration.
<b>debug redundancy application group protocol</b>	Displays the redundancy application group protocol information.
<b>debug redundancy application group rii</b>	Displays the redundancy application group RII information.
<b>debug redundancy application group transport</b>	Displays the redundancy application group transport information.
<b>redundancy application group config</b>	Displays the redundancy application configuration.
<b>debug redundancy application group media</b>	Displays the redundancy application group media information.

<b>Command</b>	<b>Description</b>
<b>debug redundancy application group protocol</b>	Displays the redundancy application group protocol information.
<b>redundancy application group config</b>	Displays the redundancy application configuration.

# debug redundancy as5850

To enable specific redundancy-related debug options, use the **debug redundancy as5850** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

```
debug redundancy as5850 { fsm | lines | master | mode | rf-client }
```

```
no debug redundancy as5850
```

## Syntax Description

<b>fsm</b>	Finite-state-machine events.
<b>lines</b>	Hardware lines.
<b>master</b>	Master (active rather than standby) route-switch-controller (RSC).
<b>mode</b>	RSC's mode: classic-split or handover-split.
<b>rf-client</b>	Redundancy-related client-application information.

## Defaults

This command is disabled

## Command Modes

Privileged EXEC

## Command History

Release	Modification
12.2(2)XB1	This command was introduced.
12.2(11)T	This command was integrated into Cisco IOS Release 12.2(11)T.
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 master form of the command to view redundancy-related debug entries. All debug entries continue to be logged even if you do not specify an option here, and you can always use the **show redundancy debug-log** command to view them.

## Examples

The output from this command consists of event announcements that can be used by authorized troubleshooting personnel.

## Related Commands

Command	Description
<b>show redundancy debug-log</b>	Displays up to 256 debug entries.

# debug registry

To turn on the debugging output for registry events or errors when Cisco IOS Software Modularity software is running, use the **debug registry** command in privileged EXEC mode. To turn off debugging output, use the **no** form of this command or the **undebug** command.

```
debug registry {events | errors} [process-name | pid]
```

```
no debug registry {events | errors} [process-name | pid]
```

## Syntax Description

<b>events</b>	Displays debugging messages about registry event messages.
<b>errors</b>	Displays debugging messages about registry error messages.
<i>process-name</i>	(Optional) Process name.
<i>pid</i>	(Optional) Process ID. Number in the range from 1 to 4294967295.

## Command Modes

Privileged EXEC

## Command History

Release	Modification
12.2(18)SXF4	This command was introduced to support Software Modularity images.
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 registry** command to troubleshoot Software Modularity registry operations.



### Caution

Use any debugging command with caution because the volume of generated output can slow or stop the router operations. We recommend that this command be used only under the supervision of a Cisco engineer.

## Examples

The following example turns on debugging messages for Software Modularity registry events for the TCP process:

```
Router# debug registry events tcp.proc
```

```
Debug registry events debugging is on
```

The following example turns on debugging messages for Software Modularity registry errors:

```
Router# debug registry errors
```

```
Debug registry errors debugging is on
```

# debug resource policy notification

To trace the Embedded Resource Manager (ERM) notification activities for resources using the ERM feature, use the **debug resource policy notification** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug resource policy notification** [**owner** *resource-owner-name*]

**no debug resource policy notification** [**owner** *resource-owner-name*]

## Syntax Description

**owner** *resource-owner-name* (Optional) Specifies the name of the resource owner (RO).

## Command Default

Disabled

## 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.
12.2SX	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.

## Examples

The following example shows different instances of the **debug resource policy notification** command:

```
Router# debug resource policy notification
```

```
Enabled notif. debugs on all owners
```

When a threshold is violated, the following messages are displayed:

```
*Mar 3 09:50:44.081: Owner: 'memory' initiated a notification:
```

```
*Mar 3 09:50:44.081: %SYS-4-RESMEMEXCEED: Resource user usrr1 has exceeded the Major memory threshold
```

```
Pool: Processor Used: 42932864 Threshold :42932860
```

```
*Mar 3 09:50:46.081: Notification from Owner: 'memory' is dispatched for User: 'usrr1' (ID: 0x10000B9)
```

```
*Mar 3 09:50:46.081: %SYS-4-RESMEMEXCEED: Resource user usrr1 has exceeded the Major memory threshold
```

```
Pool: Processor Used: 42932864 Threshold :42932860
```

```
Router# no debug resource manager notification
```

```
Disabled notif. debugs on all owners
```

```
Router# debug resource manager notification owner cpu
```

```

Enabled notif. debugs on owner 'cpu'

Router# no debug resource manager notification owner cpu

Disabled notif. debugs on owner 'cpu'

Router# debug resource manager notification owner memory

Enabled notif. debugs on owner 'memory'

Router# no debug resource manager notification owner memory

Disabled notif. debugs on owner 'memory'

Router# debug resource manager notification owner Buffer

Enabled notif. debugs on owner 'Buffer'

Router# no debug resource manager notification owner Buffer

Disabled notif. debugs on owner 'Buffer'

Router# no debug resource manager notification owner Buffer

Disabled notif. debugs on owner 'Buffer'

```

**Related Commands**

<b>Command</b>	<b>Description</b>
<b>debug resource policy registration</b>	Displays the resource policy registration debug information for the ERM resources.

# debug resource policy registration

To trace the Embedded Resource Manager (ERM) registration activities for resources using the ERM feature, use the **debug resource policy registration** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug resource policy registration**

**no debug resource policy registration**

**Syntax Description** This command has no arguments or keywords.

**Command Default** Disabled

**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.
	12.2SX	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.

**Examples** The following example shows output from the **debug resource policy registration** command:

```
Router# debug resource policy registration
```

```
Registrations debugging is on
```

When a Resource User (RU) is created, the following message is displayed:

```
*Mar 3 09:35:58.304: resource_user_register: RU: ruID: 0x10000B8, rutID: 0x1, rg_ID: 0x0
name: usrr1
```

When an RU is deleted, the following message is displayed:

```
*Mar 3 09:41:09.500: resource_user_unregister: RU: ruID: 0x10000B8, rutID: 0x1, rg_ID:
0x0 name: usrr1
```

Related Commands	Command	Description
	<b>debug resource policy notification</b>	Displays the resource policy notification debug information for the ERM resources.

# debug resource-pool

To see and trace resource pool management activity, use the **debug resource-pool** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug resource-pool**

**no debug resource-pool**

**Syntax Description** This command has no arguments or keywords.

**Defaults** Disabled.

**Command Modes** Privileged EXEC

Command History	Release	Modification
	12.0(4)XI	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.

**Usage Guidelines** Enter the **debug resource-pool** command to see and trace resource pool management activity. [Table 294](#) describes the resource pooling states.

**Table 294 Resource Pooling States**

State	Description
RM_IDLE	No call activity.
RM_RES_AUTHOR	Call waiting for authorization, message sent to authentication, authorization, and accounting (AAA).
RM_RES_ALLOCATING	Call authorized, resource-grp-mgr allocating.
RM_RES_ALLOCATED	Resource allocated, connection acknowledgment sent to signalling state. Call should get connected and become active.
RM_AUTH_REQ_IDLE	Signalling module disconnected call while in RM_RES_AUTHOR. Waiting for authorization response from AAA.
RM_RES_REQ_IDLE	Signalling module disconnected call while in RM_RES_ALLOCATING. Waiting for resource allocation response from resource-group manager.

**Table 294** Resource Pooling States (continued)

State	Description
RM_DNIS_AUTHOR	An intermediate state before proceeding with Route Processor Module (RPM) authorization.
RM_DNIS_AUTH_SUCCEEDED	Dialed number identification service (DNIS) authorization succeeded.
RM_DNIS_RES_ALLOCATED	DNIS resource allocated.
RM_DNIS_AUTH_REQ_IDLE	DNIS authorization request idle.
RM_DNIS_AUTHOR_FAIL	DNIS authorization failed.
RM_DNIS_RES_ALLOC_SUCC ESS	DNIS resource allocation succeeded.
RM_DNIS_RES_ALLOC_FAIL	DNIS resource allocation failed.
RM_DNIS_RPM_REQUEST	DNIS resource pool management requested.

You can use the resource pool state to isolate problems. For example, if a call fails authorization in the RM\_RES\_AUTHOR state, investigate further with AAA authorization debugs to determine whether the problem lies in the resource-pool manager, AAA, or dispatcher.

### Examples

The following example shows different instances where you can use the **debug resource-pool** command:

```
Router# debug resource-pool

RM general debugging is on

Router# show debug

General OS:
  AAA Authorization debugging is on
Resource Pool:
  resource-pool general debugging is on
Router #
Router #ping 21.1.1.10
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 21.1.1.10, timeout is 2 seconds:
*Jan  8 00:10:30.358: RM state:RM_IDLE event:DIALER_INCALL DS0:0:0:0:1
*Jan  8 00:10:30.358: RM: event incoming call

/* An incoming call is received by RM */

*Jan  8 00:10:30.358: RM state:RM_DNIS_AUTHOR event:RM_DNIS_RPM_REQUEST
DS0:0:0:0:1

/* Receives an event notifying to proceed with RPM authorization while
in DNIS authorization state */

*Jan  8 00:10:30.358: RM:RPM event incoming call
*Jan  8 00:10:30.358: RPM profile cp1 found

/* A customer profile "cp1" is found matching for the incoming call, in
the local database */

*Jan  8 00:10:30.358: RM state:RM_RPM_RES_AUTHOR
event:RM_RPM_RES_AUTHOR_SUCCESS DS0:0:0:0:1
```

```

/* Resource authorization success event received while in resource
authorization state*/

*Jan 8 00:10:30.358: Allocated resource from res_group isdn1
*Jan 8 00:10:30.358: RM:RPM profile "cp1", allocated resource "isdn1"
successfully
*Jan 8 00:10:30.358: RM state:RM_RPM_RES_ALLOCATING
event:RM_RPM_RES_ALLOC_SUCCESS DS0:0:0:0:1

/* Resource allocation success event received while attempting to
allocate a resource */
*Jan 8 00:10:30.358: Se0:1 AAA/ACCT/RM: doing resource-allocated
(local) (nothing to do)
*Jan 8 00:10:30.366: %LINK-3-UPDOWN: Interface Serial0:1, changed state
to up
*Jan 8 00:10:30.370: %LINK-3-UPDOWN: Interface Serial0:1, changed state
to down
*Jan 8 00:10:30.570: Se0:1 AAA/ACCT/RM: doing resource-update (local)
cp1 (nothing to do)
*Jan 8 00:10:30.578: %LINK-3-UPDOWN: Interface Serial0:0, changed
state to up
*Jan 8 00:10:30.582: %DIALER-6-BIND: Interface Serial0:0 bound to
profile Dialer0...
Success rate is 0 percent (0/5)
Router #
*Jan 8 00:10:36.662: %ISDN-6-CONNECT: Interface Serial0:0 is now
connected to 71017
*Jan 8 00:10:52.990: %DIALER-6-UNBIND: Interface Serial0:0 unbound from
profile Dialer0
*Jan 8 00:10:52.990: %ISDN-6-DISCONNECT: Interface Serial0:0
disconnected from 71017 , call lasted 22 seconds
*Jan 8 00:10:53.206: %LINK-3-UPDOWN: Interface Serial0:0, changed state
to down
*Jan 8 00:10:53.206: %ISDN-6-DISCONNECT: Interface Serial0:1
disconnected from unknown , call lasted 22 seconds
*Jan 8 00:10:53.626: RM state:RM_RPM_RES_ALLOCATED event:DIALER_DISCON
DS0:0:0:0:1

/* Received Disconnect event from signalling stack for a call which
has a resource allocated. */

*Jan 8 00:10:53.626: RM:RPM event call drop

/* RM processing the disconnect event */

*Jan 8 00:10:53.626: Deallocated resource from res_group isdn1
*Jan 8 00:10:53.626: RM state:RM_RPM_DISCONNECTING
event:RM_RPM_DISC_ACK DS0:0:0:0:1

/* An intermediate state while the DISCONNECT event is being processed
by external servers, before RM goes back into IDLE state.
*/

```

Table 295 describes the significant fields shown in the display.

**Table 295** *debug resource-pool Field Descriptions*

Field	Description
RM state:RM_IDLE	Resource manager state that displays no active calls.
RM state:RM_RES_AUTHOR	Resource authorization state.

**Table 295** *debug resource-pool Field Descriptions (continued)*

<b>Field</b>	<b>Description</b>
RES_AUTHOR_SUCCESS DS0: shelf:slot:port:channel	Actual physical resource that is used
Allocated resource from res_group	Physical resource group that accepts the call.
RM profile <x>, allocated resource <x>	Specific customer profile and resource group names used to accept the call.
RM state: RM_RES_ALLOCATING	Resource manager state that unifies a call with a physical resource.

# debug rif

To display information on entries entering and leaving the routing information field (RIF) cache, use the **debug rif** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug rif**

**no debug rif**

**Syntax Description** This command has no arguments or keywords.

**Command Modes** Privileged EXEC

**Usage Guidelines** In order to use the **debug rif** command to display traffic source-routed through an interface, fast switching of source route bridging (SRB) frames must first be disabled with the **no source-bridge route-cache** interface configuration command.

**Examples** The following is sample output from the **debug rif** command:

```

router# debug rif
SDLLC or Local-Ack entry — RIF: U chk da=9000.5a59.04f9,sa=0110.2222.33c1 [4880.3201.00A1.0050] type 8 on
                             static/remote/0
                             RIF: U chk da=0000.3080.4aed,sa=0000.0000.0000 [] type 8 on TokenRing0/0
                             RIF: U add 1000.5a59.04f9 [4880.3201.00A1.0050] type 8
Non-SDLLC or non-Local-Ack entry — RIF: L checking da=0000.3080.4aed, sa=0000.0000.0000
                             RIF: rcvd TEST response from 9000.5a59.04f9
                             RIF: U upd da=1000.5a59.04f9,sa=0110.2222.33c1 [4880.3201.00A1.0050]
                             RIF: rcvd XID response from 9000.5a59.04f9
                             SR1: sent XID response to 9000.5a59.04f9

```

S2559

The first line of output is an example of a RIF entry for an interface configured for SDLC Logical Link Control (SDLLC) or Local-Ack. [Table 296](#) describes significant fields shown in the display.

**Table 296** *debug rif* Field Descriptions

Field	Description
RIF:	This message describes RIF debugging output.
U chk	Update checking. The entry is being updated; the timer is set to zero (0).
da=9000.5a59.04f9	Destination MAC address.
sa=0110.2222.33c1	Source MAC address. This field contains values of zero (0000.0000.0000) in a non-SDLLC or non-Local-Ack entry.

**Table 296** *debug rif Field Descriptions (continued)*

Field	Description
[4880.3201.00A1.0050]	RIF string. This field is blank (null RIF) in a non-SDLLC or non-Local-Ack entry.
type 8	Possible values follow: <ul style="list-style-type: none"> <li>• 0—Null entry</li> <li>• 1—This entry was learned from a particular Token Ring port (interface)</li> <li>• 2—Statically configured</li> <li>• 4—Statically configured for a remote interface</li> <li>• 8—This entry is to be aged</li> <li>• 16—This entry (which has been learned from a remote interface) is to be aged</li> <li>• 32—This entry is not to be aged</li> <li>• 64—This interface is to be used by LAN Network Manager (and is not to be aged)</li> </ul>
on static/remote/0	This route was learned from a real Token Ring port, in contrast to a virtual ring.

The following line of output is an example of a RIF entry for an interface that is not configured for SDLLC or Local-Ack:

```
RIF: U chk da=0000.3080.4aed,sa=0000.0000.0000 [] type 8 on TokenRing0/0
```

Notice that the source address contains only zero values (0000.0000.0000), and that the RIF string is null ([ ]). The last element in the entry indicates that this route was learned from a virtual ring, rather than a real Token Ring port.

The following line shows that a new entry has been added to the RIF cache:

```
RIF: U add 1000.5a59.04f9 [4880.3201.00A1.0050] type 8
```

The following line shows that a RIF cache lookup operation has taken place:

```
RIF: L checking da=0000.3080.4aed, sa=0000.0000.0000
```

The following line shows that a TEST response from address 9000.5a59.04f9 was inserted into the RIF cache:

```
RIF: rcvd TEST response from 9000.5a59.04f9
```

The following line shows that the RIF entry for this route has been found and updated:

```
RIF: U upd da=1000.5a59.04f9,sa=0110.2222.33c1 [4880.3201.00A1.0050]
```

The following line shows that an XID response from this address was inserted into the RIF cache:

```
RIF: rcvd XID response from 9000.5a59.04f9
```

The following line shows that the router sent an XID response to this address:

```
SR1: sent XID response to 9000.5a59.04f9
```

Table 297 explains the other possible lines of **debug rif** command output.

**Table 297 Additional debug rif Field Descriptions**

Field	Description
RIF: L Sending XID for <address>	Router/bridge wanted to send a packet to <i>address</i> but did not find it in the RIF cache. It sent an XID explorer packet to determine which RIF it should use. The attempted packet is dropped.
RIF: L No buffer for XID to <address>	Similar to the previous description; however, a buffer in which to build the XID packet could not be obtained.
RIF: U remote rif too small <rif>	Packet's RIF was too short to be valid.
RIF: U rej <address> too big <rif>	Packet's RIF exceeded the maximum size allowed and was rejected. The maximum size is 18 bytes.
RIF: U upd interface <address>	RIF entry for this router/bridge's interface has been updated.
RIF: U ign <address> interface update	RIF entry that would have updated an interface corresponding to one of this router's interfaces.
RIF: U add <address> <rif>	RIF entry for <i>address</i> has been added to the RIF cache.
RIF: U no memory to add rif for <address>	No memory to add a RIF entry for <i>address</i> .
RIF: removing rif entry for <address, type code>	RIF entry for <i>address</i> has been forcibly removed.
RIF: flushed <address>	RIF entry for <i>address</i> has been removed because of a RIF cache flush.
RIF: expired <address>	RIF entry for <i>address</i> has been aged out of the RIF cache.

#### Related Commands

Command	Description
<b>debug list</b>	Filters debugging information on a per-interface or per-access list basis.

# debug route-map ipc

To display a summary of the one-way Inter-process Communications (IPC) messages set from the route processor (RP) to the Versatile Interface Processor (VIP) about NetFlow policy routing when distributed Cisco Express Forwarding (dCEF) is enabled, use the **debug route-map ipc** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug route-map ipc**

**no debug route-map ipc**

**Syntax Description** This command has no arguments or keywords.

**Command Modes** Privileged EXEC

Command History	Release	Modification
	12.0(3)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.

**Usage Guidelines** This command is especially helpful for policy routing with dCEF switching.

This command displays a summary of one-way IPC messages from the RP to the VIP about NetFlow policy routing. If you execute this command on the RP, the messages are shown as “Sent.” If you execute this command on the VIP console, the IPC messages are shown as “Received.”

**Examples** The following is sample output from the **debug route-map ipc** command executed at the RP:

```
Router# debug route-map ipc

Routemap related IPC debugging is on

Router# configure terminal

Enter configuration commands, one per line. End with CNTL/Z.

Router(config)# ip cef distributed

Router(config)#^Z
Router#

RM-IPC: Clean routemap config in slot 0
RM-IPC: Sent clean-all-routemaps; len 12
RM-IPC: Download all policy-routing related routemap config to slot 0
RM-IPC: Sent add routemap test(seq:10); n_len 5; len 17
RM-IPC: Sent add acl 1 of routemap test(seq:10); len 21
RM-IPC: Sent add min 10 max 300 of routemap test(seq:10); len 24
```

```
RM-IPC: Sent add preced 1 of routemap test(seq:10); len 17
RM-IPC: Sent add tos 4 of routemap test(seq:10); len 17
RM-IPC: Sent add nexthop 50.0.0.8 of routemap test(seq:10); len 20
RM-IPC: Sent add default nexthop 50.0.0.9 of routemap test(seq:10); len 20
RM-IPC: Sent add interface Ethernet0/0/3(5) of routemap test(seq:10); len 20
RM-IPC: Sent add default interface Ethernet0/0/2(4) of routemap test(seq:10); len 20
```

The following is sample output from the **debug route-map ipc** command executed at the VIP:

```
VIP-Slot0# debug route-map ipc
```

```
Routemap related IPC debugging is on
```

```
VIP-Slot0#
RM-IPC: Rcvd clean-all-routemaps; len 12
RM-IPC: Rcvd add routemap test(seq:10); n_len 5; len 17
RM-IPC: Rcvd add acl 1 of routemap test(seq:10); len 21
RM-IPC: Rcvd add min 10 max 300 of routemap test(seq:10); len 24
RM-IPC: Rcvd add preced 1 of routemap test(seq:10); len 17
RM-IPC: Rcvd add tos 4 of routemap test(seq:10); len 17
RP-IPC: Rcvd add nexthop 50.0.0.8 of routemap test(seq:10); len 20
RP-IPC: Rcvd add default nexthop 50.0.0.9 of routemap test(seq:10); len 20
RM-IPC: Rcvd add interface Ethernet0/3 of routemap tes; len 20
RM-IPC: Rcvd add default interface Ethernet0/2 of routemap test(seq:10); len 20
```

# debug rpms-proc preauth

To enable diagnostic reporting of preauthentication information, use the **debug rpms-proc preauth** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

```
debug rpms-proc preauth {all | h323 | sip}
```

```
no debug rpms-proc preauth {all | h323 | sip}
```

## Syntax Description

<b>all</b>	Provides information for all calls.
<b>h323</b>	Provides information for H.323 calls.
<b>sip</b>	Provides information for Session Initiation Protocol (SIP) calls.

## Defaults

No default behavior or values

## Command Modes

Privileged EXEC

## Command History

Release	Modification
12.2(11)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 example shows debugging output for two calls. The first is a leg 3 SIP call, and the second is a leg 3 H.323 call:

```
Router# debug rpms-proc preauth all
```

```
All RPMS Process preauth tracing is enabled
Feb 10 14:00:07.236: Entering rpms_proc_print_preauth_req

Feb 10 14:00:07.236: Request = 0
Feb 10 14:00:07.236: Preauth id = 8
Feb 10 14:00:07.236: EndPt Type = 1
Feb 10 14:00:07.236: EndPt = 192.168.80.70
Feb 10 14:00:07.236: Resource Service = 1
Feb 10 14:00:07.236: Call_origin = answer
Feb 10 14:00:07.236: Call_type = voip
Feb 10 14:00:07.236: Calling_num = 2220001
Feb 10 14:00:07.236: Called_num = 1120001
Feb 10 14:00:07.236: Protocol = 1
Feb 10 14:00:07.236:rpms_proc_create_node:Created node with preauth_id = 8
Feb 10 14:00:07.236:rpms_proc_send_aaa_req:uid got is 19
Feb 10 14:00:07.240:rpms_proc_preauth_response:Context is for preauth_id 8, aaa_uid 19
Feb 10 14:00:07.240:rpms_proc_preauth_response:Deleting Tree node for preauth id 8 uid 19
Feb 10 14:00:07.284: Entering rpms_proc_print_preauth_req

Feb 10 14:00:07.284: Request = 0
```

```

Feb 10 14:00:07.284: Preauth id = 9
Feb 10 14:00:07.284: EndPt Type = 1
Feb 10 14:00:07.284: EndPt = 192.168.81.102
Feb 10 14:00:07.284: Resource Service = 1
Feb 10 14:00:07.284: Call_origin = answer
Feb 10 14:00:07.284: Call_type = voip
Feb 10 14:00:07.284: Calling_num = 2210001
Feb 10 14:00:07.284: Called_num = 1#1110001
Feb 10 14:00:07.284: Protocol = 0
Feb 10 14:00:07.288:rpms_proc_create_node:Created node with preauth_id = 9
Feb 10 14:00:07.288:rpms_proc_send_aaa_req:uid got is 21
Feb 10 14:00:07.300:rpms_proc_preauth_response:Context is for preauth_id 9, aaa_uid 21
Feb 10 14:00:07.300:rpms_proc_preauth_response:Deleting Tree node for preauth id 9 uid 21

```

The following example shows the output for a single leg 3 H.323 call:

```
Router# debug rpms-proc preauth h323
```

```

RPMS Process H323 preauth tracing is enabled
Feb 10 14:04:57.867: Entering rpms_proc_print_preauth_req

Feb 10 14:04:57.867: Request = 0
Feb 10 14:04:57.867: Preauth id = 10
Feb 10 14:04:57.867: EndPt Type = 1
Feb 10 14:04:57.867: EndPt = 192.168.81.102
Feb 10 14:04:57.867: Resource Service = 1
Feb 10 14:04:57.867: Call_origin = answer
Feb 10 14:04:57.867: Call_type = voip
Feb 10 14:04:57.867: Calling_num = 2210001
Feb 10 14:04:57.867: Called_num = 1#1110001
Feb 10 14:04:57.867: Protocol = 0
Feb 10 14:04:57.867:rpms_proc_create_node:Created node with preauth_id = 10
Feb 10 14:04:57.867:rpms_proc_send_aaa_req:uid got is 25
Feb 10 14:04:57.875:rpms_proc_preauth_response:Context is for preauth_id 10, aaa_uid 25
Feb 10 14:04:57.875:rpms_proc_preauth_response:Deleting Tree node for preauth id 10 uid 25

```

The following example shows output for a single leg 3 SIP call:

```
Router# debug rpms-proc preauth sip
```

```

RPMS Process SIP preauth tracing is enabled
Feb 10 14:08:02.880: Entering rpms_proc_print_preauth_req

Feb 10 14:08:02.880: Request = 0
Feb 10 14:08:02.880: Preauth id = 11
Feb 10 14:08:02.880: EndPt Type = 1
Feb 10 14:08:02.880: EndPt = 192.168.80.70
Feb 10 14:08:02.880: Resource Service = 1
Feb 10 14:08:02.880: Call_origin = answer
Feb 10 14:08:02.880: Call_type = voip
Feb 10 14:08:02.880: Calling_num = 2220001
Feb 10 14:08:02.880: Called_num = 1120001
Feb 10 14:08:02.880: Protocol = 1
Feb 10 14:08:02.880:rpms_proc_create_node:Created node with preauth_id = 11
Feb 10 14:08:02.880:rpms_proc_send_aaa_req:uid got is 28
Feb 10 14:08:02.888:rpms_proc_preauth_response:Context is for preauth_id 11, aaa_uid 28
Feb 10 14:08:02.888:rpms_proc_preauth_response:Deleting Tree node for preauth id 11 uid 28

```

Table 298 describes the significant fields shown in the display.

**Table 298** *debug rpms-proc preauth Field Descriptions*

Field	Description
Request	Request Type—0 for preauthentication, 1 for disconnect.
Preauth id	Identifier for the preauthentication request.
EndPt Type	Call Origin End Point Type—1 for IP address, 2 for Interzone ClearToken (IZCT) value.
EndPt	Call Origin End Point Value—An IP address or IZCT value.
Resource Service	Resource Service Type—1 for Reservation, 2 for Query.
Call_origin	Answer.
Call_type	Voice over IP (VoIP).
Calling_num	Calling party number (calling line identification, or CLID).
Called_num	Called party number (dialed number identification service, or DNIS).
Protocol	0 for H.323, 1 for SIP.
function reports	Various identifiers and status reports for executed functions.

# debug rtpspi all

To debug all Routing Table Protocol (RTP) security parameter index (SPI) errors, sessions, and in/out functions, use the **debug rtpspi all** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug rtpspi all**

**no debug rtpspi all**

**Syntax Description** This command has no arguments or keywords.

**Defaults** No default behavior or values

**Command Modes** Privileged EXEC

Command History	Release	Modification
	12.0(7)XK	This command was introduced on the Cisco MC3810 and Cisco 3600 series routers (except the Cisco 3620).
	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



### Caution

Be careful when you use this command because it can result in console flooding and reduced voice quality.

## Examples

The following example shows a debug trace for RTP SPI errors, sessions, and in/out functions on a gateway:

```
Router# debug rtpspi all

RTP SPI Error, Session and function in/out tracings are enabled.

*Mar  1 00:38:59.381:rtpspi_allocate_rtp_port:Entered.
*Mar  1 00:38:59.381:rtpspi_allocate_rtp_port:allocated RTP port 16544
*Mar  1 00:38:59.381:rtpspi_allocate_rtp_port:Success. port = 16544. Leaving.
*Mar  1 00:38:59.381:rtpspi_call_setup_request:entered.
      Call Id = 5, dest = 0.0.0.0;   callInfo:
      final dest flag = 0,
      rtp_session_mode = 0x2,
      local_ip_addr = 0x5000001,remote_ip_addr = 0x0,
      local rtp port = 16544, remote rtp port = 0
```

```

*Mar 1 00:38:59.381:rtpspi_call_setup_request:spi_info copied for rtpspi_app_data_t.
*Mar 1 00:38:59.385:rtpspi_call_setup_request:leaving
*Mar 1 00:38:59.385:rtpspi_call_setup() entered
*Mar 1 00:38:59.385:rtpspi_initialize_ccb:Entered
*Mar 1 00:38:59.385:rtpspi_initialize_ccb:leaving
*Mar 1 00:38:59.385:rtpspi_call_setup:rtp_session_mode = 0x2
*Mar 1 00:38:59.385:rtpspi_call_setup:mode = CC_CALL_NORMAL.
    destination number = 0.0.0.0
*Mar 1 00:38:59.385:rtpspi_call_setup:Passed local_ip_addrs=0x5000001
*Mar 1 00:38:59.385:rtpspi_call_setup:Passed local_rtp_port = 16544
*Mar 1 00:38:59.385:rtpspi_call_setup:Saved RTCP Session = 0x1AF57E0
*Mar 1 00:38:59.385:rtpspi_call_setup:Passed remote rtp port = 0.
*Mar 1 00:38:59.389:rtpspi_start_rtcp_session:entered. rtp session mode=0x2, rem rtp=0,
rem ip=0x0
*Mar 1 00:38:59.389:rtpspi_get_rtcp_mode:entered. rtp_mode = 0x2
*Mar 1 00:38:59.389:rtpspi_start_rtcp_session:Starting RTCP session.
    Local IP addr = 0x5000001, Remote IP addr = 0x0,
    Local RTP port = 16544, Remote RTP port = 0, mode = 0x2
*Mar 1 00:38:59.389:rtpspi_start_rtcp_session:RTP Session creation Success.
*Mar 1 00:38:59.389:rtpspi_call_setup:RTP Session creation Success.
*Mar 1 00:38:59.389:rtpspi_call_setup:calling cc_api_call_connected()
*Mar 1 00:38:59.389:rtpspi_call_setup:Leaving.
*Mar 1 00:38:59.393:rtpspi_bridge:entered. conf id = 1, src i/f = 0x1859E88,
    dest i/f = 0x1964EEC, src call id = 5, dest call id = 4
    call info = 0x1919140, xmit fn = 0xDA7494, tag = 0
*Mar 1 00:38:59.393:rtpspi_get_rtcp_mode:entered. rtp_mode = 0x2
*Mar 1 00:38:59.393:rtpspi_modify_rtcp_session_parameters():xmit fn=0xDA7494,
dstIF=0x1964EEC, dstCallID=4, voip_mode=0x2, rtp_mode=0x2, ssrc_status=0
*Mar 1 00:38:59.393:rtpspi_bridge:Calling cc_api_bridge_done() for 5(0x1AF5400) and
4(0x0).
*Mar 1 00:38:59.393:rtpspi_bridge:leaving.
*Mar 1 00:38:59.397:rtpspi_caps_ind:Entered. vdb = 0x1859E88 call id = 5, srcCallId = 4
*Mar 1 00:38:59.397:rtpspi_caps_ind:caps from VTSP:codec=0x83FB, codec_bytes=0x50,
    fax rate=0x7F, vad=0x3 modem=0x0
*Mar 1 00:38:59.397:rtpspi_get_rtcp_session_parameters():CURRENT VALUES:
dstIF=0x1964EEC, dstCallID=4, current_seq_num=0x0
*Mar 1 00:38:59.397:rtpspi_get_rtcp_session_parameters():NEW VALUES:
dstIF=0x1964EEC, dstCallID=4, current_seq_num=0x261C
*Mar 1 00:38:59.397:rtpspi_caps_ind:Caps Used:codec=0x1, codec bytes=80,
    fax rate=0x1, vad=0x1, modem=0x1, dtmf_relay=0x1, seq_num_start=0x261D
*Mar 1 00:38:59.397:rtpspi_caps_ind:calling cc_api_caps_ind().
*Mar 1 00:38:59.397:rtpspi_caps_ind:Returning success
*Mar 1 00:38:59.397:rtpspi_caps_ack:Entered. call id = 5, srcCallId = 4
*Mar 1 00:38:59.397:rtpspi_caps_ack:leaving.
*Mar 1 00:38:59.618:rtpspi_call_modify:entered. call-id=5, nominator=0x7,
params=0x18DD440
*Mar 1 00:38:59.618:rtpspi_call_modify:leaving
*Mar 1 00:38:59.618:rtpspi_do_call_modify:Entered. call-id = 5
*Mar 1 00:38:59.622:rtpspi_do_call_modify:Remote RTP port changed. New port=16432
*Mar 1 00:38:59.622:rtpspi_do_call_modify:Remote IP addr changed. New IP addrs=0x6000001
*Mar 1 00:38:59.622:rtpspi_do_call_modify:new mode 2 is the same as the current mode
*Mar 1 00:38:59.622:rtpspi_do_call_modify:Starting new RTCP session.
*Mar 1 00:38:59.622:rtpspi_start_rtcp_session:entered. rtp session mode=0x2, rem
rtp=16432, rem ip=0x6000001
*Mar 1 00:38:59.622:rtpspi_get_rtcp_mode:entered. rtp_mode = 0x2
*Mar 1 00:38:59.622:rtpspi_start_rtcp_session:Removing old RTCP session.
*Mar 1 00:38:59.622:rtpspi_start_rtcp_session:Starting RTCP session.
    Local IP addr = 0x5000001, Remote IP addr = 0x6000001,
    Local RTP port = 16544, Remote RTP port = 16432, mode = 0x2
*Mar 1 00:38:59.622:rtpspi_start_rtcp_session:RTCP Timer creation Success. (5)*(5000)
*Mar 1 00:38:59.622:rtpspi_start_rtcp_session:RTP Session creation Success.
*Mar 1 00:38:59.622:rtpspi_do_call_modify:RTP Session creation Success.
*Mar 1 00:38:59.622:rtpspi_do_call_modify:Calling cc_api_call_modify(), result=0x0
*Mar 1 00:38:59.626:rtpspi_do_call_modify:success. leaving

```

```

*Mar 1 00:39:05.019:rtpspi_call_modify:entered. call-id=5, nominator=0x7,
params=0x18DD440
*Mar 1 00:39:05.019:rtpspi_call_modify:leaving
*Mar 1 00:39:05.019:rtpspi_do_call_modify:Entered. call-id = 5
*Mar 1 00:39:05.019:rtpspi_do_call_modify:New remote RTP port = old rtp port = 16432
*Mar 1 00:39:05.019:rtpspi_do_call_modify:New remote IP addr = old IP addr = 0x6000001
*Mar 1 00:39:05.019:rtpspi_do_call_modify:Mode changed. new = 3, old = 2
*Mar 1 00:39:05.019:rtpspi_get_rtcp_mode:entered. rtp_mode = 0x3
*Mar 1 00:39:05.023:rtpspi_modify_rtcp_session_parameters():xmit fn=0xDA7494,
dstIF=0x1964EEC, dstCallID=4, voip_mode=0x3, rtp_mode=0x3, ssrc_status=2
*Mar 1 00:39:05.023:rtpspi_do_call_modify:RTCP Timer start.
*Mar 1 00:39:05.023:rtpspi_do_call_modify:Calling cc_api_call_modify(), result=0x0
*Mar 1 00:39:05.023:rtpspi_do_call_modify:success. leaving
*Mar 1 00:40:13.786:rtpspi_bridge_drop:entered. src call-id=5, dest call-id=4, tag=0
*Mar 1 00:40:13.786:rtpspi_get_rtcp_mode:entered. rtp_mode = 0x3
*Mar 1 00:40:13.786:rtpspi_modify_rtcp_session_parameters():xmit fn=0x0,
dstIF=0x0, dstCallID=0, voip_mode=0x3, rtp_mode=0x3, ssrc_status=2
*Mar 1 00:40:13.786:rtpspi_bridge_drop:leaving
*Mar 1 00:40:13.790:rtpspi_call_disconnect:entered. call-id=5, cause=16, tag=0
*Mar 1 00:40:13.790:rtpspi_call_disconnect:leaving.
*Mar 1 00:40:13.790:rtpspi_do_call_disconnect:Entered. call-id = 5
*Mar 1 00:40:13.790:rtpspi_do_call_disconnect:calling rtpspi_call_cleanup(). call-id=5
*Mar 1 00:40:13.794:rtpspi_call_cleanup:entered. ccb = 0x1AF5400, call-id=5, rtp port =
16544
*Mar 1 00:40:13.794:rtpspi_call_cleanup:releasing ccb cache. RTP port=16544
*Mar 1 00:40:13.794:rtpspi_store_call_history_entry():Entered.
*Mar 1 00:40:13.794:rtpspi_store_call_history_entry():Leaving.
*Mar 1 00:40:13.794:rtpspi_call_cleanup:RTCP Timer Stop.
*Mar 1 00:40:13.794:rtpspi_call_cleanup:deallocating RTP port 16544.
*Mar 1 00:40:13.794:rtpspi_free_rtcp_session:Entered.
*Mar 1 00:40:13.794:rtpspi_free_rtcp_session:Success. Leaving
*Mar 1 00:40:13.794:rtpspi_call_cleanup:freeing ccb (0x1AF5400)
*Mar 1 00:40:13.794:rtpspi_call_cleanup:leaving
*Mar 1 00:40:13.794:rtpspi_do_call_disconnect:leaving

```

**Related Commands**

Command	Description
<b>debug rtpspi errors</b>	Debugs RTP SPI errors.
<b>debug rtpspi inout</b>	Debugs RTP SPI in/out functions.
<b>debug rtpspi send-nse</b>	Triggers the RTP SPI to send a triple redundant NSE.
<b>debug sgcp errors</b>	Debugs SGCP errors.
<b>debug sgcp events</b>	Debugs SGCP events.
<b>debug sgcp packet</b>	Debugs SGCP packets.
<b>debug vtsp send-nse</b>	Sends and debugs a triple redundant NSE from the DSP to a remote gateway.

# debug rtpspi errors

To debug Routing Table Protocol (RTP) security parameter index (SPI) errors, use the **debug rtpspi errors** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug rtpspi errors**

**no debug rtpspi errors**

## Syntax Description

This command has no arguments or keywords.

## Defaults

No default behavior or values

## Command Modes

Privileged EXEC

## Command History

Release	Modification
12.0(7)XK	This command was introduced on the Cisco MC3810 device and Cisco 3600 series routers (except the Cisco 3620).
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



### Caution

Be careful when you use this command because it can result in console flooding and reduced voice quality.

## Examples

This example shows a debug trace for RTP SPI errors on two gateways. The following example shows the debug trace on the first gateway:

```
Router# debug rtpspi errors

00:54:13.272:rtpspi_do_call_modify:new mode 2 is the same as the current mode
00:54:18.738:rtpspi_do_call_modify:New remote RTP port = old rtp port = 16452
00:54:18.738:rtpspi_do_call_modify:New remote IP addr = old IP addr = 0x6000001
```

The following example shows the debug trace on the second gateway:

```
Router# debug rtpspi errors

00:54:08:rtpspi_process_timers:
```

## ■ debug rtpspi errors

```

00:54:08:rtpspi_process_timers:Timer 0x1A5AF9C expired.
00:54:08:rtpspi_process_timers:Timer expired for callID 0x3
00:54:08:rtpspi_process_timers:
00:54:08:rtpspi_process_timers:Timer 0x1A5AF9C expired.
00:54:08:rtpspi_process_timers:Timer expired for callID 0x3
00:54:08:rtpspi_process_timers:
00:54:08:rtpspi_process_timers:Timer 0x1A5AF9C expired.
00:54:08:rtpspi_process_timers:Timer expired for callID 0x3
00:54:09:rtpspi_process_timers:
00:54:09:rtpspi_process_timers:Timer 0x1A5AFBC expired.
00:54:09:rtpspi_process_timers:Timer expired for callID 0x3
00:54:09:rtpspi_process_timers:
00:54:09:rtpspi_process_timers:Timer 0x1A5B364 expired.
00:54:09:rtpspi_process_timers:Timer expired for callID 0x3

```

**Related Commands**

<b>Command</b>	<b>Description</b>
<b>debug rtpspi all</b>	Debugs all RTP SPI errors, sessions, and in/out functions.
<b>debug rtpspi inout</b>	Debugs RTP SPI in/out functions.
<b>debug rtpspi send-nse</b>	Triggers the RTP SPI to send a triple redundant NSE.
<b>debug sgcp errors</b>	Debugs SGCP errors.
<b>debug sgcp events</b>	Debugs SGCP events.
<b>debug sgcp packet</b>	Debugs SGCP packets.
<b>debug vtsp send-nse</b>	Sends and debugs a triple redundant NSE from the DSP to a remote gateway.

# debug rtpspi inout

To debug Routing Table Protocol (RTP) security parameter index (SPI) in/out functions, use the **debug rtpspi inout** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug rtpspi inout**

**no debug rtpspi inout**

## Syntax Description

This command has no arguments or keywords.

## Defaults

No default behavior or values

## Command Modes

Privileged EXEC

## Command History

Release	Modification
12.0(7)XK	This command was introduced on the Cisco MC3810 device and Cisco 3600 series routers (except the Cisco 3620 device).
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



### Caution

Be careful when you use this command because it can result in console flooding and reduced voice quality.

## Examples

The following example shows a debug trace for RTP SPI in/out functions on a gateway:

```
Router# debug rtpspi inout

*Mar  1 00:57:24.565:rtpspi_allocate_rtp_port:Entered.
*Mar  1 00:57:24.565:rtpspi_allocate_rtp_port:Success. port = 16520. Leaving.
*Mar  1 00:57:24.565:rtpspi_call_setup_request:entered.
      Call Id = 9, dest = 0.0.0.0;  callInfo:
      final dest flag = 0,
      rtp_session_mode = 0x2,
      local_ip_addrs = 0x5000001,remote_ip_addrs = 0x0,
      local rtp port = 16520, remote rtp port = 0
*Mar  1 00:57:24.565:rtpspi_call_setup_request:spi_info copied for rtpspi_app_data_t.
*Mar  1 00:57:24.565:rtpspi_call_setup_request:leaving
*Mar  1 00:57:24.569:rtpspi_call_setup() entered
*Mar  1 00:57:24.569:rtpspi_initialize_ccb:Entered
```

```

*Mar 1 00:57:24.569:rtpspi_initialize_ccb:leaving
*Mar 1 00:57:24.569:rtpspi_start_rtcp_session:entered. rtp session mode=0x2, rem rtp=0,
rem ip=0x0
*Mar 1 00:57:24.569:rtpspi_get_rtcp_mode:entered. rtp_mode = 0x2
*Mar 1 00:57:24.569:rtpspi_call_setup:Leaving.
*Mar 1 00:57:24.573:rtpspi_bridge:entered. conf id = 3, src i/f = 0x1859E88,
dest i/f = 0x1964EEC, src call id = 9, dest call id = 8
call info = 0x1919140, xmit fn = 0xDA7494, tag = 0
*Mar 1 00:57:24.573:rtpspi_get_rtcp_mode:entered. rtp_mode = 0x2
*Mar 1 00:57:24.573:rtpspi_bridge:leaving.
*Mar 1 00:57:24.573:rtpspi_caps_ind:Entered. vdb = 0x1859E88 call id = 9, srcCallId = 8
*Mar 1 00:57:24.577:rtpspi_caps_ind:Returning success
*Mar 1 00:57:24.577:rtpspi_caps_ack:Entered. call id = 9, srcCallId = 8
*Mar 1 00:57:24.577:rtpspi_caps_ack:leaving.
*Mar 1 00:57:24.818:rtpspi_call_modify:entered. call-id=9, nominator=0x7,
params=0x18DD440
*Mar 1 00:57:24.818:rtpspi_call_modify:leaving
*Mar 1 00:57:24.818:rtpspi_do_call_modify:Entered. call-id = 9
*Mar 1 00:57:24.818:rtpspi_start_rtcp_session:entered. rtp session mode=0x2, rem
rtp=16396, rem ip=0x6000001
*Mar 1 00:57:24.822:rtpspi_get_rtcp_mode:entered. rtp_mode = 0x2
*Mar 1 00:57:24.822:rtpspi_do_call_modify:success. leaving
*Mar 1 00:57:30.296:rtpspi_call_modify:entered. call-id=9, nominator=0x7,
params=0x18DD440
*Mar 1 00:57:30.296:rtpspi_call_modify:leaving
*Mar 1 00:57:30.300:rtpspi_do_call_modify:Entered. call-id = 9
*Mar 1 00:57:30.300:rtpspi_get_rtcp_mode:entered. rtp_mode = 0x3
*Mar 1 00:57:30.300:rtpspi_do_call_modify:success. leaving
*Mar 1 00:58:39.055:rtpspi_bridge_drop:entered. src call-id=9, dest call-id=8, tag=0
*Mar 1 00:58:39.055:rtpspi_get_rtcp_mode:entered. rtp_mode = 0x3
*Mar 1 00:58:39.055:rtpspi_bridge_drop:leaving
*Mar 1 00:58:39.059:rtpspi_call_disconnect:entered. call-id=9, cause=16, tag=0
*Mar 1 00:58:39.059:rtpspi_call_disconnect:leaving.
*Mar 1 00:58:39.059:rtpspi_do_call_disconnect:Entered. call-id = 9
*Mar 1 00:58:39.059:rtpspi_call_cleanup:entered. ccb = 0x1AF5400, call-id=9, rtp port =
16520
*Mar 1 00:58:39.059:rtpspi_store_call_history_entry():Entered.
*Mar 1 00:58:39.059:rtpspi_store_call_history_entry():Leaving.
*Mar 1 00:58:39.059:rtpspi_free_rtcp_session:Entered.
*Mar 1 00:58:39.059:rtpspi_free_rtcp_session:Success. Leaving
*Mar 1 00:58:39.063:rtpspi_call_cleanup:leaving
*Mar 1 00:58:39.063:rtpspi_do_call_disconnect:leaving

```

**Related Commands**

Command	Description
<b>debug rtpspi all</b>	Debugs all RTP SPI errors, sessions, and in/out functions.
<b>debug rtpspi errors</b>	Debugs RTP SPI errors.
<b>debug rtpspi send-nse</b>	Triggers the RTP SPI to send a triple redundant NSE.
<b>debug sgcp errors</b>	Debugs SGCP errors.
<b>debug sgcp events</b>	Debugs SGCP events.
<b>debug sgcp packet</b>	Debugs SGCP packets.
<b>debug vtsp send-nse</b>	Sends and debugs a triple redundant NSE from the DSP to a remote gateway.

# debug rtpspi send-nse

To trigger the Routing Table Protocol (RTP) security parameter index (SPI) software module to send a triple redundant NSE, use the **debug rtpspi send-nse** command in privileged EXEC mode. To disable this action, use the **no** form of the command.

**debug rtpspi send-nse** *call-ID NSE-event-ID*

**no debug rtpspi send-nse** *call-ID NSE-event-ID*

## Syntax Description

<i>call-ID</i>	Specifies the call ID of the active call. The valid range is from 0 to 65535.
<i>NSE-event-ID</i>	Specifies the NSE Event ID. The valid range is from 0 to 255.

## Defaults

No default behavior or values

## Command Modes

Privileged EXEC

## Command History

Release	Modification
12.0(7)XK	This command was introduced on the Cisco MC3810 device and Cisco 3600 series routers (except the Cisco 3620 router).
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 example shows the RTP SPI software module set to send an NSE:

```
Router# debug rtpspi send-nse
```

## Related Commands

Command	Description
<b>debug rtpspi all</b>	Debugs all RTP SPI errors, sessions, and in/out functions.
<b>debug rtpspi errors</b>	Debugs RTP SPI errors.
<b>debug rtpspi inout</b>	Debugs RTP SPI in/out functions.
<b>debug sgcp errors</b>	Debugs SGCP errors.
<b>debug sgcp events</b>	Debugs SGCP events.
<b>debug sgcp packet</b>	Debugs SGCP packets.
<b>debug vtsp send-nse</b>	Sends and debugs a triple redundant NSE from the DSP to a remote gateway.

# debug rtpspi session

To debug all Routing Table Protocol (RTP) security parameter index (SPI) sessions, use the **debug rtpspi session** command in privileged EXEC mode. To disable debugging, use the **no** form of this command.

**debug rtpspi session**

**no debug rtpspi session**

**Syntax Description** This command has no arguments or keywords.

**Defaults** No default behavior or values

**Command Modes** Privileged EXEC

## Command History

Release	Modification
12.0(7)XK	This command was introduced on the Cisco MC3810 device and Cisco 3600 series routers (except the Cisco 3620 router).
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 example shows a debug trace for RTP SPI sessions on a gateway:

```
Router# debug rtpspi session

*Mar  1 01:01:51.593:rtpspi_allocate_rtp_port:allocated RTP port 16406
*Mar  1 01:01:51.593:rtpspi_call_setup:rtp_session_mode = 0x2
*Mar  1 01:01:51.593:rtpspi_call_setup:mode = CC_CALL_NORMAL.
      destination number = 0.0.0.0
*Mar  1 01:01:51.593:rtpspi_call_setup:Passed local_ip_addrs=0x5000001
*Mar  1 01:01:51.593:rtpspi_call_setup:Passed local_rtp_port = 16406
*Mar  1 01:01:51.593:rtpspi_call_setup:Saved RTCP Session = 0x1AFDFBC
*Mar  1 01:01:51.593:rtpspi_call_setup:Passed remote rtp port = 0.
*Mar  1 01:01:51.598:rtpspi_start_rtcp_session:Starting RTCP session.
      Local IP addr = 0x5000001, Remote IP addr = 0x0,
      Local RTP port = 16406, Remote RTP port = 0, mode = 0x2
*Mar  1 01:01:51.598:rtpspi_start_rtcp_session:RTP Session creation Success.
*Mar  1 01:01:51.598:rtpspi_call_setup:RTP Session creation Success.
*Mar  1 01:01:51.598:rtpspi_call_setup:calling cc_api_call_connected()
*Mar  1 01:01:51.598:rtpspi_modify_rtcp_session_parameters():xmit fn=0xDA7494,
dstIF=0x1964EEC, dstCallID=10, voip_mode=0x2, rtp_mode=0x2, ssrc_status=0
*Mar  1 01:01:51.598:rtpspi_bridge:Calling cc_api_bridge_done() for 11(0x1AF5400) and
10(0x0).
*Mar  1 01:01:51.602:rtpspi_caps_ind:caps from VTSP:codec=0x83FB, codec_bytes=0x50,
      fax_rate=0x7F, vad=0x3 modem=0x0
*Mar  1 01:01:51.602:rtpspi_get_rtcp_session_parameters():CURRENT VALUES:
dstIF=0x1964EEC, dstCallID=10, current_seq_num=0x0
```

```

*Mar 1 01:01:51.602:rtpspi_get_rtcp_session_parameters():NEW VALUES:
dstIF=0x1964EEC, dstCallID=10, current_seq_num=0xF1E
*Mar 1 01:01:51.602:rtpspi_caps_ind:Caps Used:codec=0x1, codec bytes=80,
    fax rate=0x1, vad=0x1, modem=0x1, dtmf_relay=0x1, seq_num_start=0xF1F
*Mar 1 01:01:51.602:rtpspi_caps_ind:calling cc_api_caps_ind().
*Mar 1 01:01:51.822:rtpspi_do_call_modify:Remote RTP port changed. New port=16498
*Mar 1 01:01:51.822:rtpspi_do_call_modify:Remote IP addr changed. New IP addr=0x6000001
*Mar 1 01:01:51.822:rtpspi_do_call_modify:Starting new RTCP session.
*Mar 1 01:01:51.822:rtpspi_start_rtcp_session:Removing old RTCP session.
*Mar 1 01:01:51.822:rtpspi_start_rtcp_session:Starting RTCP session.
    Local IP addr = 0x5000001, Remote IP addr = 0x6000001,
    Local RTP port = 16406, Remote RTP port = 16498, mode = 0x2
*Mar 1 01:01:51.822:rtpspi_start_rtcp_session:RTCP Timer creation Success. (5)*(5000)
*Mar 1 01:01:51.826:rtpspi_start_rtcp_session:RTP Session creation Success.
*Mar 1 01:01:51.826:rtpspi_do_call_modify:RTP Session creation Success.
*Mar 1 01:01:51.826:rtpspi_do_call_modify:Calling cc_api_call_modify(), result=0x0
*Mar 1 01:01:57.296:rtpspi_do_call_modify:Mode changed. new = 3, old = 2
*Mar 1 01:01:57.296:rtpspi_modify_rtcp_session_parameters():xmit fn=0xDA7494,
dstIF=0x1964EEC, dstCallID=10, voip_mode=0x3, rtp_mode=0x3, ssrc_status=2
*Mar 1 01:01:57.296:rtpspi_do_call_modify:RTCP Timer start.
*Mar 1 01:01:57.296:rtpspi_do_call_modify:Calling cc_api_call_modify(), result=0x0
*Mar 1 01:03:06.108:rtpspi_modify_rtcp_session_parameters():xmit fn=0x0,
dstIF=0x0, dstCallID=0, voip_mode=0x3, rtp_mode=0x3, ssrc_status=2
*Mar 1 01:03:06.112:rtpspi_do_call_disconnect:calling rtpspi_call_cleanup(). call-id=11
*Mar 1 01:03:06.112:rtpspi_call_cleanup:releasing ccb cache. RTP port=16406
*Mar 1 01:03:06.112:rtpspi_call_cleanup:RTCP Timer Stop.
*Mar 1 01:03:06.112:rtpspi_call_cleanup:deallocating RTP port 16406.
*Mar 1 01:03:06.112::rtpspi_call_cleanup freeing ccb (0x1AF5400)

```

**Related Commands**

Command	Description
<b>debug rtpspi all</b>	Debugs all RTP SPI errors, sessions, and in/out functions.
<b>debug rtpspi errors</b>	Debugs RTP SPI errors.
<b>debug rtpspi inout</b>	Debugs RTP SPI in/out functions.
<b>debug rtpspi send-nse</b>	Triggers the RTP SPI to send a triple redundant NSE.
<b>debug sgcp errors</b>	Debugs SGCP errors.
<b>debug sgcp events</b>	Debugs SGCP events.
<b>debug sgcp packet</b>	Debugs SGCP packets.
<b>sgcp</b>	Starts and allocates resources for the SCGP daemon.
<b>debug vtsp send-nse</b>	Sends and debugs a triple redundant NSE from the DSP to a remote gateway.

# debug rtr error



## Note

Effective with Cisco IOS Release 12.2(31)SB2, the **debug rtr error** command is replaced by the **debug ip sla monitor error** command. Effective with Cisco IOS Release 12.2(33)SRB, the **debug rtr error** command is replaced by the **debug ip sla error** command. See the **debug ip sla monitor error** and **debug ip sla error** commands for more information.

To enable debugging output of Cisco IOS IP Service Level Agreements (SLAs) operation run-time errors, use the **debug rtr error** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

```
debug rtr error [operation-number]
```

```
no debug rtr 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
11.2	This command was introduced.
12.0(5)T	This command was modified.
12.3(14)T	This command was replaced by the <b>debug ip sla monitor error</b> command.
12.2(31)SB2	This command was replaced by the <b>debug ip sla monitor error</b> command.
12.2(33)SRB	This command was replaced by the <b>debug ip sla error</b> command.
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

The **debug rtr 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 rtr error** command before using the **debug rtr trace** command because the **debug rtr error** command generates a lesser amount of debugging output.

**Examples**

The following is sample output from the **debug rtr 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 rtr trace** command) has the format shown in [Table 299](#).

```
Router# debug rtr 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
```

[Table 299](#) describes the significant fields shown in the display.

**Table 299** *debug rtr error Field Descriptions*

Field	Description
RTR 1	Number of the operation generating the message.
Error Return Code	Message identifier indicating the error type (or error itself).
LU0 RTR 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
<b>debug rtr trace</b>	Traces the execution of an IP SLAs operation.

# debug rtr mpls-lsp-monitor



## Note

Effective with Cisco IOS Release 12.2(31)SB2, the **debug rtr mpls-lsp-monitor** command is replaced by the **debug ip sla monitor mpls-lsp-monitor** command. Effective with Cisco IOS Release 12.2(33)SRB, the **debug rtr mpls-lsp-monitor** command is replaced by the **debug ip sla mpls-lsp-monitor** command. See the **debug ip sla monitor mpls-lsp-monitor** and **debug ip sla mpls-lsp-monitor** commands for more information.

To enable debugging output for the IP Service Level Agreements (SLAs) label switched path (LSP) Health Monitor, use the **debug rtr mpls-lsp-monitor** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

```
debug rtr mpls-lsp-monitor [operation-number]
```

```
no debug rtr mpls-lsp-monitor [operation-number]
```

## Syntax Description

<i>operation-number</i>	(Optional) Number of the LSP Health Monitor operation for which the debugging output will be displayed.
-------------------------	---

## Command Default

Debug is not enabled.

## Command Modes

Privileged EXEC

## Command History

Release	Modification
12.2(27)SBC	This command was introduced.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(31)SB2	This command was replaced by the <b>debug ip sla monitor mpls-lsp-monitor</b> command.
12.2(33)SRB	This command was replaced by the <b>debug ip sla mpls-lsp-monitor</b> command.
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 rtr mpls-lsp-monitor** command. This output shows that three VPNs associated with router 10.10.10.8 (red, blue, and green) were discovered and that this information was added to the LSP Health Monitor scan queue. Also, since router 10.10.10.8 is a newly discovered Border Gateway Protocol (BGP) next hop neighbor, a new IP SLAs operation for router 10.10.10.8 (Probe 100005) is being created and added to the LSP Health Monitor multioperation schedule. Even though router 10.10.10.8 belongs to three VPNs, only one IP SLAs operation is being created.

```
Router# debug rtr mpls-lsp-monitor
```

```
SAA MPLSLM debugging for all entries is on
*Aug 19 19:59: SAA MPLSLM(1):Next hop 10.10.10.8 added in AddQ
*Aug 19 19:59: SAA MPLSLM(1):Next hop 10.10.10.8 added in AddQ
*Aug 19 19:59: SAA MPLSLM(1):Next hop 10.10.10.8 added in AddQ
*Aug 19 19:59: SAA MPLSLM(1):Adding vrf red into tree entry 10.10.10.8
*Aug 19 19:59: SAA MPLSLM(1):Adding Probe 100005
*Aug 19 19:59: SAA MPLSLM(1):Adding ProbeID 100005 to tree entry 10.10.10.8 (1)
*Aug 19 19:59: SAA MPLSLM(1):Adding vrf blue into tree entry 10.10.10.8
*Aug 19 19:59: SAA MPLSLM(1):Duplicate in AddQ 10.10.10.8
*Aug 19 19:59: SAA MPLSLM(1):Adding vrf green into tree entry 10.10.10.8
*Aug 19 19:59: SAA MPLSLM(1):Duplicate in AddQ 10.10.10.8
*Aug 19 19:59: SAA MPLSLM(1):Added Probe(s) 100005 will be scheduled after 26 secs over
schedule period 60
```

---

**Related Commands**

Command	Description
<b>rtr mpls-lsp-monitor</b>	Begins configuration for an IP SLAs LSP Health Monitor operation and enters SAA MPLS configuration mode.

---

# debug rtr trace



## Note

Effective with Cisco IOS Release 12.2(31)SB2, the **debug rtr trace** command is replaced by the **debug ip sla monitor trace** command. Effective with Cisco IOS Release 12.2(33)SRB, the **debug rtr trace** command is replaced by the **debug ip sla trace** command. See the **debug ip sla monitor trace** and **debug ip sla trace** commands for more information.

To trace the execution of a Cisco IOS IP Service Level Agreements (SLAs) operation, use the **debug rtr trace** command in privileged EXEC mode. To disable trace debugging output, use the **no** form of this command.

```
debug rtr trace [operation-number]
```

```
no debug rtr trace [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
11.2	This command was introduced.
12.0(5)T	This command was modified.
12.3(14)T	This command was replaced by the <b>debug ip sla monitor trace</b> command.
12.2(31)SB2	This command was replaced by the <b>debug ip sla monitor trace</b> command.
12.2(33)SRB	This command was replaced by the <b>debug ip sla trace</b> command.
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

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 rtr trace** command also enables **debug rtr error** command for the specified operation. However, the **no debug rtr trace** command does not disable the **debug rtr error** command. You must manually disable the command by using the **no debug rtr error** command.

All debugging output (including **debug rtr error** command output) has the format shown in the **debug rtr error** command output example.

**Note**

The **debug rtr trace** command can generate a large number of debug messages. First use the **debug rtr error** command, and then use the **debug rtr trace** on a per-operation basis.

**Examples**

The following is sample output from the **debug rtr 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 rtr trace
```

```
Router# RTR 1:Starting An Echo Operation - IP RTR Probe 1
```

```
May 5 05:25:08.584:rtr 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:rtr hash remove:3.0.0.3 3383RTR 1:Starting An Echo Operation - IP RTR
Probe 1

May 5 05:26:08.104:rtr 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:rtr hash remove:3.0.0.3 2974RTR 1:Starting An Echo Operation - IP RTR
Probe 1
```

**Related Commands**

Command	Description
<b>debug rtr error</b>	Enables debugging output of IP SLAs operation run-time errors.

# debug rtsp

To show the status of the Real-Time Streaming Protocol (RTSP) client or server, use the **debug rtsp** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug rtsp** *type* [**all** | **api** | **error** | **pmh** | **session** | **socket**]

[**no**] **debug rtsp** *type* [**all** | **api** | **pmh** | **session** | **socket**]

## Syntax Description

<i>type</i>	Type of debug messages to display. The keywords are as follows: <ul style="list-style-type: none"> <li><b>all</b>—(Optional) Displays debug output for all clients or servers.</li> <li><b>api</b>—(Optional) Displays debug output for the client or server API.</li> <li><b>error</b>—(Optional) Displays errors when they are errors otherwise no output is displayed.</li> <li><b>pmh</b>—(Optional) Displays debug output for the Protocol Message Handler (PMH).</li> <li><b>session</b>—(Optional) Displays debug output for the client or server session.</li> <li><b>socket</b>—(Optional) Displays debug output for the client or server socket data.</li> </ul>
-------------	--

## Defaults

This command is disabled by default.

## Command Modes

Privileged EXEC (#)

## Command History

Release	Modification
12.1(3)T	This command was introduced.
12.2(11)T	The new debug header was added to the following Cisco routers: Cisco 2600 series, Cisco 3620, Cisco 3640, and Cisco 3660 series; on the following universal gateways: Cisco AS5350, Cisco AS5400, and Cisco AS5850; on the following access servers: Cisco AS5300, and Cisco AS5800; and, on the Cisco MC3810 multiservice access concentrators.
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 that displays when the **debug rtsp** command is entered with the **api** keyword:

```
Router# debug rtsp api
```

```

!
RTSP client API debugging is on
!
Jan  1 00:23:15.775:rtsp_api_create_session:sess_id=0x61A07C78, evh=0x60D6E62C
context=0x61A07B28
Jan  1 00:23:15.775:rtsp_api_request:msg=0x61C2B10C
Jan  1 00:23:15.775:rtsp_api_handle_req_set_params:msg=0x61C2B10C
Jan  1 00:23:15.775:rtsp_api_free_msg_buffer:msg=0x61C2B10C
Jan  1 00:23:15.775:rtsp_api_request:msg=0x61C293CC
Jan  1 00:23:15.775:rtsp_api_handle_req_set_params:msg=0x61C293CC
Jan  1 00:23:15.775:rtsp_api_free_msg_buffer:msg=0x61C293CC
Jan  1 00:23:15.775:rtsp_api_request:msg=0x61C2970C
Jan  1 00:23:15.775:rtsp_api_handle_req_set_params:msg=0x61C2970C
Jan  1 00:23:15.775:rtsp_api_free_msg_buffer:msg=0x61C2970C
!
Jan  1 00:23:15.775:rtsp_api_request:msg=0x61C29A4C
!
Jan  1 00:23:22.099:rtsp_api_free_msg_buffer:msg=0x61C29A4C
Jan  1 00:23:22.115:rtsp_api_request:msg=0x61C2A40C
Jan  1 00:23:22.115:rtsp_api_free_msg_buffer:msg=0x61C2A40C

```

**Related Commands**

Command	Description
<b>debug rtsp api</b>	Displays debug output for the RTSP client API.
<b>debug rtsp client session</b>	Displays debug output for the RTSP client data.
<b>debug rtsp socket</b>	Displays debug output for the RTSP client socket data.

# debug rtsp all

To display all related information about the Real Time Streaming Protocol (RTSP) data, use the **debug rtsp all** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug rtsp all**

**no debug rtsp all**

**Syntax Description** This command has no arguments or keywords.

**Defaults** Debug is not enabled.

**Command Modes** Privileged EXEC

## Command History

Release	Modification
12.1(3)T	This command was introduced.
12.2(11)T	This command was implemented on the following platforms: Cisco 2600 series, Cisco 3620, Cisco 3640, and Cisco 3660, Cisco AS5350, Cisco AS5400, Cisco AS5850, Cisco AS5300, Cisco AS5800, and Cisco MC3810.
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

We recommend that you log output from the **debug rtsp all** command to a buffer rather than sending the output to the console; otherwise, the size of the output could severely impact the performance of the gateway.

## Examples

The following example shows debugging output for the **debug rtsp all** command. The **show debug** command shows which RTSP modules are traced.

```
Router# debug rtsp all
```

```
All RTSP client debugging is on
```

```
Router# show debug
```

```
RTSP:
```

```
RTSP client Protocol Error debugging is on
```

```
RTSP client Protocol Message Handler debugging is on
```

```
RTSP client API debugging is on
```

```
RTSP client socket debugging is on
```

```
RTSP client session debugging is on
```

```

Router#
Router#!call initiated
Router#
*Mar 11 03:14:23.471: //-1//RTSP:/rtsp_get_new_scb:
*Mar 11 03:14:23.471: //-1//RTSP:/rtsp_initialize_scb:
*Mar 11 03:14:23.471: //-1//RTSP:/rtsplib_init_svr_session: 0x63A5FE6C
*Mar 11 03:14:23.471: //-1//RTSP:/rtsp_api_create_session: evh=0x6155F0D4
context=0x6345042C
*Mar 11 03:14:23.471: //-1//RTSP:/rtsp_get_new_scb:
*Mar 11 03:14:23.471: //-1//RTSP:/rtsp_initialize_scb:
*Mar 11 03:14:23.471: //-1//RTSP:/rtsplib_init_svr_session: 0x63A5D874
*Mar 11 03:14:23.471: //-1//RTSP:/rtsp_api_create_session: evh=0x6155F204
context=0x6345046C
*Mar 11 03:14:23.471: //-1//RTSP:RS45:/rtsp_api_request: msg=0x63A59FB8
*Mar 11 03:14:23.471: //-1//RTSP:RS45:/rtsp_api_handle_req_set_params: msg=0x63A59FB8
*Mar 11 03:14:23.475: //-1//RTSP:RS45:/rtsp_api_free_msg_buffer: msg=0x63A59FB8
*Mar 11 03:14:23.475: //-1//RTSP:RS45:/rtsp_api_request: msg=0x63A5A304
*Mar 11 03:14:23.475: //-1//RTSP:RS45:/rtsp_api_handle_req_set_params: msg=0x63A5A304
*Mar 11 03:14:23.475: //-1//RTSP:RS45:/rtsp_api_free_msg_buffer: msg=0x63A5A304
*Mar 11 03:14:23.475: //-1//RTSP:RS45:/rtsp_api_request: msg=0x63A5A650
*Mar 11 03:14:23.475: //-1//RTSP:RS45:/rtsp_api_handle_req_set_params: msg=0x63A5A650
*Mar 11 03:14:23.475: //166//RTSP:LP:RS45:/rtsp_api_handle_req_set_params:
*Mar 11 03:14:23.475: //-1//RTSP:RS45:/rtsp_api_free_msg_buffer: msg=0x63A5A650
*Mar 11 03:14:23.475: //-1//RTSP:RS46:/rtsp_api_request: msg=0x63A5A99C
*Mar 11 03:14:23.475: //-1//RTSP:RS46:/rtsp_api_handle_req_set_params: msg=0x63A5A99C
*Mar 11 03:14:23.475: //166//RTSP:LP:RS46:/rtsp_api_handle_req_set_params:
*Mar 11 03:14:23.475: //-1//RTSP:RS46:/rtsp_api_free_msg_buffer: msg=0x63A5A99C
Router#
Router#!call answered
Router#
Router#!digits dialed
Router#
Router#!call terminated
Router#
*Mar 11 03:14:51.603: //-1//RTSP:RS45:/rtsp_api_request: msg=0x63A5ACE8
*Mar 11 03:14:51.603: //-1//RTSP:RS46:/rtsp_api_request: msg=0x63A5B034
*Mar 11 03:14:51.607: //-1//RTSP:RS45:/rtsp_control_process_msg:
*Mar 11 03:14:51.607: //166//RTSP:/rtsp_control_process_msg: received MSG request of TYPE
0
*Mar 11 03:14:51.607: //166//RTSP:/rtsp_set_event: api_req_msg_type=RTSP_API_REQ_DESTROY
*Mar 11 03:14:51.607: //166//RTSP:/rtsp_session_cleanup:
*Mar 11 03:14:51.607: //-1//RTSP:/rtsplib_free_svr_session:
*Mar 11 03:14:51.607: //-1//RTSP:/rtsplib_stop_timer: timer(0x638D5DDC) stops
*Mar 11 03:14:51.611: //166//RTSP:/rtsp_create_session_history: scb=0x63A5FE6C,
callID=0xA6
*Mar 11 03:14:51.611: //166//RTSP:/rtsp_create_session_history: No streams in session
control block
*Mar 11 03:14:51.611: //166//RTSP:/rtsp_session_cleanup: deleting session: scb=0x63A5FE6C
*Mar 11 03:14:51.611: //-1//RTSP:RS45:/rtsp_api_free_msg_buffer: msg=0x63A5ACE8
*Mar 11 03:14:51.611: //-1//RTSP:RS46:/rtsp_control_process_msg:
*Mar 11 03:14:51.611: //166//RTSP:/rtsp_control_process_msg: received MSG request of TYPE
0
*Mar 11 03:14:51.611: //166//RTSP:/rtsp_set_event: api_req_msg_type=RTSP_API_REQ_DESTROY
*Mar 11 03:14:51.611: //166//RTSP:/rtsp_session_cleanup:
*Mar 11 03:14:51.611: //-1//RTSP:/rtsplib_free_svr_session:
*Mar 11 03:14:51.611: //-1//RTSP:/rtsplib_stop_timer: timer(0x63A60110) stops
*Mar 11 03:14:51.611: //166//RTSP:/rtsp_create_session_history: scb=0x63A5D874,
callID=0xA6
*Mar 11 03:14:51.611: //166//RTSP:/rtsp_create_session_history: No streams in session
control block
*Mar 11 03:14:51.611: //166//RTSP:/rtsp_session_cleanup: deleting session: scb=0x63A5D874
*Mar 11 03:14:51.611: //-1//RTSP:RS46:/rtsp_api_free_msg_buffer: msg=0x63A5B034

```

Table 300 describes the significant fields shown in the display.

**Table 300**      *debug rtsp all* Field Descriptions

Field	Description
//-1/	Indicates that the CallEntry ID for the module is unavailable.
//166/	Identifies the CallEntry ID.
RTSP:	Identifies the RTSP module.
<i>rtsp_function name</i>	Identifies the function name.

#### Related Commands

Command	Description
<b>debug rtsp api</b>	Displays debugging output for the RTSP client API.
<b>debug rtsp error</b>	Displays error message for RTSP data.
<b>debug rtsp pmh</b>	Displays debugging messages for the PMH.
<b>debug rtsp socket</b>	Displays debugging output for the RTSP client socket data.
<b>voice call debug</b>	Allows configuration of the voice call debugging output.

# debug rtsp api

To display information about the Real Time Streaming Protocol (RTSP) application programming interface (API) messages passed down to the RTSP client, use the **debug rtsp api** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug rtsp api**

**no debug rtsp api**

## Syntax Description

This command has no arguments or keywords.

## Defaults

Debug is not enabled.

## Command Modes

Privileged EXEC

## Command History

Release	Modification
12.1(3)T	This command was introduced.
12.2(11)T	This command was implemented on the following platforms: Cisco 2600 series, Cisco 3620, Cisco 3640, and Cisco 3660, Cisco AS5350, Cisco AS5400, Cisco AS5850, Cisco AS5300, Cisco AS5800, and Cisco MC3810.
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

We recommend that you log output from the **debug rtsp api** command to a buffer rather than sending the output to the console; otherwise, the size of the output could severely impact the performance of the gateway.

## Examples

The following example shows output from the **debug rtsp api** command:

```
Router# debug rtsp api

RTSP client API debugging is on

Router# !call initiated

*Mar 11 03:04:41.699: //-1//RTSP:/rtsp_api_create_session: evh=0x6155F0D4
context=0x6345088C
*Mar 11 03:04:41.699: //-1//RTSP:/rtsp_api_create_session: evh=0x6155F204
context=0x634508CC
*Mar 11 03:04:41.699: //-1//RTSP:RS35:/rtsp_api_request: msg=0x63A59FB8
*Mar 11 03:04:41.699: //-1//RTSP:RS35:/rtsp_api_handle_req_set_params: msg=0x63A59FB8
*Mar 11 03:04:41.699: //-1//RTSP:RS35:/rtsp_api_free_msg_buffer: msg=0x63A59FB8
*Mar 11 03:04:41.699: //-1//RTSP:RS35:/rtsp_api_request: msg=0x63A5A304
```

```
*Mar 11 03:04:41.699: //-1//RTSP:RS35:/rtsp_api_handle_req_set_params: msg=0x63A5A304
*Mar 11 03:04:41.699: //-1//RTSP:RS35:/rtsp_api_free_msg_buffer: msg=0x63A5A304
*Mar 11 03:04:41.703: //-1//RTSP:RS35:/rtsp_api_request: msg=0x63A5A650
*Mar 11 03:04:41.703: //-1//RTSP:RS35:/rtsp_api_handle_req_set_params: msg=0x63A5A650
*Mar 11 03:04:41.703: //146//RTSP:LP:RS35:/rtsp_api_handle_req_set_params:
*Mar 11 03:04:41.703: //-1//RTSP:RS35:/rtsp_api_free_msg_buffer: msg=0x63A5A650
*Mar 11 03:04:41.703: //-1//RTSP:RS36:/rtsp_api_request: msg=0x63A5A99C
*Mar 11 03:04:41.703: //-1//RTSP:RS36:/rtsp_api_handle_req_set_params: msg=0x63A5A99C
*Mar 11 03:04:41.703: //146//RTSP:LP:RS36:/rtsp_api_handle_req_set_params:
*Mar 11 03:04:41.703: //-1//RTSP:RS36:/rtsp_api_free_msg_buffer: msg=0x63A5A99C
```

Router!call answered

Router#!digits dialed

Router#!call terminated

```
*Mar 11 03:05:15.367: //-1//RTSP:RS35:/rtsp_api_request: msg=0x63A5ACE8
*Mar 11 03:05:15.367: //-1//RTSP:RS36:/rtsp_api_request: msg=0x63A5B034
*Mar 11 03:05:15.367: //-1//RTSP:RS35:/rtsp_api_free_msg_buffer: msg=0x63A5ACE8
*Mar 11 03:05:15.367: //-1//RTSP:RS36:/rtsp_api_free_msg_buffer: msg=0x63A5B034
```

Table 301 describes the significant fields shown in the display.

**Table 301** *debug rtsp api Field Descriptions*

Field	Description
//-1/	Indicates that the CallEntry ID for the module is unavailable.
//146/	Identifies the CallEntry ID.
RTSP:	Identifies the RTSP module.
rtsp_ function name	Identifies the function name.

#### Related Commands

Command	Description
debug rtsp error	Displays error message for RTSP data.
debug rtsp pmh	Displays debugging messages for the PMH.
debug rtsp socket	Displays debugging output for the RTSP client socket data.
voice call debug	Allows configuration of the voice call debugging output.

# debug rtsp client



## Note

Effective with Release 12.3(4), the **debug rtsp cleint** command is replaced by the **debug rtsp session** command. See the **debug rtsp session** command for more information.

To display client information and stream information for the stream that is currently active for the Real Time Streaming Protocol (RTSP) client, use the **debug rtsp client** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug rtsp client**

**no debug rtsp client**

## Syntax Description

This command has no arguments or keywords.

## Defaults

Debug is not enabled.

## Command Modes

Privileged EXEC

## Command History

Release	Modification
12.1(3)T	This command was introduced.
12.3(4)T	This command was replaced by the <b>debug rtsp session</b> command.
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

We recommend that you log output from the **debug rtsp client** command to a buffer rather than sending the output to the console; otherwise, the size of the output could severely impact the performance of the gateway.

## Related Commands

Command	Description
<b>debug rtsp api</b>	Displays debugging output for the RTSP client API.
<b>debug rtsp error</b>	Displays error message for RTSP data.
<b>debug rtsp pmh</b>	Displays debugging messages for the PMH.
<b>debug rtsp socket</b>	Displays debugging output for the RTSP client socket data.
<b>voice call debug</b>	Allows configuration of the voice call debugging output.

# debug rtsp client session



## Note

Effective with Release 12.3(4), the **debug rtsp cleint session** command is replaced by the **debug rtsp session** command. See the **debug rtsp session** command for more information.

To display debug messages about the Real Time Streaming Protocol (RTSP) client or the current session, use the **debug rtsp** command. To disable debugging output, use the **no** form of this command.

```
debug rtsp [client | session]
```

```
no debug rtsp [client | session]
```

## Syntax Description

<b>client</b>	(Optional) Displays client information and stream information for the stream that is currently active.
<b>session</b>	(Optional) Displays cumulative information about the session, packet statistics, and general call information such as call ID, session ID, individual RTSP stream URLs, packet statistics, and play duration.

## Defaults

Debug is not enabled.

## Command History

Release	Modification
12.1(3)T	This command was introduced.
12.3(4)T	This command was replaced by the <b>debug rtsp session</b> command.
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 example displays the debug messages of the RTSP session:

```
Router# debug rtsp session

RTSP client session debugging is on
router#
Jan  1 00:08:36.099:rtsp_get_new_scb:
Jan  1 00:08:36.099:rtsp_initialize_scb:
Jan  1 00:08:36.099:rtsp_control_process_msg:
Jan  1 00:08:36.099:rtsp_control_process_msg:received MSG request of TYPE 0
Jan  1 00:08:36.099:rtsp_set_event:
Jan  1 00:08:36.099:rtsp_set_event:api_req_msg_type=RTSP_API_REQ_PLAY
Jan  1 00:08:36.103:rtsp_set_event:url:[rtsp://rtsp-cisco.cisco.com:554/en_welcome.au]
Jan  1 00:08:36.103:rtsp_process_async_event:SCB=0x62128F08
Jan  1 00:08:36.103:rtsp_process_async_event:rtsp_state = RTSP_SES_STATE_IDLE
                    rtsp_event = RTSP_EV_PLAY_OR_REC
Jan  1 00:08:36.103:act_idle_event_play_or_rec_req:
Jan  1 00:08:36.103:rtsp_resolve_dns:
Jan  1 00:08:36.103:rtsp_resolve_dns:IP Addr = 1.13.79.6:
```

```

Jan 1 00:08:36.103:rtsp_connect_to_svr:
Jan 1 00:08:36.103:rtsp_connect_to_svr:socket=0, connection_state = 2
Jan 1 00:08:36.103:rtsp_start_timer:timer (0x62128FD0)starts - delay (10000)
Jan 1 00:08:36.107:rtsp_control_main:SOCK= 0 Event=0x1
Jan 1 00:08:36.107:rtsp_stop_timer:timer(0x62128FD0) stops
Jan 1 00:08:36.107:rtsp_process_async_event:SCB=0x62128F08
Jan 1 00:08:36.107:rtsp_process_async_event:rtsp_state = RTSP_SES_STATE_IDLE
      rtsp_event = RTSP_EV_SVR_CONNECTED
Jan 1 00:08:36.107:act_idle_event_svr_connected:
Jan 1 00:08:36.107:rtsp_control_main:SOCK= 0 Event=0x1
Jan 1 00:08:36.783:rtsp_control_main:SOCK= 0 Event=0x1
Jan 1 00:08:36.783:rtsp_process_async_event:SCB=0x62128F08
Jan 1 00:08:36.783:rtsp_process_async_event:rtsp_state = RTSP_SES_STATE_READY
      rtsp_event = RTSP_EV_SVR_DESC_OR_ANNOUNCE_RESP
Jan 1 00:08:36.783:act_ready_event_desc_or_announce_resp:
Jan 1
00:08:36.783:act_ready_event_desc_or_announce_resp:RTSP_STATUS_DESC_OR_ANNOUNCE_RESP_OK
Jan 1 00:08:37.287:rtsp_control_main:SOCK= 0 Event=0x1
Jan 1 00:08:37.287:rtsp_process_async_event:SCB=0x62128F08
Jan 1 00:08:37.287:rtsp_process_async_event:rtsp_state = RTSP_SES_STATE_READY
      rtsp_event = RTSP_EV_SVR_SETUP_RESP
Jan 1 00:08:37.287:act_ready_event_setup_resp:
Jan 1 00:08:37.287:act_ready_event_setup_resp:Remote RTP Port=13344
Jan 1 00:08:37.287:rtsp_rtp_stream_setup:scb=0x62128F08, callID=0x7 record=0
Jan 1 00:08:37.287:rtsp_rtp_stream_setup:Starting RTCP session.
      Local IP addr = 1.13.79.45, Remote IP addr = 1.13.79.6,
      Local RTP port = 18748, Remote RTP port = 13344 CallID=8
Jan 1 00:08:37.291:xmit_func = 0x0 vdbptr = 0x61A0FC98
Jan 1 00:08:37.291:rtsp_control_main:CACAPI Queue Event
Jan 1 00:08:37.291:rtsp_rtp_associate_done:ev=0x62070E08, callID=0x7
Jan 1 00:08:37.291:rtsp_rtp_associate_done:scb=0x62128F08
Jan 1 00:08:37.291:rtsp_rtp_associate_done:callID=0x7, pVdb=0x61F4FBC8,
Jan 1 00:08:37.291:      spi_context=0x6214145C
Jan 1 00:08:37.291:      disposition=0, playFunc=0x60CA2238,
Jan 1 00:08:37.291:      codec=0x5, vad=0, mediaType=6,
Jan 1 00:08:37.291:      stream_assoc_id=1
Jan 1 00:08:37.291:rtsp_rtp_modify_session:scb=0x62128F08, callID=0x7
Jan 1 00:08:37.291:rtsp_process_async_event:SCB=0x62128F08
Jan 1 00:08:37.291:rtsp_process_async_event:rtsp_state = RTSP_SES_STATE_READY
      rtsp_event = RTSP_EV_ASSOCIATE_DONE
Jan 1 00:08:37.291:act_ready_event_associate_done:
Jan 1 00:08:37.291:rtsp_get_stream:
Jan 1 00:08:37.783:rtsp_control_main:SOCK= 0 Event=0x1
Jan 1 00:08:37.783:rtsp_process_async_event:SCB=0x62128F08
Jan 1 00:08:37.783:rtsp_process_async_event:rtsp_state = RTSP_SES_STATE_READY
      rtsp_event = RTSP_EV_SVR_PLAY_OR_REC_RESP
Jan 1 00:08:37.783:act_ready_event_play_or_rec_resp:
Jan 1 00:08:37.783:rtsp_start_timer:timer (0x62128FB0)starts - delay (4249)
rtsp-5#
Jan 1 00:08:42.035:rtsp_process_timer_events:
Jan 1 00:08:42.035:rtsp_process_timer_events:PLAY OR RECORD completed
Jan 1 00:08:42.035:rtsp_process_async_event:SCB=0x62128F08
Jan 1 00:08:42.035:rtsp_process_async_event:rtsp_state = RTSP_SES_STATE_PLAY_OR_REC
      rtsp_event = RTSP_EV_PLAY_OR_REC_TIMER_EXPIRED
Jan 1 00:08:42.035:act_play_event_play_done:
Jan 1 00:08:42.035:act_play_event_play_done:elapsed play time = 4249 total play time =
4249
Jan 1 00:08:42.035:rtsp_send_teardown_to_svr:
Jan 1 00:08:42.487:rtsp_control_main:SOCK= 0 Event=0x1
Jan 1 00:08:42.487:rtsp_process_async_event:SCB=0x62128F08
Jan 1 00:08:42.487:rtsp_process_async_event:rtsp_state = RTSP_SES_STATE_PLAY_OR_REC
      rtsp_event = RTSP_EV_SVR_TEARDOWN_RESP
Jan 1 00:08:42.487:act_play_event_teardown_resp:
Jan 1 00:08:42.487:rtsp_server_closed:

```

```

Jan 1 00:08:42.487:rtsp_send_resp_to_api:
Jan 1 00:08:42.487:rtsp_send_resp_to_api:sending RESP=RTSP_STATUS_PLAY_COMPLETE
Jan 1 00:08:42.491:rtsp_rtp_teardown_stream:scb=0x62128F08, callID=0x7
Jan 1 00:08:42.491:rtsp_rtp_stream_cleanup:scb=0x62128F08, callID=0x7
Jan 1 00:08:42.491:rtsp_update_stream_stats:scb=0x62128F08, stream=0x61A43350,
Jan 1 00:08:42.491:call_info=0x6214C67C, callID=0x7
Jan 1 00:08:42.491:rtsp_update_stream_stats:rx_bytes = 25992
Jan 1 00:08:42.491:rtsp_update_stream_stats:rx_packetes = 82
Jan 1 00:08:42.491:rtsp_reinitialize_scb:
Jan 1 00:08:42.503:rtsp_control_process_msg:
Jan 1 00:08:42.503:rtsp_control_process_msg:received MSG request of TYPE 0
Jan 1 00:08:42.503:rtsp_set_event:
Jan 1 00:08:42.503:rtsp_set_event:api_req_msg_type=RTSP_API_REQ_DESTROY
Jan 1 00:08:42.503:rtsp_session_cleanup:
Jan 1 00:08:42.503:rtsp_create_session_history:scb=0x62128F08, callID=0x7
Jan 1 00:08:42.503:rtsp_insert_session_history_record:current=0x6214BDC8, callID=0x7
Jan 1 00:08:42.503:rtsp_insert_session_history_record:count = 3
Jan 1 00:08:42.503:rtsp_insert_session_history_record:starting history record
deletion_timer of10 minutes
Jan 1 00:08:42.503:rtsp_session_cleanup:deleting session:scb=0x62128F08
Router#

```

**Related Commands**

Command	Description
<b>debug rtsp all</b>	Displays debugging output for the RTSP client API.
<b>debug rtsp pmh</b>	Displays debugging messages for the PMH.
<b>debug rtsp socket</b>	Displays debugging output for the RTSP client socket data.

# debug rtsp error

To display error information about the Real-Time Streaming Protocol (RTSP) client, use the **debug rtsp error** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug rtsp error**

**no debug rtsp error**

## Syntax Description

This command has no arguments or keywords.

## Defaults

Debug is not enabled.

## Command Modes

Privileged EXEC

## Command History

Release	Modification
12.1(3)T	This command was introduced.
12.2(11)T	This command was implemented on the following platforms: Cisco 2600 series, Cisco 3620, Cisco 3640, and Cisco 3660, Cisco AS5350, Cisco AS5400, Cisco AS5850, Cisco AS5300, Cisco AS5800, and Cisco MC3810.
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

We recommend that you log output from the **debug rtsp error** command to a buffer rather than sending the output to the console; otherwise, the size of the output could severely impact the performance of the gateway.

## Related Commands

Command	Description
<b>debug rtsp api</b>	Displays debugging output for the RTSP client API.
<b>debug rtsp pmh</b>	Displays debugging messages for the PMH.
<b>debug rtsp socket</b>	Displays debugging output for the RTSP client socket data.
<b>voice call debug</b>	Allows configuration of the voice call debugging output.

# debug rtsp pmh

To display debugging information about the Protocol Message Handler (PMH), use the **debug rtsp pmh** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug rtsp pmh**

**no debug rtsp pmh**

**Syntax Description** This command has no arguments or keywords.

**Defaults** Debug is not enabled.

**Command Modes** Privileged EXEC

## Command History

Release	Modification
12.1(3)T	This command was introduced.
12.2(11)T	This command was implemented on the following platforms: Cisco 2600 series, Cisco 3620, Cisco 3640, and Cisco 3660, Cisco AS5350, Cisco AS5400, Cisco AS5850, Cisco AS5300, Cisco AS5800, and Cisco MC3810.
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

We recommend that you log output from the **debug rtsp pmh** command to a buffer rather than sending the output to the console; otherwise, the size of the output could severely impact the performance of the gateway.

## Related Commands

Command	Description
<b>debug rtsp api</b>	Displays debugging output for the RTSP client API.
<b>debug rtsp error</b>	Displays error message for RTSP data.
<b>debug rtsp socket</b>	Displays debugging output for the RTSP client socket data.
<b>voice call debug</b>	Allows configuration of the voice call debugging output.

# debug rtsp session

To display client information and stream information for the stream that is currently active for the Real Time Streaming Protocol (RTSP) client, use the **debug rtsp session** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug rtsp session**

**no debug rtsp session**

## Syntax Description

This command has no arguments or keywords.

## Defaults

Debug is not enabled.

## Command Modes

Privileged EXEC

## Command History

Release	Modification
12.1(3)T	This command was introduced.
12.2(11)T	This command was implemented on the following platforms: Cisco 2600 series, Cisco 3620, Cisco 3640, and Cisco 3660, Cisco AS5350, Cisco AS5400, Cisco AS5850, Cisco AS5300, Cisco AS5800, and Cisco MC3810.
12.3(4)T	This command replaces the <b>debug rtsp client</b> command and the <b>debug rtsp client session</b> command.
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

We recommend that you log output from the **debug rtsp session** command to a buffer rather than sending the output to the console; otherwise, the size of the output could severely impact the performance of the gateway.

## Examples

The following example shows the display of the debugging messages of the RTSP session:

```
Router# debug rtsp session

RTSP client session debugging is on
Router#
Router#!call initiated
Router#
*Mar 11 03:09:58.123: //-1//RTSP:/rtsp_get_new_scb:
*Mar 11 03:09:58.123: //-1//RTSP:/rtsp_initialize_scb:
*Mar 11 03:09:58.123: //-1//RTSP:/rtsplib_init_svr_session: 0x63A5FE6C
*Mar 11 03:09:58.123: //-1//RTSP:/rtsp_get_new_scb:
```

```

*Mar 11 03:09:58.123: //-1//RTSP:/rtsp_initialize_scb:
*Mar 11 03:09:58.123: //-1//RTSP:/rtsplib_init_svr_session: 0x63A5D874
Router#
Router#!call answered
Router#
Router#!digits dialed
Router#
Router#!call terminated
Router#
*Mar 11 03:10:38.139: //-1//RTSP:RS41:/rtsp_control_process_msg:
*Mar 11 03:10:38.139: //158//RTSP:/rtsp_control_process_msg: received MSG request of TYPE
0
*Mar 11 03:10:38.139: //158//RTSP:/rtsp_set_event: api_req_msg_type=RTSP_API_REQ_DESTROY
*Mar 11 03:10:38.139: //158//RTSP:/rtsp_session_cleanup:
*Mar 11 03:10:38.139: //-1//RTSP:/rtsplib_free_svr_session:
*Mar 11 03:10:38.139: //-1//RTSP:/rtsplib_stop_timer: timer(0x638D5DDC) stops
*Mar 11 03:10:38.143: //158//RTSP:/rtsp_create_session_history: scb=0x63A5FE6C,
callID=0x9E
*Mar 11 03:10:38.143: //158//RTSP:/rtsp_create_session_history: No streams in session
control block
*Mar 11 03:10:38.143: //158//RTSP:/rtsp_session_cleanup: deleting session: scb=0x63A5FE6C
*Mar 11 03:10:38.143: //-1//RTSP:RS42:/rtsp_control_process_msg:
*Mar 11 03:10:38.143: //158//RTSP:/rtsp_control_process_msg: received MSG request of TYPE
0
*Mar 11 03:10:38.143: //158//RTSP:/rtsp_set_event: api_req_msg_type=RTSP_API_REQ_DESTROY
*Mar 11 03:10:38.143: //158//RTSP:/rtsp_session_cleanup:
*Mar 11 03:10:38.143: //-1//RTSP:/rtsplib_free_svr_session:
*Mar 11 03:10:38.143: //-1//RTSP:/rtsplib_stop_timer: timer(0x63A60110) stops
*Mar 11 03:10:38.143: //158//RTSP:/rtsp_create_session_history: scb=0x63A5D874,
callID=0x9E
*Mar 11 03:10:38.143: //158//RTSP:/rtsp_create_session_history: No streams in session
control block
*Mar 11 03:10:38.143: //158//RTSP:/rtsp_session_cleanup: deleting session: scb=0x63A5D874

```

Table 302 describes the significant fields shown in the display.

**Table 302** *debug rtsp session Field Descriptions*

Field	Description
//-1/	Indicates that the CallEntry ID for the module is unavailable.
//158/	Identifies the CallEntry ID.
RTSP:	Identifies the RTSP module.
rtsp_<function name>	Identifies the function name.

#### Related Commands

Command	Description
<b>debug rtsp api</b>	Displays debugging output for the RTSP client API.
<b>debug rtsp error</b>	Displays error message for RTSP data.
<b>debug rtsp pmh</b>	Displays debugging messages for the PMH.
<b>debug rtsp socket</b>	Displays debugging output for the RTSP client socket data.
<b>voice call debug</b>	Allows configuration of the voice call debugging output.

# debug rtsp socket

To display debugging messages about the packets received or sent on the TCP or User Datagram Protocol (UDP) sockets, use the **debug rtsp socket** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug rtsp socket**

**no debug rtsp socket**

## Syntax Description

This command has no arguments or keywords.

## Defaults

Debug is not enabled.

## Command Modes

Privileged EXEC

## Command History

Release	Modification
12.1(3)T	This command was introduced.
12.2(11)T	This command was implemented on the following platforms: Cisco 2600 series, Cisco 3620, Cisco 3640, and Cisco 3660, Cisco AS5350, Cisco AS5400, Cisco AS5850, Cisco AS5300, Cisco AS5800, and Cisco MC3810.
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

Each Real-Time Streaming Protocol (RTSP) session has a TCP port for control and a UDP (RTP) port for delivery of data. The control connection (TCP socket) is used to exchange a set of messages (request from the RTSP client and the response from the server) for displaying a prompt. The **debug rtsp socket** command enables the user to debug the message exchanges being done on the TCP control connection.



### Note

We recommend that you log output from the **debug rtsp socket** command to a buffer rather than sending the output to the console; otherwise, the size of the output could severely impact the performance of the gateway.

## Related Commands

Command	Description
<b>debug rtsp api</b>	Displays debugging output for the RTSP client API.
<b>debug rtsp error</b>	Displays error message for RTSP data.
<b>debug rtsp pmh</b>	Displays debugging messages for the PMH.
<b>voice call debug</b>	Allows configuration of the voice call debugging output.

# debug rudpv1

For debug information for Reliable User Datagram Protocol (RUDP), use the **debug rudpv1** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

```
debug rudpv1 { application | performance | retransmit | segment | signal | state | timer |
transfer }
```

```
no debug rudpv1 { application | performance | retransmit | segment | signal | state | timer |
transfer }
```

## Syntax Description

<b>application</b>	Application debugging.
<b>performance</b>	Performance debugging.
<b>retransmit</b>	Retransmit/soft reset debugging.
<b>segment</b>	Segment debugging.
<b>signal</b>	Signals sent to applications.
<b>state</b>	State transitions.
<b>timer</b>	Timer debugging.
<b>transfer</b>	Transfer state information.

## Defaults

Debugging for rudpv1 is not enabled.

## Command Modes

Privileged EXEC

## Command History

Release	Modification
12.1(1)T	This command was introduced.
12.2(4)T	This command was implemented on the Cisco 2600 series, Cisco 3600 series, and Cisco MC3810.
12.2(2)XB	This command was implemented on the Cisco AS5350 and Cisco AS5400 universal gateways.
12.2(2)XB1	This command was implemented on the Cisco AS5850 platform.
12.2(8)T	This command was implemented on Cisco IAD2420 series integrated access devices (IADs).
12.2(11)T	This command was implemented on the Cisco AS5350, Cisco AS5400, and Cisco AS5850 platforms.
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 this command only during times of low traffic.

**Examples**

The following is sample output from the **debug rudpv1 application** command:

```
Router# debug rudpv1 application

Rudpvl:Turning application debugging on
*Jan 1 00:20:38.271:Send to appl (61F72B6C), seq 12
*Jan 1 00:20:48.271:Send to appl (61F72B6C), seq 13
*Jan 1 00:20:58.271:Send to appl (61F72B6C), seq 14
*Jan 1 00:21:08.271:Send to appl (61F72B6C), seq 15
*Jan 1 00:21:18.271:Send to appl (61F72B6C), seq 16
*Jan 1 00:21:28.271:Send to appl (61F72B6C), seq 17
*Jan 1 00:21:38.271:Send to appl (61F72B6C), seq 18
*Jan 1 00:21:48.275:Send to appl (61F72B6C), seq 19
*Jan 1 00:21:58.275:Send to appl (61F72B6C), seq 20
*Jan 1 00:22:08.275:Send to appl (61F72B6C), seq 21
*Jan 1 00:22:18.275:Send to appl (61F72B6C), seq 22
*Jan 1 00:22:28.275:Send to appl (61F72B6C), seq 23
*Jan 1 00:22:38.275:Send to appl (61F72B6C), seq 24
*Jan 1 00:22:48.279:Send to appl (61F72B6C), seq 25
*Jan 1 00:22:58.279:Send to appl (61F72B6C), seq 26
*Jan 1 00:23:08.279:Send to appl (61F72B6C), seq 27
*Jan 1 00:23:18.279:Send to appl (61F72B6C), seq 28
*Jan 1 00:23:28.279:Send to appl (61F72B6C), seq 29
```

The following is sample output from the **debug rudpv1 performance** command:

```
Router# debug rudpv1 performance

Rudpvl:Turning performance debugging on
corsair-f#
*Jan 1 00:44:27.299:
*Jan 1 00:44:27.299:Rudpvl Sent:Pkts 11, Data Bytes 236, Data Pkts 9
*Jan 1 00:44:27.299:Rudpvl Rcvd:Pkts 10, Data Bytes 237, Data Pkts 9
*Jan 1 00:44:27.299:Rudpvl Discarded:0, Retransmitted 0
*Jan 1 00:44:27.299:
*Jan 1 00:44:37.299:
*Jan 1 00:44:37.299:Rudpvl Sent:Pkts 11, Data Bytes 236, Data Pkts 9
*Jan 1 00:44:37.299:Rudpvl Rcvd:Pkts 10, Data Bytes 237, Data Pkts 9
*Jan 1 00:44:37.299:Rudpvl Discarded:0, Retransmitted 0
*Jan 1 00:44:37.299:
*Jan 1 00:44:47.299:
*Jan 1 00:44:47.299:Rudpvl Sent:Pkts 11, Data Bytes 236, Data Pkts 9
*Jan 1 00:44:47.299:Rudpvl Rcvd:Pkts 11, Data Bytes 236, Data Pkts 9
*Jan 1 00:44:47.299:Rudpvl Discarded:0, Retransmitted 0
*Jan 1 00:44:47.299:
```

The following is sample output from the **debug rudpv1 retransmit** command:

```
Router# debug rudpv1 retransmit

Rudpvl:Turning retransmit/softreset debugging on
*Jan 1 00:52:59.799:Retrans timer, set to ack 199
*Jan 1 00:52:59.903:Retrans timer, set to ack 200
*Jan 1 00:53:00.003:Retrans timer, set to ack 201
*Jan 1 00:53:00.103:Retrans timer, set to ack 202
*Jan 1 00:53:00.203:Retrans timer, set to ack 203
*Jan 1 00:53:00.419:Retrans timer, set to ack 97
*Jan 1 00:53:00.503:Retrans handler fired, 203
*Jan 1 00:53:00.503:Retrans:203:205:
*Jan 1 00:53:00.503:
*Jan 1 00:53:00.607:Retrans timer, set to ack 207
*Jan 1 00:53:00.907:Retrans timer, set to ack 210
*Jan 1 00:53:01.207:Retrans handler fired, 210
*Jan 1 00:53:01.207:Retrans:210:211:212:
```

```

*Jan 1 00:53:01.207:
*Jan 1 00:53:01.207:Retrans timer, set to ack 213
*Jan 1 00:53:01.311:Retrans timer, set to ack 214
*Jan 1 00:53:01.419:Retrans timer, set to ack 98
*Jan 1 00:53:01.611:Retrans timer, set to ack 215
*Jan 1 00:53:01.711:Retrans timer, set to ack 218
*Jan 1 00:53:01.811:Retrans timer, set to ack 219
*Jan 1 00:53:01.911:Retrans timer, set to ack 220
*Jan 1 00:53:02.011:Retrans timer, set to ack 221
*Jan 1 00:53:02.311:Retrans handler fired, 221
*Jan 1 00:53:02.311:Retrans:221:
*Jan 1 00:53:02.311:
*Jan 1 00:53:02.311:Retrans timer, set to ack 222
*Jan 1 00:53:02.415:Retrans timer, set to ack 225

```

The following is sample output from the **debug rudpv1 segment** command:

```
Router# debug rudpv1 segment
```

```

Rudpv1:Turning segment debugging on
*Jan 1 00:41:36.359:Rudpv1: (61F72DAC) Rcvd ACK 61..198 (32)
*Jan 1 00:41:36.359:Rudpv1: (61F72DAC) Send ACK 199..61 (32)
*Jan 1 00:41:36.459:Rudpv1: (61F72DAC) Rcvd ACK 62..199 (8)
*Jan 1 00:41:36.459:Rudpv1: (61F72DAC) Rcvd ACK 62..199 (32)
*Jan 1 00:41:36.459:Rudpv1: (61F72DAC) Send ACK 200..62 (32)
*Jan 1 00:41:36.559:Rudpv1: (61F72DAC) Rcvd ACK 63..200 (32)
*Jan 1 00:41:36.559:Rudpv1: (61F72DAC) Send ACK 201..63 (32)
*Jan 1 00:41:36.659:Rudpv1: (61F72DAC) Rcvd ACK 64..201 (32)
*Jan 1 00:41:36.659:Rudpv1: (61F72DAC) Send ACK 202..64 (32)
*Jan 1 00:41:36.759:Rudpv1: (61F72DAC) Rcvd ACK 65..202 (32)
*Jan 1 00:41:36.759:Rudpv1: (61F72DAC) Send ACK 203..65 (32)
*Jan 1 00:41:36.859:Rudpv1: (61F72DAC) Rcvd ACK 66..202 (32)
*Jan 1 00:41:36.859:Rudpv1: (61F72DAC) Send ACK 204..66 (32)
*Jan 1 00:41:36.959:Rudpv1: (61F72DAC) Rcvd ACK 67..202 (32)
*Jan 1 00:41:36.959:Rudpv1: (61F72DAC) Rcvd ACK EAK 68..202 (9)
*Jan 1 00:41:36.959:Rudpv1: (61F72DAC) Send ACK 203..67 (32)
*Jan 1 00:41:36.963:Rudpv1: (61F72DAC) Send ACK 205..67 (32)
*Jan 1 00:41:36.963:Rudpv1: (61F72DAC) Rcvd ACK 68..204 (8)
*Jan 1 00:41:37.051:Rudpv1: (61F72B6C) Send ACK NUL 118..96 (8)
*Jan 1 00:41:37.051:Rudpv1: (61F72B6C) Rcvd ACK 97..118 (8)
*Jan 1 00:41:37.059:Rudpv1: (61F72DAC) Rcvd ACK 68..205 (32)
*Jan 1 00:41:37.063:Rudpv1: (61F72DAC) Send ACK 206..68 (32)
*Jan 1 00:41:37.263:Rudpv1: (61F72DAC) Rcvd ACK 70..206 (32)
*Jan 1 00:41:37.363:Rudpv1: (61F72DAC) Send ACK EAK 207..68 (9)
*Jan 1 00:41:37.363:Rudpv1: (61F72DAC) Rcvd ACK 71..206 (32)
*Jan 1 00:41:37.363:Rudpv1: (61F72DAC) Rcvd ACK 69..206 (32)
*Jan 1 00:41:37.363:Rudpv1: (61F72DAC) Send ACK 207..71 (8)
*Jan 1 00:41:37.363:Rudpv1: (61F72DAC) Send ACK 207..71 (32)
*Jan 1 00:41:37.363:Rudpv1: (61F72DAC) Send ACK 208..71 (32)
*Jan 1 00:41:37.363:Rudpv1: (61F72DAC) Send ACK 209..71 (32)
*Jan 1 00:41:37.367:Rudpv1: (61F72DAC) Rcvd ACK 72..209 (8)
*Jan 1 00:41:37.463:Rudpv1: (61F72DAC) Rcvd ACK 72..209 (32)
*Jan 1 00:41:37.463:Rudpv1: (61F72DAC) Send ACK 210..72 (32)
*Jan 1 00:41:37.563:Rudpv1: (61F72DAC) Rcvd ACK 73..210 (32)
*Jan 1 00:41:37.563:Rudpv1: (61F72DAC) Send ACK 211..73 (32)

```

The following is sample output from the **debug rudpv1 signal** command:

Router# **debug rudpv1 signal**

```
Rudpv1:Turning signal debugging on
*Jan 1 00:39:59.551:Rudpv1:Sent CONN_FAILED to connID 61F72DAC, sess 33
*Jan 1 00:39:59.551:
*Jan 1 00:39:59.551:Rudpv1:Sent CONN_TRANS_STATE to connID 61F72B6C, sess 34
*Jan 1 00:39:59.551:
*Jan 1 00:39:59.551:Rudpv1:Sent CONN_TRANS_STATE to connID 61F72DAC, sess 33
*Jan 1 00:39:59.551:
*Jan 1 00:39:59.551:Rudpv1:Sent CONN_OPEN to connID 61F72B6C, sess 34

*Jan 1 00:39:59.551:Rudpv1:Sent AUTO_RESET to connID 61F72DAC, sess 33
*Jan 1 00:39:59.551:
*Jan 1 00:40:00.739:%LINK-5-CHANGED:Interface FastEthernet0, changed state
to administratively down
*Jan 1 00:40:01.739:%LINEPROTO-5-UPDOWN:Line protocol on Interface
FastEthernet0, changed state to down
*Jan 1 00:40:04.551:Rudpv1:Sent CONN_RESET to connID 61F72DAC, sess 33
*Jan 1 00:40:04.551:
*Jan 1 00:40:05.051:Rudpv1:Clearing conn rec values, index 2, connid
61F72DAC
*Jan 1 00:40:10.051:Rudpv1:Sent CONN_RESET to connID 61F72DAC, sess 33
*Jan 1 00:40:10.051:
*Jan 1 00:40:10.551:Rudpv1:Clearing conn rec values, index 2, connid
61F72DAC
*Jan 1 00:40:15.551:Rudpv1:Sent CONN_RESET to connID 61F72DAC, sess 33
*Jan 1 00:40:15.551:
*Jan 1 00:40:16.051:Rudpv1:Clearing conn rec values, index 2, connid
61F72DAC

*Jan 1 00:40:21.051:Rudpv1:Sent CONN_RESET to connID 61F72DAC, sess 33
*Jan 1 00:40:21.051:
*Jan 1 00:40:21.551:Rudpv1:Clearing conn rec values, index 2, connid
61F72DAC
*Jan 1 00:40:25.587:%LINK-3-UPDOWN:Interface FastEthernet0, changed state
to up
*Jan 1 00:40:26.551:Rudpv1:Sent CONN_RESET to connID 61F72DAC, sess 33
*Jan 1 00:40:26.551:
*Jan 1 00:40:26.587:%LINEPROTO-5-UPDOWN:Line protocol on Interface
FastEthernet0, changed state to up
*Jan 1 00:40:27.051:Rudpv1:Clearing conn rec values, index 2, connid
61F72DAC
*Jan 1 00:40:28.051:Rudpv1:Sent CONN_OPEN to connID 61F72DAC, sess 33
```

The following is sample output from the **debug rudpv1 state** command:

Router# **debug rudpv1 state**

```
Rudpv1:Turning state debugging on

*Jan 1 00:38:37.323:Rudpv1: (61F72DAC) State Change:OPEN -> CONN_FAILURE
*Jan 1 00:38:37.323:Rudpv1: (61F72B6C) State Change:OPEN -> TRANS_STATE
*Jan 1 00:38:37.323:Rudpv1: (61F72DAC) State Change:CONN_FAILURE ->
TRANS_STATE
*Jan 1 00:38:37.323:Rudpv1: (61F72B6C) State Change:TRANS_STATE -> OPEN
*Jan 1 00:38:37.323:Rudpv1: (61F72DAC) State Change:TRANS_STATE -> SYN_SENT
*Jan 1 00:38:37.455:%LINK-5-CHANGED:Interface FastEthernet0, changed state
to administratively down
*Jan 1 00:38:38.451:%LINEPROTO-5-UPDOWN:Line protocol on Interface
FastEthernet0, changed state to down
*Jan 1 00:38:42.323:Rudpv1: (61F72DAC) State Change:SYN_SENT -> CLOSED
*Jan 1 00:38:42.823:Rudpv1: (61F72DAC) State Change:INACTIVE -> SYN_SENT
```

```
*Jan 1 00:38:47.823:Rudpv1: (61F72DAC) State Change:SYN_SENT -> CLOSED
*Jan 1 00:38:48.323:Rudpv1: (61F72DAC) State Change:INACTIVE -> SYN_SENT
*Jan 1 00:38:53.323:Rudpv1: (61F72DAC) State Change:SYN_SENT -> CLOSED
*Jan 1 00:38:53.823:Rudpv1: (61F72DAC) State Change:INACTIVE -> SYN_SENT
*Jan 1 00:38:56.411:%LINK-3-UPDOWN:Interface FastEthernet0, changed state
to up
*Jan 1 00:38:57.411:%LINEPROTO-5-UPDOWN:Line protocol on Interface
FastEthernet0, changed state to up
*Jan 1 00:38:57.823:Rudpv1: (61F72DAC) State Change:SYN_SENT -> OPEN
```

The following is sample output from the **debug rudpv1 timer** command:

```
Router# debug rudpv1 timer
```

```
Rudpv1:Turning timer debugging on
*Jan 1 00:53:40.647:Starting Retrans timer for connP = 61F72B6C, delay = 300
*Jan 1 00:53:40.647:Stopping SentList timer for connP = 61F72B6C
*Jan 1 00:53:40.747:Starting NullSeg timer for connP = 61F72B6C, delay = 1000
*Jan 1 00:53:40.747:Stopping Retrans timer for connP = 61F72B6C
*Jan 1 00:53:40.747:Starting Retrans timer for connP = 61F72B6C, delay = 300
*Jan 1 00:53:40.747:Stopping SentList timer for connP = 61F72B6C
*Jan 1 00:53:40.847:Starting NullSeg timer for connP = 61F72B6C, delay = 1000
*Jan 1 00:53:40.847:Stopping Retrans timer for connP = 61F72B6C
*Jan 1 00:53:40.847:Starting Retrans timer for connP = 61F72B6C, delay = 300
*Jan 1 00:53:40.847:Stopping SentList timer for connP = 61F72B6C
*Jan 1 00:53:40.947:Starting NullSeg timer for connP = 61F72B6C, delay = 1000
*Jan 1 00:53:40.947:Stopping Retrans timer for connP = 61F72B6C
*Jan 1 00:53:40.947:Starting Retrans timer for connP = 61F72B6C, delay = 300
*Jan 1 00:53:40.947:Stopping SentList timer for connP = 61F72B6C
*Jan 1 00:53:41.047:Starting NullSeg timer for connP = 61F72B6C, delay = 1000
*Jan 1 00:53:41.147:Starting NullSeg timer for connP = 61F72B6C, delay = 1000
*Jan 1 00:53:41.151:Starting NullSeg timer for connP = 61F72B6C, delay = 1000
*Jan 1 00:53:41.151:Starting NullSeg timer for connP = 61F72B6C, delay = 1000
*Jan 1 00:53:41.151:Stopping Retrans timer for connP = 61F72B6C
*Jan 1 00:53:41.151:Starting SentList timer for connP = 61F72B6C, delay = 300
*Jan 1 00:53:41.419:Timer Keepalive (NullSeg) triggered for conn = 61F72DAC
*Jan 1 00:53:41.419:Starting Retrans timer for connP = 61F72DAC, delay = 300
*Jan 1 00:53:41.419:Stopping SentList timer for connP = 61F72DAC
*Jan 1 00:53:41.419:Starting NullSeg timer for connP = 61F72DAC, delay = 1000
*Jan 1 00:53:41.419:Stopping Retrans timer for connP = 61F72DAC
*Jan 1 00:53:41.451:Timer SentList triggered for conn = 61F72B6C
*Jan 1 00:53:41.451:Starting SentList timer for connP = 61F72B6C, delay = 300
*Jan 1 00:53:41.451:Starting NullSeg timer for connP = 61F72B6C, delay = 1000
*Jan 1 00:53:41.451:Stopping SentList timer for connP = 61F72B6C
*Jan 1 00:53:41.551:Starting NullSeg timer for connP = 61F72B6C, delay = 1000
*Jan 1 00:53:41.551:Starting NullSeg timer for connP = 61F72B6C, delay = 1000
*Jan 1 00:53:41.551:Starting NullSeg timer for connP = 61F72B6C, delay = 1000
*Jan 1 00:53:41.551:Starting NullSeg timer for connP = 61F72B6C, delay = 1000
```

The following is sample output from the **debug rudpv1 transfer** command:

```
Router# debug rudpv1 transfer
```

```
Rudpv1:Turning transfer debugging on
*Jan 1 00:37:30.567:Rudpv1:Send TCS, connId 61F72B6C, old connId 61F72DAC
*Jan 1 00:37:30.567:Rudpv1:Initiate transfer state, old conn 61F72DAC to
new conn 61F72B6C
*Jan 1 00:37:30.567:Rudpv1:Old conn send window 51 .. 52
*Jan 1 00:37:30.567:Rudpv1:New conn send window 255 .. 2
*Jan 1 00:37:30.567:Rudpv1:Rcvd TCS 142, next seq 142
*Jan 1 00:37:30.567:Rudpv1:Rcv'ing trans state, old conn 61F72DAC to new
conn 61F72B6C
*Jan 1 00:37:30.567:Rudpv1:Seq adjust factor 148
*Jan 1 00:37:30.567:Rudpv1:New rcvCur 142
```

```
*Jan 1 00:37:30.567:Rudpv1:Send transfer state, old conn 61F72DAC to new
conn 61F72B6C
*Jan 1 00:37:30.567:Rudpv1:Send TCS, connId 61F72B6C, old connId 61F72DAC,
seq adjust 208, indication 0
*Jan 1 00:37:30.567:Rudpv1:Transfer seg 51 to seg 3 on new conn
*Jan 1 00:37:30.567:Rudpv1:Finishing transfer state, old conn 61F72DAC to
new conn 61F72B6C
*Jan 1 00:37:30.567:Rudpv1:Send window 2 .. 4
```

**Related Commands**

<b>Command</b>	<b>Description</b>
<b>clear rudpv1 statistics</b>	Clears RUDP statistics and failure counters.
<b>show rudpv1</b>	Displays RUDP failures, parameters, and statistics.

# debug saa apm



## Note

Effective with Cisco IOS Release 12.3(14)T, the **debug saa apm** command is replaced by the **debug ip sla monitor apm** command. See the **debug ip sla monitor apm** command for more information.

To enable debugging output for Cisco IOS IP Service Level Agreements (SLAs) Application Performance Monitor (APM) operations, use the **debug saa apm** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug saa apm**

**no debug saa apm**

## 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.
12.3(14)T	This command was replaced by the <b>debug ip sla monitor apm</b> command.
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 saa apm** command:

```
Router# debug saa apm
Router# configure terminal
Router(config)# saa apm operation 123 start ftp://apm/config/iptv.cf

21:40:27: SAA-APM-123: downloading file (apm/config/iptv.cf) of size (534)
21:40:29: SAA-APM-123: downloading file (apm/scheduler/master.sch) of size (2500)
21:40:30: SAA-APM-123: downloading file (apm/scripts/iptv.scr) of size (1647)
21:40:32: SAA-APM-123: downloading file (apm/data/iptv.dat) of size (118)
21:40:32: SAA-APM-123: sending APM_CAPABILITIES_REQUEST message
21:40:32: sending control msg:
21:40:32: Ver: 1 ID: 29 Len: 48
21:40:32: SAA-APM-123: apm_engine version: major<1>, minor<0>
21:40:32: SAA-APM-123: sending APM_SCRIPT_DNLD message
21:40:32: sending control msg:
21:40:32: Ver: 1 ID: 30 Len: 148
21:40:37: SAA-APM-123: sending APM_SCRIPT_DNLD_STATUS message
21:40:37: sending control msg:
21:40:37: Ver: 1 ID: 31 Len: 148
21:40:38: SAA-APM-123: starting the operation
21:40:38: SAA-APM-123: sending APM_SCRIPT_START message
21:40:38: sending control msg:
```

```
21:40:38: Ver: 1 ID: 32 Len: 148
21:40:41: SAA-APM: 0,2144,0
.
.
.
21:49:42: SAA-APM-123: waiting for ageout timer to expire
21:55:13: SAA-APM-123: sending APM_SCRIPT_DONE message
21:55:13: sending control msg:
21:55:13: Ver: 1 ID: 42 Len: 148
21:55:13: SAA-APM-123: operation done
```

```
Router(config)# no saa apm
```

```
21:55:13: SAA-APM-123: sending APM_SCRIPT_DONE message
21:55:13: sending control msg:
21:55:13: Ver: 1 ID: 42 Len: 148
21:55:13: SAA-APM-123: operation done
```

# debug saa slm



## Note

Effective with Cisco IOS Release 12.3(14)T, the **debug saa slm** command is replaced by the **debug ip sla monitor slm** command. See the **debug ip sla monitor slm** command for more information.

To enable debugging output of detailed event messages for Cisco IOS IP Service Level Agreements (SLAs) Service Level Monitoring (SLM) Asynchronous Transfer Mode (ATM) operations, use the **debug saa slm** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug saa slm**

**no debug saa slm**

## Syntax Description

This command has no arguments or keywords.

## Command Modes

Privileged EXEC

## Command History

Release	Modification
12.2(11)T	This command was introduced.
12.3(14)T	This command was replaced by the <b>debug ip sla monitor slm</b> command.
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

IP SLAs SLM ATM performance statistics cannot be retrieved from Cisco IOS devices using Simple Network Management Protocol (SNMP). The IP SLAs SLM ATM feature was designed to provide data by responding to extensible markup language (XML) requests.



## Note

This command may generate a large number of debugging messages.

## Examples

In the following example, debugging is enabled for the IP SLAs SLM ATM feature and the IP SLAs XML feature for the purposes of debugging the XML requests and responses:

```
debug saa slm
debug saa xml
```

**Related Commands**

<b>Command</b>	<b>Description</b>
<b>debug saa xml</b>	Enables debugging output of XML requests and responses for IP SLAs operations.

# debug saa xml



## Note

Effective with Cisco IOS Release 12.3(14)T, the **debug saa xml** command is replaced by the **debug ip sla monitor xml** command. See the **debug ip sla monitor xml** command for more information.

To enable debugging output of eXtensible Markup Language (XML) requests and responses for Cisco IOS IP Service Level Agreements (SLAs) operations, use the **debug saa xml** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug saa xml**

**no debug saa xml**

## Syntax Description

This command has no arguments or keywords.

## Command Modes

Privileged EXEC

## Command History

Release	Modification
12.2(11)T	This command was introduced.
12.3(14)T	This command was replaced by the <b>debug ip sla monitor xml</b> command.
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

In the following example, debugging is enabled for the IP SLAs SLM ATM feature and the IP SLAs eXtensible Markup Language (XML) feature for the purposes of debugging the XML requests and responses:

```
debug saa slm
debug saa xml
```

## Related Commands

Command	Description
<b>debug saa slm</b>	Enables debugging output of detailed event messages for IP SLAs SLM ATM operations.

# debug sampler

To enable debugging output for Flexible NetFlow samplers, use the **debug sampler** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

```
debug sampler [[name] sampler-name [detailed] [error] | [name] sampler-name sampling
samples]
```

```
no debug sampler [[name] sampler-name [detailed] [error] | [name] sampler-name sampling
samples]
```

## Syntax Description

<b>name</b> <i>sampler-name</i>	(Optional) The name of a sampler that you previously configured.
<b>sampling</b>	(Optional) Enables debugging for sampling.
<i>samples</i>	(Optional) Number of Samples to debug.
<b>detailed</b>	(Optional) Enables detailed debugging for sampler elements.
<b>error</b>	(Optional) Enables debugging for sampler errors.

## Command Default

Debugging output for Flexible NetFlow samplers is disabled.

## Command Modes

Privileged EXEC

## Command History

Release	Modification
12.4(9)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

You must have already enabled traffic monitoring with Flexible NetFlow using a sampler before you can use the **debug sampler** command.

## Examples

The following output shows that the debug process obtained the ID for the sampler named SAMPLER-1:

```
Router# debug sampler detailed
```

```
*Oct 28 04:14:30.883: Sampler: Sampler(SAMPLER-1: flow monitor NFC-DC-PHOENIX (ip,Et1/0,0)
get ID succeeded:1
```

```
*Oct 28 04:14:30.971: Sampler: Sampler(SAMPLER-1: flow monitor NFC-DC-PHOENIX (ip,Et0/0,I)
get ID succeeded:1
```

## Related Commands

<b>Command</b>	<b>Description</b>
<b>clear sampler</b>	Clears the Flexible NetFlow sampler statistics.
<b>debug sampler</b>	Enables debugging output for Flexible NetFlow samplers.
<b>mode</b>	Configures a packet interval for a Flexible NetFlow sampler.
<b>sampler</b>	Creates a Flexible NetFlow Sampler
<b>show sampler</b>	Displays Flexible NetFlow sampler status and statistics.

# debug satellite

To enable debugging output for the Cisco IP VSAT satellite WAN network module (NM-1VSAT-GILAT), use the **debug satellite** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

```
debug satellite {all | errors | events | hsrp | rbc}
```

```
no debug satellite {all | errors | events | hsrp | rbc}
```

## Syntax Description

<b>all</b>	Displays all types of satellite debug information.
<b>errors</b>	Displays debug information for satellite error events.
<b>events</b>	Displays debug information for software events.
<b>hsrp</b>	Displays debug information for satellite Hot Standby Router Protocol (HSRP) events.
<b>rbc</b>	Displays debug information for satellite Router Blade Control Protocol (RBCP) messages.

## Defaults

No default behavior or values

## Command Modes

Privileged EXEC

## Command History

Release	Modification
12.3(14)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

The **debug satellite errors** command is useful for catching unusual conditions when troubleshooting unexpected behavior. Because this command typically generates very little output, you can enter the **debug satellite errors** command every time you troubleshoot satellite network connectivity.

## Examples

This section provides the following examples:

- [Sample Output for the debug satellite rbc Command, page 2204](#)
- [Sample Output for the debug satellite events Command, page 2204](#)
- [Sample Output for the debug satellite hsrp Command, page 2204](#)
- [Combined Sample Output for the debug satellite hsrp and debug standby Commands, page 2205](#)

### Sample Output for the debug satellite rbcv Command

Every 2 minutes, the NM-1VSAT-GILAT network module sends the router an RBCP message requesting any updates to the routing table. The following example shows how to monitor the route-update messages:

```
Router# debug satellite rbcv
```

```
...
```

The NM-1VSAT-GILAT network module requests IP route information:

```
*May 16 09:18:54.475:Satellitel/0 RBCP Request msg Recd:IPROUTE_REQ(0x22)
```

The Cisco IOS software acknowledges that it received the message from the NM-1VSAT-GILAT network module:

```
*May 16 09:18:54.475:Satellitel/0 RBCP Response msg Sent:IPROUTE_REQ(0x22)
```

The Cisco IOS software sends the IP route information to the NM-1VSAT-GILAT network module:

```
*May 16 09:18:54.475:Satellitel/0 RBCP Request msg Sent:IPROUTE_UPD(0x23)
```

The NM-1VSAT-GILAT network module acknowledges that it received the routing update from the Cisco IOS software:

```
*May 16 09:18:54.475:Satellitel/0 RBCP Response msg Recd:IPROUTE_UPD(0x23)
```

### Sample Output for the debug satellite events Command

The following example shows how to monitor the periodic heartbeats that the NM-1VSAT-GILAT network module sends to the Cisco IOS software:

```
Router# debug satellite events
```

```
satellite major software events debugging is on
.Dec 16 12:57:52.108:Satellitel/0 FSM transition LINK_UP-->LINK_UP, ev=got_heartbeat
.Dec 16 12:58:08.888:Satellitel/0 FSM transition LINK_UP-->LINK_UP, ev=got_heartbeat
.Dec 16 12:58:25.664:Satellitel/0 FSM transition LINK_UP-->LINK_UP, ev=got_heartbeat
.Dec 16 12:58:42.440:Satellitel/0 FSM transition LINK_UP-->LINK_UP, ev=got_heartbeat
```

### Sample Output for the debug satellite hsrp Command

The following example shows the **debug satellite hsrp** command messages that appear when the active router is forced to standby status because the HSRP-tracked satellite interface is shut down:

```
Router# configure terminal
```

```
Enter configuration commands, one per line. End with CNTL/Z.
```

```
Router(config)# interface satellite 1/0
```

```
Router(config-if)# shutdown
```

```
Router(config-if)# end
```

```
Router#
```

```
01:03:48:%SYS-5-CONFIG_I:Configured from console by console
```

```
01:03:49:%LINK-5-CHANGED:Interface Satellitel/0, changed state to administratively down
```

```
01:03:50:%LINEPROTO-5-UPDOWN:Line protocol on Interface Satellitel/0, changed state to down
```

```
01:04:22:%HSRP-6-STATECHANGE:FastEthernet0/0 Grp 1 state Active -> Speak
```

```
01:04:22:HSRP-sat:IPred group grp-x update state ACTIVE --> SPEAK
```

```
01:04:22:Satellitel/0 HSRP-sat:fsm crank ACTIVE-->STANDBY
```

```
01:04:22:Satellitel/0 HSRP-sat:send standby msg STANDBY
```

```
01:04:32:HSRP-sat:IPred group grp-x update state SPEAK --> STANDBY
```

```
01:04:32:Satellitel/0 HSRP-sat:fsm crank STANDBY-->STANDBY
```

```
01:04:32:Satellitel/0 HSRP-sat:send standby msg STANDBY
```

```
01:04:42:Satellitel/0 HSRP-sat:send standby msg STANDBY
```

```
01:04:52:Satellitel/0 HSRP-sat:standby msg STANDBY deferred, not in operational state
```

```
01:05:02:Satellitel/0 HSRP-sat:standby msg STANDBY deferred, not in operational state
```

```
01:05:12:Satellitel/0 HSRP-sat:standby msg STANDBY deferred, not in operational state
```

```
01:05:22:Satellite1/0 HSRP-sat:standby msg STANDBY deferred, not in operational state
01:05:32:Satellite1/0 HSRP-sat:standby msg STANDBY not sent, already in state
01:06:47:%VSAT-5-STANDBY_MODE:Satellite1/0 module configured for standby mode
01:09:32:Satellite1/0 HSRP-sat:fsm crank STANDBY-->STANDBY-UP
```

### Combined Sample Output for the debug satellite hsrp and debug standby Commands

The following example shows HSRP-related debug output for both the router and the NM-1VSAT-GILAT network module when the router goes from standby to active state because the HSRP-tracked satellite interface is reenabled:

```
Router# show debugging

SATCOM:
  satellite HSRP events debugging is on

HSRP:
  HSRP Errors debugging is on
  HSRP Events debugging is on
  HSRP Packets debugging is on
```

The satellite interface is reenabled:

```
Router# configure terminal
Router(config)# interface satellite 1/0
Router(config-if)# no shutdown
Router(config-if)# end
Router#
```

The effective HSRP priority of the router changes as the tracked satellite interface comes up:

```
02:14:37:HSRP:Fa0/0 Grp 1 Hello in 10.123.96.2 Active pri 90 vIP 10.123.96.100
02:14:39:HSRP:Fa0/0 API 10.1.0.6 is not an HSRP address
02:14:39:HSRP:Fa0/0 Grp 1 Hello out 10.123.96.3 Standby pri 90 vIP 10.123.96.100
02:14:39:HSRP:Fa0/0 Grp 1 Track 1 object changed, state Down -> Up
02:14:39:HSRP:Fa0/0 Grp 1 Priority 90 -> 100
Router#
```

The router changes from standby to active state because its priority is now highest in the hot standby group, and preemption is enabled:

```
02:14:40:HSRP:Fa0/0 Grp 1 Hello in 10.123.96.2 Active pri 90 vIP 10.123.96.100
02:14:40:HSRP:Fa0/0 Grp 1 Standby:h/Hello rcvd from lower pri Active router
(90/10.123.96.2)
02:14:40:HSRP:Fa0/0 Grp 1 Active router is local, was 10.123.96.2
02:14:40:HSRP:Fa0/0 Grp 1 Standby router is unknown, was local
02:14:40:HSRP:Fa0/0 Redirect adv out, Active, active 1 passive 3
02:14:40:HSRP:Fa0/0 Grp 1 Coup out 10.123.96.3 Standby pri 100 vIP 10.123.96.100
02:14:40:HSRP:Fa0/0 Grp 1 Standby -> Active
02:14:40:%HSRP-6-STATECHANGE:FastEthernet0/0 Grp 1 state Standby -> Active
```

The HSRP status of the satellite interface also changes from standby to active state because the **service-module ip redundancy** command was previously entered to link the HSRP status of the satellite interface to the primary HSRP interface, Fast Ethernet 0/0.

```
02:14:40:HSRP:Fa0/0 Grp 1 Redundancy "grp-x" state Standby -> Active
02:14:40:HSRP-sat:IPred group grp-x update state STANDBY --> ACTIVE
02:14:40:Satellite1/0 HSRP-sat:fsm crank STANDBY-UP-->ACTIVE-COND
02:14:40:HSRP:Fa0/0 Redirect adv out, Active, active 1 passive 2
02:14:40:HSRP:Fa0/0 Grp 1 Hello out 10.123.96.3 Active pri 100 vIP 10.123.96.100
02:14:40:HSRP:Fa0/0 REDIRECT adv in, Passive, active 0, passive 2, from 10.123.96.2
02:14:40:HSRP:Fa0/0 REDIRECT adv in, Passive, active 0, passive 1, from 10.123.96.15
02:14:40:HSRP:Fa0/0 Grp 1 Hello in 10.123.96.2 Speak pri 90 vIP 10.123.96.100
```

Line protocols come up, and HSRP states become fully active:

```
02:14:41:%LINK-3-UPDOWN:Interface Satellitel/0, changed state to up
02:14:42:%LINEPROTO-5-UPDOWN:Line protocol on Interface Satellitel/0, changed state to up

02:14:43:HSRP:Fa0/0 Grp 1 Hello out 10.123.96.3 Active pri 100 vIP 10.123.96.100
02:14:43:HSRP:Fa0/0 Grp 1 Redundancy group grp-x state Active -> Active
02:14:43:HSRP-sat:IPred group grp-x update state ACTIVE --> ACTIVE
02:14:43:Satellitel/0 HSRP-sat:fsm crank ACTIVE-COND-->ACTIVE-COND
02:14:43:HSRP:Fa0/0 Grp 1 Hello in 10.123.96.2 Speak pri 90 vIP 10.123.96.100
02:14:46:HSRP:Fa0/0 Grp 1 Hello out 10.123.96.3 Active pri 100 vIP 10.123.96.100
02:14:46:HSRP:Fa0/0 Grp 1 Redundancy group grp-x state Active -> Active
02:14:46:HSRP-sat:IPred group grp-x update state ACTIVE --> ACTIVE
02:14:46:Satellitel/0 HSRP-sat:fsm crank ACTIVE-COND-->ACTIVE-COND
02:14:46:HSRP:Fa0/0 Grp 1 Hello in 10.123.96.2 Speak pri 90 vIP 10.123.96.100
02:14:49:HSRP:Fa0/0 Grp 1 Hello out 10.123.96.3 Active pri 100 vIP 10.123.96.100
02:14:49:HSRP:Fa0/0 Grp 1 Hello in 10.123.96.2 Speak pri 90 vIP 10.123.96.100
02:14:50:HSRP:Fa0/0 Grp 1 Hello in 10.123.96.2 Standby pri 90 vIP 10.123.96.100
02:14:50:HSRP:Fa0/0 Grp 1 Standby router is 10.123.96.2
02:14:51:Satellitel/0 HSRP-sat:send standby msg ACTIVE
02:14:52:HSRP:Fa0/0 Grp 1 Hello out 10.123.96.3 Active pri 100 vIP 10.123.96.100
02:14:53:HSRP:Fa0/0 Grp 1 Hello in 10.123.96.2 Standby pri 90 vIP 10.123.96.100
02:14:55:HSRP:Fa0/0 Grp 1 Hello out 10.123.96.3 Active pri 100 vIP 10.123.96.100
```

#### Related Commands

Command	Description
<b>debug satellite firmware</b>	Enables debugging output for the Cisco IP VSAT satellite WAN network module (NM-1VSAT-GILAT) firmware.
<b>debug standby</b>	Displays all HSRP errors, events, and packets.

# debug satellite firmware

To enable debugging output for the Cisco IP VSAT satellite WAN network module (NM-1VSAT-GILAT) firmware, use the **debug satellite firmware** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

```
debug satellite firmware {all | level number | option}
```

```
no debug satellite firmware
```

## Syntax Description

<b>all</b>	Displays all satellite firmware events.
<b>level number</b>	Satellite debug level. The debug level affects what information is displayed for subsequently entered <b>debug satellite firmware</b> commands. See <a href="#">Table 303</a> .
<b>option</b>	One of the following options. See <a href="#">Table 303</a> . <ul style="list-style-type: none"> <li>• <b>bb</b>—Satellite backbone events</li> <li>• <b>buf</b>—Satellite buffer events</li> <li>• <b>en</b>—Satellite firmware encryption events</li> <li>• <b>ip</b>—Satellite IP events</li> <li>• <b>rbc</b>—Satellite RBCP events</li> <li>• <b>rpa</b>—Satellite Remote Page Acceleration (RPA) events</li> <li>• <b>sat</b>—Satellite inbound and outbound packet statistics</li> <li>• <b>tcp</b>—Satellite TCP events</li> <li>• <b>trc</b>—Satellite backbone traces</li> </ul>

## Defaults

No default behavior or values.

## Command Modes

Privileged EXEC

## Command History

Release	Modification
12.3(14)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

The output from this command is generally useful for diagnostic tasks performed by technical support. The level number affects which debug messages the system displays for subsequently entered **debug satellite firmware** commands. [Table 303](#) describes what each command option displays at each debug level.

**Note**

Level 3 debugging produces significant amounts of output that may negatively impact the performance of both the NM-1 VSAT-GILAT network module and the router. When you enter debug level 3, a warning message and confirmation prompt appear.

**Table 303** *debug satellite firmware Command Level Options*

Option	Level 1 Output	Level 2 Output	Level 3 Output
<b>bb</b>	Backbone link information	Frame statistics for the backbone link to the hub	—
<b>buf</b>	Buffer information	Buffer owners	—
<b>en</b>	Satellite firmware-based encryption events	—	—
<b>ip</b>	IP statistics	—	Driver transmission statistics
<b>rbc</b>	Number of transmitted and received RBCP messages	—	Satellite Control Protocol (SCP) message summaries
<b>rpa</b>	RPA statistics	Tunnel connect and disconnect events	—
<b>tcp</b>	TCP statistics	TCP connection information	TCP statistics and TCP connection information
<b>sat</b>	Inbound and outbound packet statistics	Inbound and outbound packet statistics	Inbound and outbound packet statistics
<b>trc</b>	—	—	Backbone receive and transmit traces

**Examples**

This section provides the following sample output for the **debug satellite firmware** command:

- [Sample Output for the debug satellite firmware all Command, page 2209](#)
- [Sample Output for the bb Option at Level 1, page 2209](#)
- [Sample Output for the bb Option at Level 2, page 2210](#)
- [Sample Output for the buf Option at Level 1, page 2210](#)
- [Sample Output for the buf Option at Level 2, page 2210](#)
- [Sample Output for the ip Option at Level 1, page 2211](#)
- [Sample Output for the rbc Option at Level 1, page 2211](#)
- [Sample Output for the rpa Option at Level 1, page 2211](#)
- [Sample Output for the rpa Option at Level 2, page 2211](#)
- [Sample Output for the sat Option at All Levels, page 2212](#)
- [Sample Output for the tcp Option at Level 1, page 2212](#)
- [Sample Output for the tcp Option at Level 2, page 2212](#)
- [Sample Output for the tcp Option at Level 3, page 2212](#)
- [Sample Output for the trc Option at Level 3, page 2213](#)

**Sample Output for the debug satellite firmware all Command**

The following example shows all satellite firmware events and statistics:

```
Router# debug satellite firmware all

2d06h: Satellite2/0
buffers 4856 min 4486 list_str 683798 list_end 6885c8
emp 686030 fil 685de0 start 6885c8 end fb4fe8

2d06h: Satellite2/0
TCP stats: NetRXBytes=223 NetTXBytes=4775126 NetRxPkts=104213 ToIOSPkts=104166

2d06h: Satellite2/0
SAT stats: OUTbound_pkts=114131, INbound_pkts=182347

2d06h: Satellite2/0
RBCP statistics: TXcount=975 RXCount=975

2d06h: Satellite2/0
RPA stats: ToTunnel=0 FromTunnel=0
TunnelGets=0 TunnelNotGets=0
BlksUsed=0 BlksIn-Use=0 Max=300

2d06h: Satellite2/0
EN:
RX encrypted bytes received = 0
RX: compressed=0 -> Uncompressed=0
TX: compressed=0 -> Uncompressed=0

2d06h: Satellite2/0
BB 6 LINK state=INFO_STATE
  Status = 0x79, LOW NOT READY, HI PRI READY
  RSP Q free=230, Max HI=228, Max LOW=224, Max DG=232
  IN RA mode
  Curr DG BW=50000, HighDG BW=100000, Curr BW=98094
  MaxDG BW=1250000, Max BW=2500000
  PD Queue lengths:
    q_wtos=0, q_wtos=57, q_wtos_high=0, q_defrag=d
  DG Queue lengths:
    q_dg_wtos=0, q_dg_wtos_hi=0, q_dg_defrag=0
  Congestion Levels: TX LOCAL = 7, TX NET = 0

2d06h: Satellite2/0
IP stats: ToIOS_Pkts=234193, ToIOS_Bytes=183444492 FromIOS_Pkts=143 From_IOS_Bytes=12204

2d06h: Satellite2/0 NO Trace at levels 1 or 2

2d06h: Satellite2/0 NO Trace at levels 1 or 2
```

**Sample Output for the bb Option at Level 1**

The following example shows backbone link information:

```
Router# debug satellite firmware level 1
Router# debug satellite firmware bb

satellite BackBone events debugging is on
Router#
2d06h: Satellite2/0
BB 6 LINK state=INFO_STATE
  Status = 0x79, LOW NOT READY, HI PRI READY
  RSP Q free=240, Max HI=228, Max LOW=224, Max DG=232
  IN RA mode
  Curr DG BW=50000, HighDG BW=100000, Curr BW=96188
```

```

MaxDG BW=1250000, Max BW=2500000
PD Queue lengths:
  q_wtog=0, q_wtos=95, q_wtos_high=0, q_defrag=d
DG Queue lengths:
  q_dg_wtos=0, q_dg_wtos_hi=0, q_dg_defrag=0
Congestion Levels:      TX LOCAL = 7, TX NET = 0

2d06h: Satellite2/0
BB 6 LINK state=INFO_STATE
  Status = 0x7b, LOW READY, HI PRI READY
  RSP Q free=27, Max HI=228, Max LOW=224, Max DG=232
  IN RA mode
  Curr DG BW=50000, HighDG BW=100000, Curr BW=92376
  MaxDG BW=1250000, Max BW=2500000
  PD Queue lengths:
    q_wtog=0, q_wtos=24, q_wtos_high=0, q_defrag=d
  DG Queue lengths:
    q_dg_wtos=0, q_dg_wtos_hi=0, q_dg_defrag=0
  Congestion Levels:      TX LOCAL = 4, TX NET = 0

```

### Sample Output for the bb Option at Level 2

The following example shows frame statistics for the backbone link to the hub:

```

Router# debug satellite firmware level 2
Router# debug satellite firmware bb

satellite BackBone events debugging is on
Router#
2d06h: Satellite2/0 BB link statistics

```

Frame Type	# Received	# Transmitted
INFORMATION	00096238	00184811
UNNUMBERED	00000000	00000067
RETRANSMITTED	00000000	00000000
POLLS	00000000	00000000
ACKS	00006640	00000455
NAKS	00000000	00000000
PACKS	00000000	00000000
UA	00000001	00000000
SABME	00000000	00000001
DISC	00000000	00000000

### Sample Output for the buf Option at Level 1

The following example shows buffer information:

```

Router# debug satellite firmware level 1
Router# debug satellite firmware buf

*May 13 15:58:54.498:Satellit1/0
buffers 4951 min 4945 list_str 681858 list_end 686688
emp 683abc fil 6839e8 start 686688 end fb30a8

```

### Sample Output for the buf Option at Level 2

The following example shows buffer owners:

```

Router# debug satellite firmware level 2
Router# debug satellite firmware buf

*May 13 15:59:13.438:Satellit1/0 inuse 49 free 4951
Trace byte 1
Trace byte = 0x169 Count = 49
Trace byte 2

```

```
Trace byte = 0x 0    Count =    49
  0 buffers with BB Rel only
  0 buffers with in lower layer set
  0 buffers with do not transmit set
  0 buffers on BB retransmit queues
```

### Sample Output for the ip Option at Level 1

The following example shows IP statistics:

```
Router# debug satellite firmware level 1
Router# debug satellite firmware ip

*Nov  7 08:27:56.440: Satellite3/0
IP stats: ToIOS_Pkts=0, ToIOS_Bytes=0 FromIOS_Pkts=84751 From_IOS_Bytes=5941124
```

### Sample Output for the rbcP Option at Level 1

The following example shows the number of RBCP messages transmitted and received since the most recent reset of the Cisco IOS software on the router or the VSAT software on the NM-1VSAT-GILAT network module:

```
Router# debug satellite firmware level 1
Router# debug satellite firmware rbcP

RBCP statistics:TXcount=301154 RXCount=301155
```

### Sample Output for the rpa Option at Level 1

The following example shows RPA statistics:

```
Router# debug satellite firmware level 1
Router# debug satellite firmware rpa

*Nov  7 08:27:13.488:Satellite3/0
RPA stats:ToTunnel=0 FromTunnel=0
TunnelGets=0 TunnelNotGets=0
BlksUsed=0 BlksIn-Use=0 Max=400
```

### Sample Output for the rpa Option at Level 2

The following example shows a tunnel being disconnected:

```
Router# debug satellite firmware level 2
Router# debug satellite firmware rpa

*May 13 18:27:59.779:Satellit1/0 RPA Tunnel DOWN
RPA:InitTunnelConn Successful locIP e000006 locPort 1090, RemIP c0a80186,
RemPort 9876
RPA Tunnel DOWN
RPA:InitTunnelConn Successful locIP e000006 locPort 1091, RemIP c0a80186,
RemPort 9876
RPA Tunnel DOWN
RPA:InitTunnelConn Successful locIP e000006 locPort 1092, RemIP c0a80186,
RemPort 9876
RPA Tunnel DOWN
RPA:InitTunnelConn Successful locIP e000006 locPort 1093, RemIP c0a80186,
RemPort 9876
RPA Tunnel DOWN
RPA:InitTunnelConn Successful locIP e000006 locPort 1094, RemIP c0a80186,
RemPort 9876
```

**Sample Output for the sat Option at All Levels**

The following example shows inbound and outbound packet statistics. Note that for all levels, the debug output is the same for the **sat** option.

```
Router# debug satellite firmware level 1
Router# debug satellite firmware sat

satellite related trace events debugging is on
Router#
1d16h: Satellite2/0
SAT stats: OUTbound_pkts=25660796, INbound_pkts=3235932

1d16h: Satellite2/0
SAT stats: OUTbound_pkts=25660800, INbound_pkts=3235934

1d16h: Satellite2/0
SAT stats: OUTbound_pkts=25660803, INbound_pkts=3235934

1d16h: Satellite2/0
SAT stats: OUTbound_pkts=25660803, INbound_pkts=3235934
```

**Sample Output for the tcp Option at Level 1**

The following example shows TCP statistics:

```
Router# debug satellite firmware level 1
Router# debug satellite firmware tcp

satellite tcp events debugging is on
Router#
2d06h: Satellite2/0
TCP stats: NetRXBytes=631292 NetTXBytes=4009436 NetRxPkts=49244 ToIOSPkts=49246

2d06h: Satellite2/0
TCP stats: NetRXBytes=1154356 NetTXBytes=4086106 NetRxPkts=49621 ToIOSPkts=49629
```

**Sample Output for the tcp Option at Level 2**

The following example shows the TCP connections:

```
Router# debug satellite firmware level 2
Router# debug satellite firmware tcp

satellite tcp events debugging is on
Router#
2d06h: Satellite2/0 TCP connections:
ID=48, locIP=192.168.107.2 remIP=172.25.1.2, locP=2962, remP=21 state=17 iosQ=0
ID=49, locIP=192.168.107.2 remIP=172.25.1.2, locP=2963, remP=20 state=17 iosQ=0
ID=58, locIP=192.168.107.2 remIP=172.25.1.28, locP=2972, remP=21 state=17 iosQ=0
ID=59, locIP=192.168.107.2 remIP=172.25.1.28, locP=2973, remP=20 state=17 iosQ=7

2d06h: Satellite2/0 TCP connections:
ID=48, locIP=192.168.107.2 remIP=172.25.1.2, locP=2962, remP=21 state=17 iosQ=0
ID=49, locIP=192.168.107.2 remIP=172.25.1.2, locP=2963, remP=20 state=7 iosQ=0
ID=60, locIP=192.168.107.2 remIP=172.25.1.28, locP=2974, remP=21 state=3 iosQ=0
```

**Sample Output for the tcp Option at Level 3**

The following example shows TCP statistics and connections:

```
Router# debug satellite firmware level 3
Output may be extensive and affect performance. Continue? [yes]: yes
Router# debug satellite firmware tcp

satellite tcp events debugging is on
```

```

Router#
2d06h: Satellite2/0
TCP stats: NetRXBytes=279 NetTXBytes=9436111 NetRxPkts=64991 ToIOSPkts=64999

2d06h: Satellite2/0 TCP connections:
ID=48, locIP=192.168.107.2 remIP=172.25.1.2, locP=2962, remP=21 state=7 iosQ=0
ID=49, locIP=192.168.107.2 remIP=172.25.1.2, locP=2963, remP=20 state=7 iosQ=0
ID=62, locIP=192.168.107.2 remIP=172.25.1.28, locP=2976, remP=21 state=7 iosQ=0

2d06h: Satellite2/0
TCP stats: NetRXBytes=382 NetTXBytes=9582924 NetRxPkts=64993 ToIOSPkts=65001

2d06h: Satellite2/0 TCP connections:
ID=48, locIP=192.168.107.2 remIP=172.25.1.2, locP=2962, remP=21 state=17 iosQ=0
ID=49, locIP=192.168.107.2 remIP=172.25.1.2, locP=2963, remP=20 state=17 iosQ=0
ID=62, locIP=192.168.107.2 remIP=172.25.1.28, locP=2976, remP=21 state=7 iosQ=0

```

### Sample Output for the trc Option at Level 3

The following example shows detailed receive and transmit traces for the backbone link:

```

Router# debug satellite firmware level 3
Output may be extensive and affect performance. Continue? [yes]: yes
Router# debug satellite firmware trc

satellite BackBone trace debugging is on
Router#
2d06h: Satellite2/0 strrec 0, rec 0, count 256, trc 1a6dd78, str 1a5c600, end 1a
74600
count 4096, emp 1a6dd78, fil 1a6d8b0, lnknum=6
  0 xmt  6 len  951  9 pd    con 0 PF  3 ns  169 nr   15 a c12 0   0.000
  1 xmt  6 len  951  9 pd    con 0 PF  3 ns  170 nr   15 a c12 0   0.010
  2 xmt  6 len  951  9 pd    con 0 PF  3 ns  171 nr   15 a c12 0   0.010
  3 xmt  6 len  951  9 pd    con 0 PF  3 ns  172 nr   15 a c12 0   0.010
  4 xmt  6 len  951  9 pd    con 0 PF  3 ns  173 nr   15 a c12 0   0.030
  5 xmt  6 len
2d06h: Satellite2/0 951
2d06h: Satellite2/0 9 pd    con 0 PF  3 ns  174 nr   15 a c12 0   0.010
  6 xmt  6 len  951  9 pd    con 0 PF  3 ns  175 nr   15 a c12 0   0.010
  7 xmt  6 len  951  9 pd    con 0 PF  3 ns  176 nr   15 a c12 0   0.010
  8 xmt  6 len  951  9 pd    con 0 PF  3 ns  177 nr   15 a c12 0   0.010
  9 xmt  6 len  951  9 pd    con 0 PF  3 ns  178 nr   15 a c12 0   0.010
 10 xmt  6 len  951  9 pd    con 0 PF  3 ns  179 nr   15 a c12 0   0.010
 11 xmt  6 len  951  9 pd    con 0 PF  3 ns  180 nr   15 a c12 0   0.010

```

### Related Commands

Command	Description
<b>debug satellite</b>	Enables debugging output for the Cisco IP VSAT satellite WAN network module (NM-1VSAT-GILAT).

# debug sccp

To display debugging information for Simple Client Control Protocol (SCCP) and its related applications (transcoding and conferencing), use the **debug sccp** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

```
debug sccp {all | errors | events | packets | parser}
```

```
no debug sccp
```

## Syntax Description

<b>all</b>	All SCCP debug-trace information.
<b>errors</b>	SCCP errors.
<b>events</b>	SCCP events.
<b>packets</b>	SCCP packets.
<b>parser</b>	SCCP parser and builder.

## Command Modes

Privileged EXEC

## Command History

Release	Modification
12.1(5)YH	This command was introduced on the Cisco VG200.
12.2(13)T	This command was implemented on the Cisco 2600 series, Cisco 3620, Cisco 3640, Cisco 3660, and Cisco 3700 series.
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

The router on which this command is used must be equipped with one or more digital T1/E1 packet voice trunk network modules (NM-HDVs) or high-density voice (HDV) transcoding and conferencing digital signal processor (DSP) farms (NM-HDV-FARMS) to provide DSP resources.

Debugging is turned on for all DSP farm service sessions. You can debug multiple sessions simultaneously, with different levels of debugging for each.

## Examples

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

```
Router# debug sccp events
```

```
Skinny Client Control Protocol events debugging is on
```

```
*Mar 1 00:46:29: sccp_create_application: send keepalive msg, appl 6248F760, appl_type 1,
count 0
*Mar 1 00:46:29: sccp_keepalive: send keepalive id 0, len 4
*Mar 1 00:46:29: sccp_process_mtp_pdu: appl - 6248F760, mbuf - 6248F7D4
*Mar 1 00:46:29: sccp_process_mtp_pdu: msg_ptr 6248F7DC, len 4, offset 12, msg_id 256
*Mar 1 00:46:30: sccp_create_application: send keepalive msg, appl 6248FC10, appl_type 2,
count 0
```

```
*Mar 1 00:46:30: sccp_keepalive: send keepalive id 0, len 4
*Mar 1 00:46:30: sccp_process_mtp_pdu: appl - 6248FC10, mbuf - 6248FC84
*Mar 1 00:46:30: sccp_process_mtp_pdu: msg_ptr 6248FC8C, len 4, offset 12, msg_id 256
*Mar 1 00:46:37: sccp_create_application: send keepalive msg, appl 6248F760, appl_type 1,
count 0
*Mar 1 00:46:37: sccp_keepalive: send keepalive id 0, len 4
*Mar 1 00:46:37: sccp_process_mtp_pdu: appl - 6248F760, mbuf - 6248F7D4
*Mar 1 00:46:37: sccp_process_mtp_pdu: msg_ptr 6248F7DC, len 4, offset 12, msg_id 256
*Mar 1 00:46:37: sccp_create_application: send keepalive msg, appl 6248FC10, appl_type 2,
count 0
*Mar 1 00:46:37: sccp_keepalive: send keepalive id 0, len 4
*Mar 1 00:46:38: sccp_process_mtp_pdu: appl - 6248FC10, mbuf - 6248FC84
*Mar 1 00:46:38: sccp_process_mtp_pdu: msg_ptr 6248FC8C, len 4, offset 12, msg_id 256
*Mar 1 00:46:43: sccp_process_mtp_pdu: appl - 6248FC10, mbuf - 6248FC84
*Mar 1 00:46:43: sccp_process_mtp_pdu: msg_ptr 6248FC8C, len 28, offset 36, msg_id 261
*Mar 1 00:46:43: xapp_open_receive_chnl: SCCP orc_msg - 6248FC8C, appl - 6248FC10
*Mar 1 00:46:43: xapp_search_for_chnl_rec: sess_id 27, conn_id 2769
*Mar 1 00:46:43: xapp_add_chnl_rec: chnl 631142BC
*Mar 1 00:46:43: xapp_add_sess_rec: Add sess_rec (63114360) record
*Mar 1 00:46:43: xapp_open_receive_chnl: stat 0, eve 0, sid 27, cid 2769, codec 1,
pkt-period 20
*Mar 1 00:46:43: xapp_open_chnl_request: chnl_rec 631142BC
*Mar 1 00:46:43: xapp_open_chnl_request: chnl_rec 631142BC, sess_id 27, conn_id 2769,
cstate 0, nstate 1
*Mar 1 00:46:43: xapp_dequeue_and_process_dspf_events: chnl_rec 631142BC, state 1, eve_id
1
*Mar 1 00:46:43: xapp_open_chnl_success: chnl_rec 631142BC
*Mar 1 00:46:43: xapp_open_chnl_success: chnl_rec 631142BC, sess_id 27, conn_id 2769,
cstate 1, nstate 2, lc_ipaddr 10.10.1.1, lport 21066
*Mar 1 00:46:43: sccp_process_mtp_pdu: appl - 6248FC10, mbuf - 6248FC84
*Mar 1 00:46:43: sccp_process_mtp_pdu: msg_ptr 6248FC8C, len 28, offset 36, msg_id 261
*Mar 1 00:46:43: xapp_open_receive_chnl: SCCP orc_msg - 6248FC8C, appl - 6248FC10
*Mar 1 00:46:43: xapp_search_for_chnl_rec: sess_id 27, conn_id 2785
*Mar 1 00:46:43: xapp_add_chnl_rec: chnl 631142E4
*Mar 1 00:46:43: xapp_open_receive_chnl: stat 0, eve 0, sid 27, cid 2785, codec 1,
pkt-period 20
*Mar 1 00:46:43: xapp_open_chnl_request: chnl_rec 631142E4
*Mar 1 00:46:43: xapp_open_chnl_request: chnl_rec 631142E4, sess_id 27, conn_id 2785,
cstate 0, nstate 1
*Mar 1 00:46:43: xapp_dequeue_and_process_dspf_events: chnl_rec 631142E4, state 1, eve_id
1
*Mar 1 00:46:43: xapp_open_chnl_success: chnl_rec 631142E4
*Mar 1 00:46:43: xapp_open_chnl_success: chnl_rec 631142E4, sess_id 27, conn_id 2785,
cstate 1, nstate 2, lc_ipaddr 10.10.1.1, lport 25706
*Mar 1 00:46:43: sccp_process_mtp_pdu: appl - 6248FC10, mbuf - 6248FC84
*Mar 1 00:46:43: sccp_process_mtp_pdu: msg_ptr 6248FC8C, len 44, offset 52, msg_id 138
*Mar 1 00:46:43: xapp_start_media_transmission: SCCP stmt_msg - 6248FC8C, appl - 6248FC10
*Mar 1 00:46:43: xapp_search_for_chnl_rec: sess_id 27, conn_id 2769
*Mar 1 00:46:43: xapp_start_media_transmission: chnl_rec 631142BC, stat 2, sid 27, cid
2769, ripaddr 10.10.1.5, rport 32148, codec 1, pkt-period 20, pre 11, silen 16777500, mfp
1
*Mar 1 00:46:43: xapp_modify_chnl_request: chnl_rec 631142BC
*Mar 1 00:46:43: xapp_modify_chnl_request: chnl_rec 631142BC, sess_id 27, conn_id 2769,
cstate 2, nstate 2
*Mar 1 00:46:43: xapp_dequeue_and_process_dspf_events: chnl_rec 631142BC, state 2, eve_id
4
*Mar 1 00:46:43: xapp_modify_chnl_success: chnl_rec 631142BC, sess_id 27, conn_id 2769,
cstate 2
*Mar 1 00:46:43: sccp_process_mtp_pdu: appl - 6248FC10, mbuf - 6248FC84
*Mar 1 00:46:43: sccp_process_mtp_pdu: msg_ptr 6248FC8C, len 44, offset 52, msg_id 138
*Mar 1 00:46:43: xapp_start_media_transmission: SCCP stmt_msg - 6248FC8C, appl - 6248FC10
*Mar 1 00:46:43: xapp_search_for_chnl_rec: sess_id 27, conn_id 2785
```

```

*Mar 1 00:46:43: xapp_start_media_transmission: chnl_rec 631142E4, stat 2, sid 27, cid
2785, ripaddr 10.10.1.7, rport 16422, codec 1, pkt-period 20, pre 11, silen 16777501, mfpp
1
*Mar 1 00:46:43: xapp_modify_chnl_request: chnl_rec 631142E4
*Mar 1 00:46:43: xapp_modify_chnl_request: chnl_rec 631142E4, sess_id 27, conn_id 2785,
cstate 2, nstate 2
*Mar 1 00:46:43: xapp_dequeue_and_process_dspf_events: chnl_rec 631142E4, state 2, eve_id
4
*Mar 1 00:46:43: xapp_modify_chnl_success: chnl_rec 631142E4, sess_id 27, conn_id 2785,
cstate 2
*Mar 1 00:46:44: sccp_create_application: send keepalive msg, appl 6248F760, appl_type 1,
count 0
*Mar 1 00:46:44: sccp_keepalive: send keepalive id 0, len 4
*Mar 1 00:46:45: sccp_process_mtp_pdu: appl - 6248F760, mbuf - 6248F7D4
*Mar 1 00:46:45: sccp_process_mtp_pdu: msg_ptr 6248F7DC, len 4, offset 12, msg_id 256
*Mar 1 00:46:45: sccp_create_application: send keepalive msg, appl 6248FC10, appl_type 2,
count 0
*Mar 1 00:46:45: sccp_keepalive: send keepalive id 0, len 4
*Mar 1 00:46:46: sccp_process_mtp_pdu: appl - 6248FC10, mbuf - 6248FC84
*Mar 1 00:46:46: sccp_process_mtp_pdu: msg_ptr 6248FC8C, len 4, offset 12, msg_id 256
*Mar 1 00:46:47: sccp_process_mtp_pdu: appl - 6248FC10, mbuf - 6248FC84
*Mar 1 00:46:47: sccp_process_mtp_pdu: msg_ptr 6248FC8C, len 28, offset 36, msg_id 261
*Mar 1 00:46:47: xapp_open_receive_chnl: SCCP orc_msg - 6248FC8C, appl - 6248FC10
*Mar 1 00:46:47: xapp_search_for_chnl_rec: sess_id 27, conn_id 2817
*Mar 1 00:46:47: xapp_add_chnl_rec: chnl 6311430C
*Mar 1 00:46:47: xapp_open_receive_chnl: stat 0, eve 0, sid 27, cid 2817, codec 1,
pkt-period 20
*Mar 1 00:46:47: xapp_open_chnl_request: chnl_rec 6311430C
*Mar 1 00:46:47: xapp_open_chnl_request: chnl_rec 6311430C, sess_id 27, conn_id 2817,
cstate 0, nstate 1
*Mar 1 00:46:47: xapp_dequeue_and_process_dspf_events: chnl_rec 6311430C, state 1, eve_id
1
*Mar 1 00:46:47: xapp_open_chnl_success: chnl_rec 6311430C
*Mar 1 00:46:47: xapp_open_chnl_success: chnl_rec 6311430C, sess_id 27, conn_id 2817,
cstate 1, nstate 2, lc_ipaddr 10.10.1.1, lport 16730
*Mar 1 00:46:47: sccp_process_mtp_pdu: appl - 6248FC10, mbuf - 6248FC84
*Mar 1 00:46:47: sccp_process_mtp_pdu: msg_ptr 6248FC8C, len 44, offset 52, msg_id 138
*Mar 1 00:46:47: xapp_start_media_transmission: SCCP stmt_msg - 6248FC8C, appl - 6248FC10
*Mar 1 00:46:47: xapp_search_for_chnl_rec: sess_id 27, conn_id 2817
*Mar 1 00:46:47: xapp_start_media_transmission: chnl_rec 6311430C, stat 2, sid 27, cid
2817, ripaddr 10.10.1.6, rport 18160, codec 1, pkt-period 20, pre 11, silen 16777502, mfpp
1
*Mar 1 00:46:47: xapp_modify_chnl_request: chnl_rec 6311430C
*Mar 1 00:46:47: xapp_modify_chnl_request: chnl_rec 6311430C, sess_id 27, conn_id 2817,
cstate 2, nstate 2
*Mar 1 00:46:47: xapp_dequeue_and_process_dspf_events: chnl_rec 6311430C, state 2, eve_id
4
*Mar 1 00:46:47: xapp_modify_chnl_success: chnl_rec 6311430C, sess_id 27, conn_id 2817,
cstate 2
*Mar 1 00:46:52: sccp_create_application: send keepalive msg, appl 6248F760, appl_type 1,
count 0
*Mar 1 00:46:52: sccp_keepalive: send keepalive id 0, len 4
*Mar 1 00:46:52: sccp_process_mtp_pdu: appl - 6248F760, mbuf - 6248F7D4
*Mar 1 00:46:52: sccp_process_mtp_pdu: msg_ptr 6248F7DC, len 4, offset 12, msg_id 256
*Mar 1 00:46:53: sccp_create_application: send keepalive msg, appl 6248FC10, appl_type 2,
count 0
*Mar 1 00:46:53: sccp_keepalive: send keepalive id 0, len 4
*Mar 1 00:46:54: sccp_process_mtp_pdu: appl - 6248FC10, mbuf - 6248FC84
*Mar 1 00:46:54: sccp_process_mtp_pdu: msg_ptr 6248FC8C, len 4, offset 12, msg_id 256
*Mar 1 00:46:59: sccp_create_application: send keepalive msg, appl 6248F760, appl_type 1,
count 0
*Mar 1 00:46:59: sccp_keepalive: send keepalive id 0, len 4
*Mar 1 00:47:00: sccp_process_mtp_pdu: appl - 6248F760, mbuf - 6248F7D4
*Mar 1 00:47:00: sccp_process_mtp_pdu: msg_ptr 6248F7DC, len 4, offset 12, msg_id 256

```

```

*Mar 1 00:47:01: sccp_create_application: send keepalive msg, appl 6248FC10, appl_type 2,
count 0
*Mar 1 00:47:01: sccp_keepalive: send keepalive id 0, len 4
*Mar 1 00:47:01: sccp_process_mtp_pdu: appl - 6248FC10, mbuf - 6248FC84
*Mar 1 00:47:01: sccp_process_mtp_pdu: msg_ptr 6248FC8C, len 4, offset 12, msg_id 256
*Mar 1 00:47:07: sccp_create_application: send keepalive msg, appl 6248F760, appl_type 1,
count 0
*Mar 1 00:47:07: sccp_keepalive: send keepalive id 0, len 4
*Mar 1 00:47:07: sccp_process_mtp_pdu: appl - 6248F760, mbuf - 6248F7D4
*Mar 1 00:47:07: sccp_process_mtp_pdu: msg_ptr 6248F7DC, len 4, offset 12, msg_id 256
*Mar 1 00:47:08: sccp_create_application: send keepalive msg, appl 6248FC10, appl_type 2,
count 0
*Mar 1 00:47:08: sccp_keepalive: send keepalive id 0, len 4
*Mar 1 00:47:09: sccp_process_mtp_pdu: appl - 6248FC10, mbuf - 6248FC84
*Mar 1 00:47:09: sccp_process_mtp_pdu: msg_ptr 6248FC8C, len 4, offset 12, msg_id 256
*Mar 1 00:47:14: sccp_create_application: send keepalive msg, appl 6248F760, appl_type 1,
count 0
*Mar 1 00:47:14: sccp_keepalive: send keepalive id 0, len 4
*Mar 1 00:47:15: sccp_process_mtp_pdu: appl - 6248F760, mbuf - 6248F7D4
*Mar 1 00:47:15: sccp_process_mtp_pdu: msg_ptr 6248F7DC, len 4, offset 12, msg_id 256
*Mar 1 00:47:16: sccp_create_application: send keepalive msg, appl 6248FC10, appl_type 2,
count 0
*Mar 1 00:47:16: sccp_keepalive: send keepalive id 0, len 4
*Mar 1 00:47:16: sccp_process_mtp_pdu: appl - 6248FC10, mbuf - 6248FC84
*Mar 1 00:47:16: sccp_process_mtp_pdu: msg_ptr 6248FC8C, len 4, offset 12, msg_id 256
*Mar 1 00:47:22: sccp_create_application: send keepalive msg, appl 6248F760, appl_type 1,
count 0
*Mar 1 00:47:22: sccp_keepalive: send keepalive id 0, len 4
*Mar 1 00:47:22: sccp_process_mtp_pdu: appl - 6248F760, mbuf - 6248F7D4
*Mar 1 00:47:22: sccp_process_mtp_pdu: msg_ptr 6248F7DC, len 4, offset 12, msg_id 256
*Mar 1 00:47:23: sccp_create_application: send keepalive msg, appl 6248FC10, appl_type 2,
count 0
*Mar 1 00:47:23: sccp_keepalive: send keepalive id 0, len 4
*Mar 1 00:47:24: sccp_process_mtp_pdu: appl - 6248FC10, mbuf - 6248FC84
*Mar 1 00:47:24: sccp_process_mtp_pdu: msg_ptr 6248FC8C, len 4, offset 12, msg_id 256
*Mar 1 00:47:29: sccp_create_application: send keepalive msg, appl 6248F760, appl_type 1,
count 0
*Mar 1 00:47:29: sccp_keepalive: send keepalive id 0, len 4
*Mar 1 00:47:30: sccp_process_mtp_pdu: appl - 6248F760, mbuf - 6248F7D4
*Mar 1 00:47:30: sccp_process_mtp_pdu: msg_ptr 6248F7DC, len 4, offset 12, msg_id 256
*Mar 1 00:47:31: sccp_create_application: send keepalive msg, appl 6248FC10, appl_type 2,
count 0
*Mar 1 00:47:31: sccp_keepalive: send keepalive id 0, len 4
*Mar 1 00:47:31: sccp_process_mtp_pdu: appl - 6248FC10, mbuf - 6248FC84
*Mar 1 00:47:31: sccp_process_mtp_pdu: msg_ptr 6248FC8C, len 4, offset 12, msg_id 256

```

**Related Commands**

Command	Description
<b>debug frame-relay vc-bundle</b>	Sets debugging levels for the DSP-farm service.
<b>dspfarm (DSP farm)</b>	Enables DSP-farm service.
<b>sccp</b>	Enables SCCP and its associated transcoding and conferencing applications.
<b>show sccp</b>	Displays the SCCP configuration information and current status.

# debug sccp config

To enable Skinny Client Control Protocol (SCCP) event debugging, use the **debug sccp config** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

```
debug sccp config {all | errors | events | parser}
```

```
no debug sccp config {all | errors | events | parser}
```

## Syntax Description

<b>all</b>	Displays all SCCP auto-config debug trace.
<b>errors</b>	Displays SCCP auto-config errors.
<b>events</b>	Displays SCCP auto-config events.
<b>parser</b>	Displays SCCP auto-config parser.

## Command Default

Disabled

## Command Modes

Privileged EXEC

## Command History

Release	Modification
12.3(8)XY	This command was introduced on the Communication Media Module.
12.3(14)T	This command was integrated into Cisco IOS Release 12.3(14)T.
12.4(3)	This command was integrated into Cisco IOS Release 12.4(3).
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 example shows the **debug sccp config** command used to enable SCCP event debugging and to display SCCP auto-configuration events:

```
Router# debug sccp config events
...
Feb  8 02:17:31.119: mp_auto_cfg_request(req_id=2, prof=995, ccm_group_id=0)
Feb  8 02:17:31.123: mp_auto_cfg_is_up: SCCP auto-config is enabled & registered
...
```

Table 304 describes the significant fields shown in the display.

**Table 304** *debug sccp config Field Descriptions*

Field	Description
prof=995	Indicates the profile ID. If generated by media processor auto-configuration, profile IDs are preceded by 99.
SCCP auto-config is enabled & registered	Indicates the registration of sccp when auto-config is complete.

#### Related Commands

Command	Description
<b>auto-config</b>	Enables auto-configuration or enters auto-config application configuration mode for the SCCP application.
<b>debug auto-config</b>	Enables debugging for auto-configuration applications.
<b>show auto-config</b>	Displays the current status of auto-configuration applications.

# debug sccp keepalive

To display granular debugging information by Cisco Call Manager (CCM) group for Simple Client Control Protocol (SCCP) and its related applications (transcoding and conferencing), use the **debug sccp keepalive** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug sccp keepalive [identifier]**

**no debug sccp keepalive**

<b>Syntax Description</b>	<b>identifier</b>	Requests that only keepalive traffic from a specific CCM be reported, for more granular debugging.
---------------------------	-------------------	--

<b>Command Modes</b>	Privileged EXEC (#)
----------------------	---------------------

<b>Command History</b>	<b>Release</b>	<b>Modification</b>
	12.2(5)T	This command was introduced.
	12.4(20)T	The output for this command was modified to identify which CCM the keepalives are being sent to and received from to help users distinguish which devices have proper connectivity and which do not. Prior to this modification there was no information as to which devices keepalive messages were being exchanged between.

**Examples** The following example shows output from the **debug sccp keepalive** command:

```
Router# debug sccp keepalive
```

```
Skinnny Client Control Protocol keepalive messages debugging is on
```

```
May 18 13:53:39.242 EDT: SCCP:send KeepAliveMessage to ccm identifier 1
May 18 13:53:39.242 EDT: SCCP:rcvd KeepAliveAckMessage from ccm identifier 1
May 18 13:53:39.570 EDT: SCCP:send KeepAliveMessage to ccm identifier 2
May 18 13:53:39.570 EDT: SCCP:rcvd KeepAliveAckMessage from ccm identifier 2
May 18 13:54:09.243 EDT: SCCP:send KeepAliveMessage to ccm identifier 1
May 18 13:54:09.243 EDT: SCCP:rcvd KeepAliveAckMessage from ccm identifier 1
May 18 13:54:09.571 EDT: SCCP:send KeepAliveMessage to ccm identifier 2
May 18 13:54:09.571 EDT: SCCP:rcvd KeepAliveAckMessage from ccm identifier 2
May 18 13:54:39.243 EDT: SCCP:send KeepAliveMessage to ccm identifier 1
May 18 13:54:39.243 EDT: SCCP:rcvd KeepAliveAckMessage from ccm identifier 1
May 18 13:54:39.571 EDT: SCCP:send KeepAliveMessage to ccm identifier 2
May 18 13:54:39.571 EDT: SCCP:rcvd KeepAliveAckMessage from ccm identifier 2
```

**Related Commands**

<b>Command</b>	<b>Description</b>
<b>debug sccp</b>	Displays debugging information for SCCP and its related applications (transcoding and conferencing).
<b>sccp</b>	Enables SCCP and its associated transcoding and conferencing applications.

# debug sdlc

To display information on Synchronous Data Link Control (SDLC) frames received and sent by any router serial interface involved in supporting SDLC end station functions, use the **debug sdlc** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug sdlc**

**no debug sdlc**

---

## Syntax Description

This command has no arguments or keywords.

---

## Command Modes

Privileged EXEC

---

## Usage Guidelines



### Note

---

Because the **debug sdlc** command can generate many messages and alter timing in the network node, use it only when instructed by authorized support personnel.

---

---

## Examples

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

```
Router# debug sdlc

SDLC: Sending RR at location 4
Serial3: SDLC O (12495952) C2 CONNECT (2) RR P/F 6
Serial3: SDLC I (12495964) [C2] CONNECT (2) RR P/F 0 (R) [VR: 6 VS: 0]
Serial3: SDLC T [C2] 12496064 CONNECT 12496064 0
SDLC: Sending RR at location 4
Serial3: SDLC O (12496064) C2 CONNECT (2) RR P/F 6
Serial3: SDLC I (12496076) [C2] CONNECT (2) RR P/F 0 (R) [VR: 6 VS: 0]
Serial3: SDLC T [C2] 12496176 CONNECT 12496176 0
```

The following line of output indicates that the router is sending a Receiver Ready packet at location 4 in the code:

```
SDLC: Sending RR at location 4
```

The following line of output describes a frame output event:

```
Serial1/0: SDLC O 04 CONNECT (285) IFRAME P/F 6
```

Table 305 describes the significant fields shown in the display.

**Table 305** *debug sdlc Field Descriptions for a Frame Output Event*

Field	Description
Serial1/0	Interface type and unit number reporting the frame event.
SDLC	Protocol providing the information.
O	Command mode of frame event. Possible values are as follows: <ul style="list-style-type: none"> <li>• I—Frame input</li> <li>• O—Frame output</li> <li>• T—T1 timer expired</li> </ul>
04	SDLC address of the SDLC connection.
CONNECT	State of the protocol when the frame event occurred. Possible values are as follows: <ul style="list-style-type: none"> <li>• CONNECT</li> <li>• DISCONNECT</li> <li>• DISCSENT (disconnect sent)</li> <li>• ERROR (FRMR frame sent)</li> <li>• REJSENT (reject frame sent)</li> <li>• SNRMSSENT (SNRM frame sent)</li> <li>• USBUSY</li> <li>• THEMBUSY</li> <li>• BOTHBUSY</li> </ul>
(285)	Size of the frame (in bytes).
IFRAME	Frame type name. Possible values are as follows: <ul style="list-style-type: none"> <li>• DISC—Disconnect</li> <li>• DM—Disconnect mode</li> <li>• FRMR—Frame reject</li> <li>• IFRAME—Information frame</li> <li>• REJ—Reject</li> <li>• RNR—Receiver not ready</li> <li>• RR—Receiver ready</li> <li>• SIM—Set Initialization mode command</li> <li>• SNRM—Set Normal Response Mode</li> <li>• TEST—Test frame</li> <li>• UA—Unnumbered acknowledgment</li> <li>• XID—EXchange ID</li> </ul>

**Table 305** *debug sdlc Field Descriptions for a Frame Output Event (continued)*

Field	Description
P/F	Poll/Final bit indicator. Possible values are as follows: <ul style="list-style-type: none"> <li>F—Final (printed for Response frames)</li> <li>P—Poll (printed for Command frames)</li> <li>P/F—Poll/Final (printed for RR, RNR, and REJ frames, which can be either Command or Response frames)</li> </ul>
6	Receive count; range: 0 to 7.

The following line of output describes a frame input event:

```
Serial1/0: SDLC I 02 CONNECT (16) IFRAME P 7 0,[VR: 7 VS: 0]
```

[Table 306](#) describes the significant fields shown in the display.

**Table 306** *debug sdlc Field Descriptions for a Frame Input Event*

Field	Description
02	SDLC address.
IFRAME	Traffic engineering type.
P	Poll bit P is on.
VR: 7	Receive count; range: 0 to 7.
VS: 0	Send count; range: 0 to 7.

The following line of output describes a frame timer event:

```
Serial1/0: SDLC T 02 CONNECT 0x9CB69E8 P 0
```

[Table 307](#) describes the significant fields shown in the display.

**Table 307** *debug sdlc Field Descriptions for a Timer Event*

Field	Description
Serial1/0	Interface type and unit number reporting the frame event.
SDLC	Protocol providing the information.
T	Timer has expired.
02	SDLC address of this SDLC connection.

**Table 307** *debug sdlc Field Descriptions for a Timer Event (continued)*

Field	Description
CONNECT	State of the protocol when the frame event occurred. Possible values are as follows: <ul style="list-style-type: none"> <li>• BOTHBUSY</li> <li>• CONNECT</li> <li>• DISCONNECT</li> <li>• DISCSENT (disconnect sent)</li> <li>• ERROR (FRMR frame sent)</li> <li>• REJSENT (reject frame sent)</li> <li>• SNRMSSENT (SNRM frame sent)</li> <li>• THEMBUSY</li> <li>• USBUSY</li> </ul>
0x9CB69E8	Top timer.
0	Retry count; default: 0.

**Related Commands**

Command	Description
<b>debug list</b>	Filters debugging information on a per-interface or per-access list basis.

# debug sdlc local-ack

To display information on the local acknowledgment feature, use the **debug sdlc local-ack** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug sdlc local-ack** *[number]*

**no debug sdlc local-ack** *[number]*

## Syntax Description

*number* (Optional) Frame-type that you want to monitor. See the “Usage Guidelines” section.

## Command Modes

Privileged EXEC

## Usage Guidelines

You can select the frame types you want to monitor; the frame types correspond to bit flags. You can select 1, 2, 4, or 7, which is the decimal value of the bit flag settings. If you select 1, the octet is set to 00000001. If you select 2, the octet is set to 0000010. If you select 4, the octet is set to 00000100. If you want to select all frame types, select 7; the octet is 00000111. The default is 7 for all events. [Table 308](#) defines these bit flags.

**Table 308** *debug sdlc local-ack Debugging Levels*

Debug Command	Meaning
<b>debug sdlc local-ack 1</b>	Only U-Frame events
<b>debug sdlc local-ack 2</b>	Only I-Frame events
<b>debug sdlc local-ack 4</b>	Only S-Frame events
<b>debug sdlc local-ack 7</b>	All Synchronous Data Link Control (SDLC) Local-Ack events (default setting)



### Caution

Because using this command is processor intensive, it is best to use it after hours, rather than in a production environment. It is also best to use this command by itself, rather than in conjunction with other **debugging** commands.

**Examples**

The following is sample output from the **debug sdlc local-ack** command:

Group of associated operations

```
router# debug sdlc local-ack 1
```

```
SLACK (Serial3): Input      = Network, LinkupRequest
SLACK (Serial3): Old State = AwaitSdlcOpen           New State = AwaitSdlcOpen

SLACK (Serial3): Output     = SDLC, SNRM

SLACK (Serial3): Input      = SDLC, UA
SLACK (Serial3): Old State = AwaitSdlcOpen           New State = Active

SLACK (Serial3): Output     = Network, LinkResponse
```

SP560

The first line shows the input to the SDLC local acknowledgment state machine:

```
SLACK (Serial3): Input      = Network, LinkupRequest
```

[Table 309](#) describes the significant fields shown in the display.

**Table 309** *debug sdlc local-ack Field Descriptions*

Field	Description
SLACK	SDLC local acknowledgment feature is providing the information.
(Serial3):	Interface type and unit number reporting the event.
Input = Network	Source of the input.
LinkupRequest	Op code. A LinkupRequest is an example of possible values.

The second line shows the change in the SDLC local acknowledgment state machine. In this case the AwaitSdlcOpen state is an internal state that has not changed while this display was captured.

```
SLACK (Serial3): Old State = AwaitSdlcOpen           New State = AwaitSdlcOpen
```

The third line shows the output from the SDLC local acknowledgment state machine:

```
SLACK (Serial3): Output     = SDLC, SNRM
```

# debug sdlc packet

To display packet information on Synchronous Data Link Control (SDLC) frames received and sent by any router serial interface involved in supporting SDLC end station functions, use the **debug sdlc packet** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug sdlc packet** [*max-bytes*]

**no debug sdlc packet** [*max-bytes*]

## Syntax Description

*max-bytes* (Optional) Limits the number of bytes of data that are printed to the display.

## Command Modes

Privileged EXEC

## Usage Guidelines

This command requires intensive CPU processing; therefore, we recommend not using it when the router is expected to handle normal network loads, such as in a production environment. Instead, use this command when network response is noncritical. We also recommend that you use this command by itself, rather than in conjunction with other **debug** commands.

## Examples

The following is sample output from the **debug sdlc packet** command with the packet display limited to 20 bytes of data:

```
Router# debug sdlc packet 20

Serial3 SDLC Output
00000 C3842C00 02010010 019000C5 C5C5C5C5 Cd.....EEEE
00010 C5C5C5C5                               EEEE
Serial3 SDLC Output
00000 C3962C00 02010011 039020F2           Co.....2
Serial3 SDLC Output
00000 C4962C00 0201000C 039020F2           Do.....2
Serial3 SDLC Input
00000      C491                               Dj
```

# debug serial interface

To display information on a serial connection failure, use the **debug serial interface** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug serial interface**

**no debug serial interface**

---

**Syntax Description**

This command has no arguments or keywords.

---

**Command Modes**

Privileged EXEC

---

**Usage Guidelines**

If the **show interface serial** EXEC command shows that the line and protocol are down, you can use the **debug serial interface** command to isolate a timing problem as the cause of a connection failure. If the keepalive values in the mineseq, yourseen, and myseen fields are not incrementing in each subsequent line of output, there is a timing or line problem at one end of the connection.

**Caution**

Although the **debug serial interface** command typically does not generate a substantial amount of output, nevertheless use it cautiously during production hours. When Switched Multimegabit Data Service (SMDS) is enabled, for example, it can generate considerable output.

The output of the **debug serial interface** command can vary, depending on the type of WAN configured for an interface: Frame Relay, High-Level Data Link Control (HDLC), High-Speed Serial Interface (HSSI), SMDS, or X.25. The output also can vary depending on the type of encapsulation configured for that interface. The hardware platform also can affect **debug serial interface** output.

---

**Examples**

The following sections show and describe sample **debug serial interface** output for various configurations.

**Debug Serial Interface for Frame Relay Encapsulation**

The following message is displayed if the encapsulation for the interface is Frame Relay (or HDLC) and the router attempts to send a packet containing an unknown packet type:

```
Illegal serial link type code xxx
```

**Debug Serial Interface for HDLC**

The following is sample output from the **debug serial interface** command for an HDLC connection when keepalives are enabled. This output shows that the remote router is not receiving all the keepalives the router is sending. When the difference in the myseq and mineseq fields exceeds three, the line goes down and the interface is reset.

```

router# debug serial interface

Serial1: HDLC myseq 636119, mineseen 636119, yourseen 515032, line up
Serial1: HDLC myseq 636120, mineseen 636120, yourseen 515033, line up
Serial1: HDLC myseq 636121, mineseen 636121, yourseen 515034, line up
Serial1: HDLC myseq 636122, mineseen 636122, yourseen 515035, line up
Serial1: HDLC myseq 636123, mineseen 636123, yourseen 515036, line up
Serial1: HDLC myseq 636124, mineseen 636124, yourseen 515037, line up
Serial1: HDLC myseq 636125, mineseen 636125, yourseen 515038, line up
Serial1: HDLC myseq 636126, mineseen 636126, yourseen 515039, line up

1 missed   Serial1: HDLC myseq 636127, mineseen 636127, yourseen 515040, line up
keepalive  Serial1: HDLC myseq 636128, mineseen 636127, yourseen 515041, line up
           Serial1: HDLC myseq 636129, mineseen 636129, yourseen 515042, line up

3 missed   Serial1: HDLC myseq 636130, mineseen 636130, yourseen 515043, line up
keepalives Serial1: HDLC myseq 636131, mineseen 636130, yourseen 515044, line up
;          Serial1: HDLC myseq 636132, mineseen 636130, yourseen 515045, line up
line goes  Serial1: HDLC myseq 636133, mineseen 636130, yourseen 515046, line down
down and  Serial1: HDLC myseq 636127, mineseen 636127, yourseen 515040, line up
interface Serial1: HDLC myseq 636128, mineseen 636127, yourseen 515041, line up
is reset  Serial1: HDLC myseq 636129, mineseen 636129, yourseen 515042, line up

```

S2561

Table 310 describes the significant fields shown in the display.

**Table 310** *debug serial interface Field Descriptions for HDLC*

Field	Description
Serial 1	Interface through which the serial connection is taking place.
HDLC	Serial connection is an HDLC connection.
myseq 636119	Myseq counter increases by one each time the router sends a keepalive packet to the remote router.
mineseen 636119	Value of the mineseen counter reflects the last myseq sequence number the remote router has acknowledged receiving from the router. The remote router stores this value in its yourseen counter and sends that value in a keepalive packet to the router.
yourseen 515032	Yourseen counter reflects the value of the myseq sequence number the router has received in a keepalive packet from the remote router.
line up	Connection between the routers is maintained. Value changes to “line down” if the values of the myseq and myseen fields in a keepalive packet differ by more than three. Value returns to “line up” when the interface is reset. If the line is in loopback mode, (“looped”) appears after this field.

Table 311 describes additional error messages that the **debug serial interface** command can generate for HDLC.

**Table 311** *debug serial interface Error Messages for HDLC*

Field	Description
Illegal serial link type code <xxx>, PC = 0xnnnnnnn	Router attempted to send a packet containing an unknown packet type.
Illegal HDLC serial type code <xxx>, PC = 0xnnnnnn	Unknown packet type is received.
Serial 0: attempting to restart	Interface is down. The hardware is then reset to correct the problem, if possible.
Serial 0: Received bridge packet sent to <nnnnnnnnnn>	Bridge packet is received over a serial interface configured for HDLC, and bridging is not configured on that interface.

**Debug Serial Interface for HSSI**

On an HSSI interface, the **debug serial interface** command can generate the following additional error message:

```
HSSI0: Reset from 0xnnnnnnnn
```

This message indicates that the HSSI hardware has been reset. The 0xnnnnnnnn variable is the address of the routine requesting that the hardware be reset; this value is useful only to development engineers.

**Debug Serial Interface for ISDN Basic Rate**

[Table 312](#) describes error messages that the **debug serial interface** command can generate for ISDN Basic Rate.

**Table 312** *debug serial interface Error Messages for ISDN Basic Rate*

Message	Description
BRI: D-chan collision	Collision on the ISDN D channel has occurred; the software will retry transmission.
Received SID Loss of Frame Alignment int.	ISDN hardware has lost frame alignment. This usually indicates a problem with the ISDN network.
Unexpected IMP int: ipr = 0xnn	ISDN hardware received an unexpected interrupt. The 0xnn variable indicates the value returned by the interrupt register.
BRI(d): RX Frame Length Violation. Length=n BRI(d): RX Nonoctet Aligned Frame BRI(d): RX Abort Sequence BRI(d): RX CRC Error BRI(d): RX Overrun Error BRI(d): RX Carrier Detect Lost	Any of these messages can be displayed when a receive error occurs on one of the ISDN channels. The (d) indicates which channel it is on. These messages can indicate a problem with the ISDN network connection.
BRI0: Reset from 0xnnnnnnnn	BRI hardware has been reset. The 0xnnnnnnnn variable is the address of the routine that requested that the hardware be reset; it is useful only to development engineers.

**Table 312** *debug serial interface Error Messages for ISDN Basic Rate (continued)*

Message	Description
BRI(d): Bad state in SCMs scm1= <i>x</i> scm2= <i>x</i> scm3= <i>x</i> BRI(d): Bad state in SCONs scon1= <i>x</i> scon2 = <i>x</i> scon3= <i>x</i> BRI(d): Bad state ub SCR; SCR= <i>x</i>	Any of these messages can be displayed if the ISDN hardware is not in the proper state. The hardware is then reset. If the message is displayed constantly, it usually indicates a hardware problem.
BRI(d): Illegal packet encapsulation= <i>n</i>	Packet is received, but the encapsulation used for the packet is not recognized. The interface might be misconfigured.

**Debug Serial Interface for an MK5025 Device**

[Table 313](#) describes the additional error messages that the **debug serial interface** command can generate for an MK5025 device.

**Table 313** *debug serial interface Error Messages for an MK5025 Device*

Message	Description
MK5(d): Reset from 0xnnnnnnnn	Hardware has been reset. The 0xnnnnnnnn variable is the address of the routine that requested that the hardware be reset; it is useful only to development engineers.
MK5(d): Illegal packet encapsulation= <i>n</i>	Packet is received, but the encapsulation used for the packet is not recognized. Interface might be misconfigured.
MK5(d): No packet available for packet realignment	Serial driver attempted to get a buffer (memory) and was unable to do so.
MK5(d): Bad state in CSR0=( <i>x</i> )	This message is displayed if the hardware is not in the proper state. The hardware is reset. If this message is displayed constantly, it usually indicates a hardware problem.
MK5(d): New serial state= <i>n</i>	Hardware has interrupted the software. It displays the state that the hardware is reporting.
MK5(d): DCD is down. MK5(d): DCD is up.	If the interrupt indicates that the state of carrier has changed, one of these messages is displayed to indicate the current state of DCD.

**Debug Serial Interface for SMDS Encapsulation**

When encapsulation is set to SMDS, the **debug serial interface** command displays SMDS packets that are sent and received, and any error messages resulting from SMDS packet transmission.

The error messages that the **debug serial interface** command can generate for SMDS follow.

The following message indicates that a new protocol requested SMDS to encapsulate the data for transmission. SMDS is not yet able to encapsulate the protocol.

```
SMDS: Error on Serial 0, encapsulation bad protocol = x
```

The following message indicates that SMDS was asked to encapsulate a packet, but no corresponding destination E.164 SMDS address was found in any of the static SMDS tables or in the ARP tables:

```
SMDS send: Error in encapsulation, no hardware address, type = x
```

The following message indicates that a protocol such as Connectionless Network Service (CLNS) or IP has been enabled on an SMDS interface, but the corresponding multicast addresses have not been configured. The *n* variable displays the link type for which encapsulation was requested.

```
SMDS: Send, Error in encapsulation, type=n
```

The following messages can occur when a corrupted packet is received on an SMDS interface. The router expected *x*, but received *y*.

```
SMDS: Invalid packet, Reserved NOT ZERO, x y  
SMDS: Invalid packet, TAG mismatch x y  
SMDS: Invalid packet, Bad TRAILER length x y
```

The following messages can indicate an invalid length for an SMDS packet:

```
SMDS: Invalid packet, Bad BA length x  
SMDS: Invalid packet, Bad header extension length x  
SMDS: Invalid packet, Bad header extension type x  
SMDS: Invalid packet, Bad header extension value x
```

The following messages are displayed when the **debug serial interface** command is enabled:

```
Interface Serial 0 Sending SMDS L3 packet:  
SMDS: dgsizex type:0xn src:y dst:z
```

If the **debug serial interface** command is enabled, the following message can be displayed when a packet is received on an SMDS interface, but the destination SMDS address does not match any on that interface:

```
SMDS: Packet n, not addressed to us
```

# debug serial lead-transition

To activate the leads status transition debug capability for all capable ports, use the **debug serial lead-transition** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug serial lead-transition**

**no debug serial lead-transition**

**Syntax Description** This command has no arguments or keywords.

**Defaults** Debugging is not turned on.

**Command Modes** Privileged EXEC

## Command History

Release	Modification
Release 12.2(15)ZJ	This command was introduced on the following platforms: Cisco 2610XM, Cisco 2611XM, Cisco 2620XM, Cisco 2621XM, Cisco 2650XM, Cisco 2651XM, Cisco 2691, Cisco 3631, Cisco 3660, Cisco 3725, and Cisco 3745 routers.
Release 12.3(2)T	This command was integrated into Cisco IOS Release 12.3(2)T.
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

To control which port is to be reported and therefore reduce the risk of flooding the console screen with debug information, enter the **debug condition interface serial slot/port** command after using the **debug serial lead-transition** command to set the condition.



### Caution

To avoid having the debug message flood the console screen with debug information, use these commands only when traffic on the IP network is low, so other activity on the system is not adversely affected.

## Examples

The following example shows the serial control leads reported for slot 1, port 1:

```
Router# debug serial lead-transition
Router# debug condition interface serial 1/1

*Mar  1 00:17:15.040:slot(1) Port(1):DSR/DTR is Deasserted
*Mar  1 00:17:15.040:slot(1) Port(1):CTS/RTS is Deasserted

*Mar  1 00:17:47.955:slot(1) Port(1):DCD/Local Loop is Deasserted
```

```
*Mar 1 00:17:47.955:slot(1) Port(1):DSR/DTR is Deasserted
*Mar 1 00:17:47.955:slot(1) Port(1):CTS/RTS is Deasserted

Router# no shut down serial 1/1

*Mar 1 00:16:52.298:slot(1) Port(1):DSR/DTR is Asserted
*Mar 1 00:16:52.298:slot(1) Port(1):CTS/RTS is Asserted

*Mar 1 00:16:31.648:slot(1) Port(1):DCD/Local Loop is Asserted
*Mar 1 00:16:31.648:slot(1) Port(1):DSR/DTR is Asserted
*Mar 1 00:16:31.648:slot(1) Port(1):CTS/RTS is Asserted
```

Table 314 describes significant fields shown in the displays.

**Table 314** *debug serial lead-transition Field Descriptions*

Field	Description
DSR/DTR is Asserted/Deasserted	The DSR or DTE signal is activated or inactivated.
CTS/RTS is Asserted/Deasserted	The CTS or RTS signal is activated or inactivated.
DCD/Local Loop is Asserted/Deasserted	The DCD or Local Loopback signal is activated or inactivated.

#### Related Commands

Command	Description
<b>debug condition interface serial</b>	Enables conditional debugging on a serial interface.

# debug serial packet

To display more detailed serial interface debugging information than you can obtain using the **debug serial interface** command, use the **debug serial packet** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug serial packet**

**no debug serial packet**

---

## Syntax Description

This command has no arguments or keywords.

---

## Command Modes

Privileged EXEC

---

## Usage Guidelines

The **debug serial packet** command generates output that is dependent on the type of serial interface and the encapsulation running on that interface. The hardware platform also can impact **debug serial packet** output.

The **debug serial packet** command displays output for only Switched Multimegabit Data Service (SMDS) encapsulations.

---

## Examples

The following is sample output from the **debug serial packet** command when SMDS is enabled on the interface:

```
Router# debug serial packet

Interface Serial2 Sending SMDS L3 packet:
SMDS Header: Id: 00 RSVD: 00 Bntag: EC Basize: 0044
Dest:E18009999999FFFF Src:C12015804721FFFF Xh:04030000030001000000000000000000
SMDS LLC: AA AA 03 00 00 00 80 38
SMDS Data: E1 19 01 00 00 80 00 00 0C 00 38 1F 00 0A 00 80 00 00 0C 01 2B 71
SMDS Data: 06 01 01 0F 1E 24 00 EC 00 44 00 02 00 00 83 6C 7D 00 00 00 00 00
SMDS Trailer: RSVD: 00 Bntag: EC Length: 0044
```

As the output shows, when encapsulation is set to SMDS, the **debug serial packet** command displays the entire SMDS header (in hexadecimal notation), and some payload data on transmit or receive. This information is useful only when you have an understanding of the SMDS protocol. The first line of the output indicates either Sending or Receiving.

# debug service-group

To enable debugging of service-group events and errors, use the **debug service-group** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

```
debug service-group {all | error | feature | group | interface | ipc | member | qos | stats}
```

```
no debug service-group {all | error | feature | group | interface | ipc | member | qos | stats}
```

## Syntax Description

<b>all</b>	All service-group debugging.
<b>error</b>	Service-group errors.
<b>feature</b>	Service-group features.
<b>group</b>	Service-group events.
<b>interface</b>	Service-group interface events.
<b>ipc</b>	Service-group Inter-Process Communication (IPC) messaging.
<b>member</b>	Service-group member events.
<b>qos</b>	Service-group Quality of Service (QoS).
<b>stats</b>	Service-group statistics.

## Command Modes

Privileged EXEC (#)

## Command History

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

## Examples

In the following example, service-group debugging for service-group member events has been enabled:

```
Router> enable
Router# debug service-group member

%Service Group membership debugging is on
```

# debug service-module

To display debugging information that monitors the detection and clearing of network alarms on the integrated channel service unit/data service unit (CSU/DSU) modules, use the **debug service-module** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug service-module**

**no debug service-module**

---

**Syntax Description** This command has no arguments or keywords.

---

**Command Modes** Privileged EXEC

---

**Usage Guidelines** Use this command to enable and disable debug logging for the serial 0 and serial 1 interfaces when an integrated CSU/DSU is present. This command enables debugging on all interfaces. Network alarm status can also be viewed through the use of the **show service-module** command.

**Note**

---

The debug output varies depending on the type of service module installed in the router.

---

---

**Examples** The following is sample output from the **debug service-module** command:

```
Router# debug service-module

SERVICE_MODULE(1): loss of signal ended after duration 00:05:36
SERVICE_MODULE(1): oos/oof ended after duration 01:05:14
SERVICE_MODULE(0): Unit has no clock
SERVICE_MODULE(0): detects loss of signal
SERVICE_MODULE(0): loss of signal ended after duration 00:00:33
```

# debug sgbp dial-bids

To display large-scale dial-out negotiations between the primary network access server (NAS) and alternate NASs, use the **debug sgbp dial-bids** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug sgbp dial-bids**

**no debug sgbp dial-bids**

## Syntax Description

This command has no arguments or keywords.

## Command Modes

Privileged EXEC

## Usage Guidelines

Use this command only when the **sgbp dial-bids** command has been configured.

## Examples

The following is sample output from the **debug sgbp dial-bids** command:

```
Router# debug sgbp dial-bids

*Jan 1 00:25:03.643: SGBP-RES: New bid add request: 4B0 8 2 1 DAC0 1 1
This indicates a new dialout bid has started.
*Jan 1 00:25:03.643: SGBP-RES: Sent Discover message to ID 7B09B71E 49 bytes
The bid request has been sent.
*Jan 1 00:25:03.647: SGBP-RES: Received Message of 49 length:

*Jan 1 00:25:03.647: SGBP-RES: header 5 30 0 31
2 0 0 2D 0 0 0 0 0 0 0 3 0 0 0 1 1E AF 3A 41 7B 9 B7 1E 8 15 B
3 2 C 6 0 0 DA C0 D 4 0 0 E 3 1 F 3 1
*Jan 1 00:25:03.647:
*Jan 1 00:25:03.647: SGBP RES: Scan: Message type: Offer
*Jan 1 00:25:03.647: SGBP RES: Scan: Len is 45
*Jan 1 00:25:03.647: SGBP RES: Scan: Transaction ID: 3
*Jan 1 00:25:03.647: SGBP RES: Scan: Message ID: 1
*Jan 1 00:25:03.647: SGBP RES: Scan: Client ID: 1EAF3A41
*Jan 1 00:25:03.651: SGBP RES: Scan: Server ID: 7B09B71E
*Jan 1 00:25:03.651: SGBP RES: Scan: Resource type 8 length 21
*Jan 1 00:25:03.651: SGBP RES: Scan: Phy-Port Media type: ISDN
*Jan 1 00:25:03.651: SGBP RES: Scan: Phy-Port Min BW: 56000
*Jan 1 00:25:03.651: SGBP RES: Scan: Phy-Port Num Links: 0
*Jan 1 00:25:03.651: SGBP RES: Scan: Phy-Port User class: 1
*Jan 1 00:25:03.651: SGBP RES: Scan: Phy-Port Priority: 1
*Jan 1 00:25:03.651: SGBP-RES: received 45 length Offer packet
*Jan 1 00:25:03.651: SGBP-RES: Offer from 7B09B71E for Transaction 3 accepted
*Jan 1 00:25:03.651: SGBP RES: Server is uncongested. Immediate win
An alternate network access server has responded and won the bid.
*Jan 1 00:25:03.651: SGBP-RES: Bid Succeeded handle 7B09B71E Server-id 4B0
*Jan 1 00:25:03.651: SGBP-RES: Sent Dial-Req message to ID 7B09B71E 66 bytes
The primary network access server has asked the alternate server to dial.
*Jan 1 00:25:04.651: SGBP-RES: QScan: Purging entry
*Jan 1 00:25:04.651: SGBP-RES: deleting entry 6112E204 1EAF3A41 from list...
```

# debug sgbp error

To display debugging messages about routing problems between members of a stack group, use the **debug sgbp error** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug sgbp error**

**no debug sgbp error**

## Syntax Description

This command has no arguments or keywords.

## Command Modes

Privileged EXEC

## Command History

Release	Modification
11.2(9)	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.

## Usage Guidelines

Enter the **debug sgbp error** command to enable the display of debugging messages about routing problems between members of a stack group.



### Note

In unusual cases you may see debugging messages that are not documented on this command reference page. These debugging messages are intended for expert diagnostic interpretation by the Cisco Technical Assistance Center (TAC).

## Examples

One common configuration error is setting a source IP address for a stack member that does not match the locally defined IP address for the same stack member. The following debugging output shows the error message that results from this misconfiguration:

```
Systema# debug sgbp error
```

```
%SGBP-7-DIFFERENT - systemb's addr 10.1.1.2 is different from hello's addr 10.3.4.5
```

This error means that the source IP address of the Stack Group Bidding Protocol (SGBP) hello message received from systemb does not match the IP address configured locally for systemb (through the **sgbp member** command). Correct this configuration error by going to systemb and checking for multiple interfaces by which the SGBP hello can send the message.

Another common error message is:

```
Systema# debug sgbp error
```

```
%SGBP-7-MISCONF, Possible misconfigured member routerk (10.1.1.6)
```

This error message means that routerk is not defined locally, but is defined on another stack member. Correct this configuration error by defining routerk across all members of the stack group using the **sgbp member** command.

The following error message indicates that an SGBP peer is leaving the stack group:

```
Systema# debug sgbp error

%SGBP-7-LEAVING:Member systemc leaving group stack1
```

This error message indicates that the peer systemc is leaving the stack group. Systemc could be leaving the stack group intentionally, or a connectivity problem may exist.

The following error message indicates that an SGBP event was detected from an unknown peer:

```
Systema# debug sgbp error

%SGBP-7-UNKNOWPEER:Event 0x10 from peer at 172.21.54.3
```

An SGBP event came from a network host that was not recognizable as an SGBP peer. Check to see if a network media error could have corrupted the address, or if peer equipment is malfunctioning to generate corrupted packets. Depending on the network topology and firewall of your network, SGBP packets from a nonpeer host could indicate probing and attempts to breach security.



#### Note

If there is a chance your network is under attack, obtain knowledgeable assistance from TAC.

#### Related Commands

Command	Description
<b>debug sgbp hellos</b>	Displays debugging messages for authentication between stack group members.
<b>sgbp group</b>	Defines a named stack group and makes this router a member of that stack group.
<b>sgbp member</b>	Specifies the hostname and IP address of a router or access server that is a peer member of a stack group.
<b>show sgbp</b>	Displays the status of the stack group members.
<b>username</b>	Establishes a username-based authentication system.

# debug sgbp hellos

To display debugging messages for authentication between stack members, use the **debug sgbp hellos** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug sgbp hellos**

**no debug sgbp hellos**

## Syntax Description

This command has no arguments or keywords.

## Command Modes

Privileged EXEC

## Command History

Release	Modification
11.2(9)	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.

## Usage Guidelines

Use the **debug sgbp hellos** command to enable the display of debugging messages for authentication between routers configured as members of a stack group.



### Note

In unusual cases you may see debugging messages that are not documented on this command reference page. These debugging messages are intended for expert diagnostic interpretation by the Cisco Technical Assistance Center (TAC).

## Examples

The following output from the **debug sgbp hellos** command shows systema sending a successful Challenge Handshake Authentication Protocol (CHAP) challenge to and receiving a response from systemb. Similarly, systemb sends out a challenge and receives a response from systema.

```
systema# debug sgbp hellos

%SGBP-7-CHALLENGE: Send Hello Challenge to systemb group stack1
%SGBP-7-CHALLENGED: Hello Challenge message from member systemb (10.1.1.2)
%SGBP-7-RESPONSE: Send Hello Response to systemb group stack1
%SGBP-7-CHALLENGE: Send Hello Challenge to systemb group stack1
%SGBP-7-RESPONDED: Hello Response message from member systemb (10.1.1.2)
%SGBP-7-AUTHOK: Send Hello Authentication OK to member systemb (10.1.1.2)
%SGBP-7-INFO: Addr = 10.1.1.2 Reference = 0xC347DF7
%SGBP-5-ARRIVING: New peer event for member systemb
```

This debug output is self-explanatory.

If authentication fails, you may see one of the following messages in your debug output:

```
%SGBP-7-AUTHFAILED - Member systemb failed authentication
```

This error message means that the remote systemb password for the stack group does not match the password defined on systema. To correct this error, make sure that both systema and systemb have the same password defined using the **username** command.

```
%SGBP-7-NORESP -Fail to respond to systemb group stack1, may not have password.
```

This error message means that systema does not have a username or password defined. To correct this error, define a common group password across all stack members using the **username** command.

**Related Commands**

<b>Command</b>	<b>Description</b>
<b>debug sgbp error</b>	Displays debugging messages about routing problems between members of a stack group.
<b>sgbp group</b>	Defines a named stack group and makes this router a member of that stack group.
<b>sgbp member</b>	Specifies the hostname and IP address of a router or access server that is a peer member of a stack group.
<b>show sgbp</b>	Displays the status of the stack group members.
<b>username</b>	Establishes a username-based authentication system.

# debug sgcp

To debug the Simple Gateway Control Protocol (SGCP), use the **debug sgcp** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

```
debug sgcp {errors | events | packet}
```

```
no debug sgcp {errors | events | packet}
```

## Syntax Description

<b>errors</b>	Displays debug information about SGCP errors.
<b>events</b>	Displays debug information about SGCP events.
<b>packet</b>	Displays debug information about SGCP packets.

## Command Modes

Privileged EXEC

## Command History

Release	Modification
12.0(5)T	This command was introduced.
12.0(7)T	Support for this command was extended to the Cisco uBR924 cable access router.
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

See the following examples to enable and disable debugging at the specified level:

```
Router# debug sgcp errors
```

```
Simple Gateway Control Protocol errors debugging is on
```

```
Router# no debug sgcp errors
```

```
Simple Gateway Control Protocol errors debugging is off
```

```
Router#
```

```
Router# debug sgcp events
```

```
Simple Gateway Control Protocol events debugging is on
```

```
Router# no debug sgcp events
```

```
Simple Gateway Control Protocol events debugging is off
```

```
Router#
```

```
Router# debug sgcp packet
```

```
Simple Gateway Control Protocol packets debugging is on
```

```
Router# no debug sgcp packet
```

```
Simple Gateway Control Protocol packets debugging is off
```

**Related Commands**

<b>Command</b>	<b>Description</b>
<b>sgcp</b>	Starts and allocates resources for the SCGP daemon.

# debug sgcp errors

To debug Simple Gateway Control Protocol (SGCP) errors, use the **debug sgcp errors** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug sgcp errors** [**endpoint** *string*]

**no debug sgcp errors**

<b>Syntax Description</b>	<p><b>endpoint</b> <i>string</i></p> <p>(Optional) Specifies the endpoint string if you want to debug SGCP errors for a specific endpoint.</p> <p>On the Cisco MC3810 router, the endpoint string syntax takes the following forms:</p> <ul style="list-style-type: none"> <li>DS1 endpoint: <b>DS1</b>-<i>slot/port</i></li> <li>POTS endpoint: <b>aaln</b>/<i>slot/port</i></li> </ul> <p>On the Cisco 3600 router, the endpoint string syntax takes the following forms:</p> <ul style="list-style-type: none"> <li>DS1 endpoint: <i>slot/subunit</i>/<b>DS1</b>-<i>ds1 number/ds0 number</i></li> <li>POTS endpoint: <b>aaln</b>/<i>slot/subunit/port</i></li> </ul>
---------------------------	--

<b>Defaults</b>	No default behavior or values
-----------------	-------------------------------

<b>Command Modes</b>	Privileged EXEC
----------------------	-----------------

<b>Command History</b>	<b>Release</b>	<b>Modification</b>
	12.0(5)T	This command was introduced on the Cisco AS5300 access server in a private release that was not generally available.
	12.0(7)XK	Support for this command was extended to the Cisco MC3810 and the Cisco 3600 series routers (except for the Cisco 3620). Also, the <b>endpoint</b> keyword was 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.

<b>Examples</b>	The following example shows the debugging of SGCP errors being enabled:
-----------------	---

```
Router# debug sgcp errors
```

```
Simple Gateway Control Protocol errors debugging is on
no errors since call went through successfully.
```

The following example shows a debug trace for SGCP errors on a specific endpoint:

```
Router# debug sgcp errors endpoint DS1-0/1

End point name for error debug:DS1-0/1 (1)
00:08:41:DS1 = 0, DS0 = 1
00:08:41:Call record found
00:08:41:Enable error end point debug for (DS1-0/1)
```

**Related Commands**

Command	Description
<b>debug rtpspi all</b>	Debugs all RTP SPI errors, sessions, and in/out functions.
<b>debug rtpspi errors</b>	Debugs RTP SPI errors.
<b>debug rtpspi inout</b>	Debugs RTP SPI in/out functions.
<b>debug rtpspi send-nse</b>	Triggers the RTP SPI to send a triple redundant NSE.
<b>debug sgcp events</b>	Debugs SGCP events.
<b>debug sgcp packet</b>	Debugs SGCP packets.
<b>debug vtsp send-nse</b>	Sends and debugs a triple redundant NSE from the DSP to a remote gateway.

# debug sgcp events

To debug Simple Gateway Control Protocol (SGCP) events, use the **debug sgcp events** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug sgcp events** [**endpoint** *string*]

**no debug sgcp events**

<b>Syntax Description</b>	<p><b>endpoint</b> <i>string</i></p> <p>(Optional) Specifies the endpoint string if you want to debug SGCP errors for a specific endpoint.</p> <p>On the Cisco MC3810 router, the endpoint string syntax takes the following forms:</p> <ul style="list-style-type: none"> <li>DS1 endpoint: <b>DS1</b>-<i>slot/port</i></li> <li>POTS endpoint: <b>aaln</b>/<i>slot/port</i></li> </ul> <p>On the Cisco 3600 router, the endpoint string syntax takes the following forms:</p> <ul style="list-style-type: none"> <li>DS1 endpoint: <i>slot/subunit</i>/<b>DS1</b>-<i>ds1 number</i>/<i>ds0 number</i></li> <li>POTS endpoint: <b>aaln</b>/<i>slot/subunit/port</i></li> </ul>								
<b>Defaults</b>	No default behavior or values								
<b>Command Modes</b>	Privileged EXEC								
<b>Command History</b>	<table border="1"> <thead> <tr> <th>Release</th> <th>Modification</th> </tr> </thead> <tbody> <tr> <td>12.0(5)T</td> <td>This command was introduced on the Cisco AS5300 access server in a private release that was not generally available.</td> </tr> <tr> <td>12.0(7)XK</td> <td>Support for this command was extended to the Cisco MC3810 and the Cisco 3600 series routers (except for the Cisco 3620 router). Also, the <b>endpoint</b> keyword was added.</td> </tr> <tr> <td>12.2SX</td> <td>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.</td> </tr> </tbody> </table>	Release	Modification	12.0(5)T	This command was introduced on the Cisco AS5300 access server in a private release that was not generally available.	12.0(7)XK	Support for this command was extended to the Cisco MC3810 and the Cisco 3600 series routers (except for the Cisco 3620 router). Also, the <b>endpoint</b> keyword was 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.
Release	Modification								
12.0(5)T	This command was introduced on the Cisco AS5300 access server in a private release that was not generally available.								
12.0(7)XK	Support for this command was extended to the Cisco MC3810 and the Cisco 3600 series routers (except for the Cisco 3620 router). Also, the <b>endpoint</b> keyword was 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.								

## Examples

The following example shows a debug trace for SGCP events on a specific endpoint:

```
Router# debug sgcp events endpoint DS1-0/1

End point name for event debug:DS1-0/1 (1)
00:08:54:DS1 = 0, DS0 = 1
00:08:54:Call record found
00:08:54:Enable event end point debug for (DS1-0/1)
```

The following example shows a debug trace for all SGCP events on a gateway:

```
Router# debug sgcp events
```

```
*Mar 1 01:13:31.035:callp :19196BC, state :0, call ID :-1, event :23

*Mar 1 01:13:31.035:voice_if->call_agent_ipaddr used as Notify entityNotify entity
available for Tx SGCP msg
NTFY send to ipaddr=1092E01 port=2427
*Mar 1 01:13:31.039:Push msg into SGCP wait ack queue* (1)[25]
*Mar 1 01:13:31.039:Timed Out interval [1):(2000)
*Mar 1 01:13:31.039:Timed Out interval [1):(2000)(0):E[25]
*Mar 1 01:13:31.075:Removing msg :
NTFY 25 ds1-1/13@mc1 SGCP 1.1
X:358258758
O:hd

*Mar 1 01:13:31.075:Unqueue msg from SGCP wait ack q** (0)[25]DS1 = 1, DS0 = 13

*Mar 1 01:13:31.091:callp :19196BC, vdbptr :1964EEC, state :1
*Mar 1 01:13:31.091:Checking ack (trans ID 237740140) :

*Mar 1 01:13:31.091:is_capability_ok:caps.codec=5, caps.pkt=10, caps.nt=8
*Mar 1 01:13:31.091:is_capability_ok:supported signal=0x426C079C, signal2=0x80003,
event=0x6003421F, event2=0x3FD
requested signal=0x0, signal2=0x0,
event=0x20000004, event2=0xC
*Mar 1 01:13:31.091:Same digit map is download (ds1-1/13@mc1)

*Mar 1 01:13:31.091:R:requested trans_id (237740140)

*Mar 1 01:13:31.091:process_signal_ev:seizure possible=1, signal mask=0x4, mask2=0x0
*Mar 1 01:13:32.405:SGCP Session Appl:ignore CCAPI event 10

*Mar 1 01:13:32.489:callp :19196BC, state :1, call ID :16, event :9

*Mar 1 01:13:32.610:SGCP Session Appl:ignore CCAPI event 10

*Mar 1 01:13:32.670:callp :19196BC, state :1, call ID :16, event :9

*Mar 1 01:13:32.766:SGCP Session Appl:ignore CCAPI event 10

*Mar 1 01:13:32.810:callp :19196BC, state :1, call ID :16, event :9

*Mar 1 01:13:32.931:SGCP Session Appl:ignore CCAPI event 10

*Mar 1 01:13:32.967:callp :19196BC, state :1, call ID :16, event :9

*Mar 1 01:13:33.087:SGCP Session Appl:ignore CCAPI event 10

*Mar 1 01:13:33.132:callp :19196BC, state :1, call ID :16, event :9

*Mar 1 01:13:33.240:SGCP Session Appl:ignore CCAPI event 10

*Mar 1 01:13:33.280:callp :19196BC, state :1, call ID :16, event :9

*Mar 1 01:13:33.389:SGCP Session Appl:ignore CCAPI event 10

*Mar 1 01:13:33.433:callp :19196BC, state :1, call ID :16, event :9

*Mar 1 01:13:33.537:SGCP Session Appl:ignore CCAPI event 10

*Mar 1 01:13:33.581:callp :19196BC, state :1, call ID :16, event :9
```

```

*Mar 1 01:13:33.702:SGCP Session Appl:ignore CCAPI event 10

*Mar 1 01:13:33.742:callp :19196BC, state :1, call ID :16, event :9

*Mar 1 01:13:33.742:voice_if->call_agent_ipaddr used as Notify entityNotify entity
available for Tx SGCP msg
NTFY send to ipaddr=1092E01 port=2427
*Mar 1 01:13:33.742:Push msg into SGCP wait ack queue* (1)[26]
*Mar 1 01:13:33.742:Timed Out interval [1]:(2000)
*Mar 1 01:13:33.742:Timed Out interval [1]:(2000)(0):E[26]
*Mar 1 01:13:33.786:Removing msg :
NTFY 26 ds1-1/13@mc1 SGCP 1.1
X:440842371
O:k0, 4081037, s0

*Mar 1 01:13:33.786:Unqueue msg from SGCP wait ack q** (0)[26]DS1 = 1, DS0 = 13

*Mar 1 01:13:33.802:callp :19196BC, vdbptr :1964EEC, state :1
*Mar 1 01:13:33.802:Checking ack (trans ID 698549528) :

*Mar 1 01:13:33.802:is_capability_ok:caps.codec=5, caps.pkt=10, caps.nt=8
*Mar 1 01:13:33.802:is_capability_ok:supported signal=0x426C079C, signal2=0x80003,
event=0x6003421F, event2=0x3FD
requested signal=0x0, signal2=0x0,
event=0x4, event2=0x0
*Mar 1 01:13:33.802:R:requested trans_id (698549528)

*Mar 1 01:13:33.802:set_up_voip_call_leg:peer_addr=0, peer_port=0.
*Mar 1 01:13:33.806:call_setting_crcx:Enter CallProceeding state rc = 0, call_id=16

*Mar 1 01:13:33.806:callp :19196BC, state :4, call ID :16, event :31

*Mar 1 01:13:33.810:callp :1AF5798, state :2, call ID :17, event :8
call_pre_bridge!

*Mar 1 01:13:33.810:send_oc_create_ack:seizure_possiblle=1, ack-lready-sent=0, ack_send=0
*Mar 1 01:13:33.814:callp :1AF5798, state :4, call ID :17, event :28

*Mar 1 01:13:33.814:Call Connect:Raw Msg ptr=0x1995360, no-offhook=0; call-id=17
*Mar 1 01:13:33.814:SGCP Session Appl:ignore CCAPI event 37

*Mar 1 01:13:33.947:callp :19196BC, state :5, call ID :16, event :32
process_nse_on_orig
DS1 = 1, DS0 = 13

*Mar 1 01:13:34.007:callp :19196BC, vdbptr :1964EEC, state :5
*Mar 1 01:13:34.007:Checking ack (trans ID 123764791) :

*Mar 1 01:13:34.007:is_capability_ok:caps.codec=5, caps.pkt=10, caps.nt=8
*Mar 1 01:13:34.007:is_capability_ok:supported signal=0x426C079C, signal2=0x80003,
event=0x6003421F, event2=0x3FD
requested signal=0x0, signal2=0x0,
event=0x4, event2=0x0
*Mar 1 01:13:34.007:R:requested trans_id (123764791)

*Mar 1 01:13:34.007:process_signal_ev:seizure possible=1, signal mask=0x0, mask2=0x0
*Mar 1 01:13:34.007:modify_connection:echo_cancel=1.
*Mar 1 01:13:34.007:modify_connection:vad=0.
*Mar 1 01:13:34.007:modify_connection:peer_addr=6000001, peer_port=0->16500.
*Mar 1 01:13:34.007:modify_connection:conn_mode=2.
*Mar 1 01:13:34.011:callp :19196BC, state :5, call ID :16, event :31

*Mar 1 01:13:34.011:callp :1AF5798, state :5, call ID :17, event :31
process_nse_event

```

```

*Mar 1 01:13:34.051:callp :19196BC, state :5, call ID :16, event :39

*Mar 1 01:13:34.051:call_id=16, ignore_ccapi_ev:ignore 19 for state 5
DS1 = 1, DS0 = 13

*Mar 1 01:13:39.497:callp :19196BC, vdbptr :1964EEC, state :5
*Mar 1 01:13:39.497:Checking ack (trans ID 553892443) :

*Mar 1 01:13:39.497:is_capability_ok:caps.codec=5, caps.pkt=10, caps.nt=8
*Mar 1 01:13:39.497:is_capability_ok:supported signal=0x426C079C, signal2=0x80003,
event=0x6003421F, event2=0x3FD
requested signal=0x8, signal2=0x0,
event=0x4, event2=0x0
*Mar 1 01:13:39.497:R:requested trans_id (553892443)

*Mar 1 01:13:39.497:process_signal_ev:seizure possible=1, signal mask=0x0, mask2=0x0
*Mar 1 01:13:39.497:modify_connection:echo_cancel=1.
*Mar 1 01:13:39.497:modify_connection:vad=0.
*Mar 1 01:13:39.497:modify_connection:peer_addr=6000001, peer_port=16500->16500.
*Mar 1 01:13:39.497:modify_connection:conn_mode=3.
*Mar 1 01:13:39.497:callp :19196BC, state :5, call ID :16, event :31

*Mar 1 01:13:39.501:callp :1AF5798, state :5, call ID :17, event :31

*Mar 1 01:14:01.168:Removing ack (trans ID 237740140) :
200 237740140 OK

*Mar 1 01:14:03.883:Removing ack (trans ID 698549528) :
200 698549528 OK
I:7

v=0
c=IN IP4 5.0.0.1
m=audio 16400 RTP/AVP 0

*Mar 1 01:14:04.087:Removing ack (trans ID 123764791) :
200 123764791 OK
I:7

v=0
c=IN IP4 5.0.0.1
m=audio 16400 RTP/AVP 0

*Mar 1 01:14:09.573:Removing ack (trans ID 553892443) :
200 553892443 OK
I:7

v=0
c=IN IP4 5.0.0.1
m=audio 16400 RTP/AVP 0

*Mar 1 01:14:48.091:callp :19196BC, state :5, call ID :16, event :12

*Mar 1 01:14:48.091:voice_if->call_agent_ipaddr used as Notify entityNotify entity
available for Tx SGCP msg
NTFY send to ipaddr=1092E01 port=2427
*Mar 1 01:14:48.091:Push msg into SGCP wait ack queue* (1)[27]
*Mar 1 01:14:48.091:Timed Out interval [1):(2000)
*Mar 1 01:14:48.091:Timed Out interval [1):(2000) (0):E[27]
*Mar 1 01:14:48.128:Removing msg :

```

```

NTFY 27 dsl-1/13@mc1 SGCP 1.1
X:97849341
O:hu

*Mar 1 01:14:48.128:Unqueue msg from SGCP wait ack q** (0)[27]DS1 = 1, DS0 = 13

*Mar 1 01:14:48.212:callp :19196BC, vdbptr :1964EEC, state :5
*Mar 1 01:14:48.212:Checking ack (trans ID 79307869) :

*Mar 1 01:14:48.212:is_capability_ok:caps.codec=5, caps.pkt=10, caps.nt=8
*Mar 1 01:14:48.212:is_capability_ok:supported signal=0x426C079C, signal2=0x80003,
event=0x6003421F, event2=0x3FD
requested signal=0x4, signal2=0x0,
event=0x0, event2=0x0
*Mar 1 01:14:48.212:delete_call:callp:19196BC, call ID:16
*Mar 1 01:14:48.212:sgcp delete_call:Setting disconnect_by_dlcx to 1
*Mar 1 01:14:48.216:callp :1AF5798, state :6, call ID :17, event :29

*Mar 1 01:14:48.216:Call disconnect:Raw Msg ptr = 0x0, call-id=17
*Mar 1 01:14:48.216:disconnect_call_leg O.K. call_id=17
*Mar 1 01:14:48.216:SGCP:Call disconnect:No need to send onhook
*Mar 1 01:14:48.216:Call disconnect:Raw Msg ptr = 0x19953B0, call-id=16
*Mar 1 01:14:48.216:disconnect_call_leg O.K. call_id=16
*Mar 1 01:14:48.220:callp :1AF5798, state :7, call ID :17, event :13

*Mar 1 01:14:48.220:Processing DLCX signal request :4, 0, 0

*Mar 1 01:14:48.220:call_disconnected:call_id=17, peer 16 is not idle yet.DS1 = 1, DS0 =
13

*Mar 1 01:14:48.272:callp :19196BC, vdbptr :1964EEC, state :7
*Mar 1 01:14:48.272:Checking ack (trans ID 75540355) :

*Mar 1 01:14:48.272:is_capability_ok:caps.codec=5, caps.pkt=10, caps.nt=8
*Mar 1 01:14:48.272:is_capability_ok:supported signal=0x426C079C, signal2=0x80003,
event=0x6003421F, event2=0x3FD
requested signal=0x0, signal2=0x0,
event=0x8, event2=0x0
*Mar 1 01:14:48.272:R:requested trans_id (75540355)

*Mar 1 01:14:48.272:process_signal_ev:seizure possible=1, signal mask=0x4, mask2=0x0
*Mar 1 01:14:49.043:callp :19196BC, state :7, call ID :16, event :27

*Mar 1 01:14:49.043:process_call_feature:Onhook event
*Mar 1 01:14:49.043:callp :19196BC, state :7, call ID :16, event :13

*Mar 1 01:15:18.288:Removing ack (trans ID 79307869) :
250 79307869 OK

*Mar 1 01:15:18.344:Removing ack (trans ID 75540355) :
200 75540355 OK

```

**Related Commands**

<b>Command</b>	<b>Description</b>
<b>debug rtpspi all</b>	Debugs all RTP SPI errors, sessions, and in/out functions.
<b>debug rtpspi errors</b>	Debugs RTP SPI errors.
<b>debug rtpspi inout</b>	Debugs RTP SPI in/out functions.
<b>debug rtpspi send-nse</b>	Triggers the RTP SPI to send a triple redundant NSE.
<b>debug sgcp errors</b>	Debugs SGCP errors.
<b>debug sgcp packet</b>	Debugs SGCP packets.
<b>debug vtsp send-nse</b>	Sends and debugs a triple redundant NSE from the DSP to a remote gateway.

# debug sgcp packet

To debug the Simple Gateway Control Protocol (SGCP), use the **debug sgcp packet** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug sgcp packet** [**endpoint** *string*]

**no debug sgcp packet**

<b>Syntax Description</b>	<p><b>endpoint</b> <i>string</i></p> <p>(Optional) Specifies the endpoint string if you want to debug SGCP errors for a specific endpoint.</p> <p>On the Cisco MC3810, the endpoint string syntax takes the following forms:</p> <ul style="list-style-type: none"> <li>DS1 endpoint: <b>DS1</b>-<i>slot/port</i></li> <li>POTS endpoint: <b>aaln</b>/<i>slot/port</i></li> </ul> <p>On the Cisco 3600, the endpoint string syntax takes the following forms:</p> <ul style="list-style-type: none"> <li>DS1 endpoint: <i>slot/subunit</i>/<b>DS1</b>-<i>ds1 number/ds0 number</i></li> <li>POTS endpoint: <b>aaln</b>/<i>slot/subunit/port</i></li> </ul>
---------------------------	--

<b>Defaults</b>	No default behavior or values
-----------------	-------------------------------

<b>Command Modes</b>	Privileged EXEC
----------------------	-----------------

<b>Command History</b>	<b>Release</b>	<b>Modification</b>
	12.0(5)T	This command was introduced on the Cisco AS5300 in a private release that was not generally available.
	12.0(7)XK	Support for this command was extended to the Cisco MC3810 and the Cisco 3600 series routers (except for the Cisco 3620). Also, the <b>endpoint</b> keyword was 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.

<b>Examples</b>	The following example shows a debug trace for SGCP packets on a specific endpoint:
-----------------	--

```
Router# debug sgcp packet endpoint DS1-0/1

End point name for packet debug:DS1-0/1 (1)
00:08:14:DS1 = 0, DS0 = 1
00:08:14:Enable packet end point debug for (DS1-0/1)
```

The following example shows a debug trace for all SGCP packets on a gateway:

Router# **debug sgcp packet**

```
*Mar 1 01:07:45.204:SUCCESS:Request ID string building is OK
*Mar 1 01:07:45.204:SUCCESS:Building SGCP Parameter lines is OK
*Mar 1 01:07:45.204:SUCCESS:SGCP message building OK
*Mar 1 01:07:45.204:SUCCESS:END of building
*Mar 1 01:07:45.204:SGCP Packet sent --->
NTFY 22 ds1-1/13@mc1 SGCP 1.1
X:550092018
O:hd
<---

*Mar 1 01:07:45.204:NTFY Packet sent successfully.
*Mar 1 01:07:45.240:Packet received -

200 22

*Mar 1 01:07:45.244:SUCCESS:SGCP Header parsing was OK
*Mar 1 01:07:45.244:SUCCESS:END of Parsing
*Mar 1 01:07:45.256:Packet received -

RQNT 180932866 ds1-1/13@mc1 SGCP 1.1
X:362716780
R:hu,k0(A),s0(N),[0-9T](A) (D)
D:(9xx|xxxxxxx)

*Mar 1 01:07:45.256:SUCCESS:SGCP Header parsing was OK
*Mar 1 01:07:45.256:SUCCESS:Request ID string(362716780) parsing is OK
*Mar 1 01:07:45.260:SUCCESS:Requested Event parsing is OK
*Mar 1 01:07:45.260:SUCCESS:Digit Map parsing is OK
*Mar 1 01:07:45.260:SUCCESS:END of Parsing
*Mar 1 01:07:45.260:SUCCESS:SGCP message building OK
*Mar 1 01:07:45.260:SUCCESS:END of building
*Mar 1 01:07:45.260:SGCP Packet sent --->
200 180932866 OK

<---

*Mar 1 01:07:47.915:SUCCESS:Request ID string building is OK
*Mar 1 01:07:47.915:SUCCESS:Building SGCP Parameter lines is OK
*Mar 1 01:07:47.919:SUCCESS:SGCP message building OK
*Mar 1 01:07:47.919:SUCCESS:END of building
*Mar 1 01:07:47.919:SGCP Packet sent --->
NTFY 23 ds1-1/13@mc1 SGCP 1.1
X:362716780
O:k0, 4081037, s0
<---

*Mar 1 01:07:47.919:NTFY Packet sent successfully.
*Mar 1 01:07:47.955:Packet received -

200 23

*Mar 1 01:07:47.955:SUCCESS:SGCP Header parsing was OK
*Mar 1 01:07:47.955:SUCCESS:END of Parsing
*Mar 1 01:07:47.971:Packet received -

CRCX 938694984 ds1-1/13@mc1 SGCP 1.1
M:recvonly
L:p:10,e:on,s:off, a:G.711u
```

```

R:hu
C:6

*Mar 1 01:07:47.971:SUCCESS:SGCP Header parsing was OK
*Mar 1 01:07:47.971:SUCCESS:Connection Mode parsing is OK
*Mar 1 01:07:47.971:SUCCESS:Packet period parsing is OK
*Mar 1 01:07:47.971:SUCCESS:Echo Cancellation parsing is OK
*Mar 1 01:07:47.971:SUCCESS:Silence Supression parsing is OK
*Mar 1 01:07:47.971:SUCCESS:CODEC strings parsing is OK
*Mar 1 01:07:47.971:SUCCESS:Local Connection option parsing is OK
*Mar 1 01:07:47.971:SUCCESS:Requested Event parsing is OK
*Mar 1 01:07:47.975:SUCCESS:Call ID string(6) parsing is OK
*Mar 1 01:07:47.975:SUCCESS:END of Parsing
*Mar 1 01:07:47.979:SUCCESS:Conn ID string building is OK
*Mar 1 01:07:47.979:SUCCESS:Building SGCP Parameter lines is OK
*Mar 1 01:07:47.979:SUCCESS:SGCP message building OK
*Mar 1 01:07:47.979:SUCCESS:END of building
*Mar 1 01:07:47.979:SGCP Packet sent --->
200 938694984 OK
I:6

v=0
c=IN IP4 5.0.0.1
m=audio 16538 RTP/AVP 0

<---

*Mar 1 01:07:48.188:Packet received -

MDCX 779665338 ds1-1/13@mc1 SGCP 1.1
I:6
M:recvonly
L:p:10,e:on,s:off,a:G.711u
R:hu
C:6

v=0
c=IN IP4 6.0.0.1
m=audio 16392 RTP/AVP 0

*Mar 1 01:07:48.188:SUCCESS:SGCP Header parsing was OK
*Mar 1 01:07:48.188:SUCCESS:Conn ID string(6) parsing is OK
*Mar 1 01:07:48.192:SUCCESS:Connection Mode parsing is OK
*Mar 1 01:07:48.192:SUCCESS:Packet period parsing is OK
*Mar 1 01:07:48.192:SUCCESS:Echo Cancellation parsing is OK
*Mar 1 01:07:48.192:SUCCESS:Silence Supression parsing is OK
*Mar 1 01:07:48.192:SUCCESS:CODEC strings parsing is OK
*Mar 1 01:07:48.192:SUCCESS:Local Connection option parsing is OK
*Mar 1 01:07:48.192:SUCCESS:Requested Event parsing is OK
*Mar 1 01:07:48.192:SUCCESS:Call ID string(6) parsing is OK
*Mar 1 01:07:48.192:SUCCESS:SDP Protocol version parsing OK
*Mar 1 01:07:48.192:SUCCESS:SDP Conn Data OK
*Mar 1 01:07:48.192:SUCCESS:END of Parsing
*Mar 1 01:07:48.200:SUCCESS:Conn ID string building is OK
*Mar 1 01:07:48.200:SUCCESS:Building SGCP Parameter lines is OK
*Mar 1 01:07:48.200:SUCCESS:SGCP message building OK
*Mar 1 01:07:48.200:SUCCESS:END of building
*Mar 1 01:07:48.200:SGCP Packet sent --->
200 779665338 OK
I:6

v=0
c=IN IP4 5.0.0.1
m=audio 16538 RTP/AVP 0

```

<---

\*Mar 1 01:07:53.674:Packet received -

MDCX 177780432 ds1-1/13@mc1 SGCP 1.1

I:6

M:sendrecv

X:519556004

L:p:10,e:on, s:off,a:G.711u

C:6

R:hu

S:hd

v=0

c=IN IP4 6.0.0.1

m=audio 16392 RTP/AVP 0

\*Mar 1 01:07:53.674:SUCCESS:SGCP Header parsing was OK

\*Mar 1 01:07:53.674:SUCCESS:Conn ID string(6) parsing is OK

\*Mar 1 01:07:53.674:SUCCESS:Connection Mode parsing is OK

\*Mar 1 01:07:53.674:SUCCESS:Request ID string(519556004) parsing is OK

\*Mar 1 01:07:53.678:SUCCESS:Packet period parsing is OK

\*Mar 1 01:07:53.678:SUCCESS:Echo Cancellation parsing is OK

\*Mar 1 01:07:53.678:SUCCESS:Silence Supression parsing is OK

\*Mar 1 01:07:53.678:SUCCESS:CODEC strings parsing is OK

\*Mar 1 01:07:53.678:SUCCESS:Local Connection option parsing is OK

\*Mar 1 01:07:53.678:SUCCESS:Call ID string(6) parsing is OK

\*Mar 1 01:07:53.678:SUCCESS:Requested Event parsing is OK

\*Mar 1 01:07:53.678:SUCCESS:Signal Requests parsing is OK

\*Mar 1 01:07:53.678:SUCCESS:SDP Protocol version parsing OK

\*Mar 1 01:07:53.678:SUCCESS:SDP Conn Data OK

\*Mar 1 01:07:53.678:SUCCESS:END of Parsing

\*Mar 1 01:07:53.682:SUCCESS:Conn ID string building is OK

\*Mar 1 01:07:53.682:SUCCESS:Building SGCP Parameter lines is OK

\*Mar 1 01:07:53.682:SUCCESS:SGCP message building OK

\*Mar 1 01:07:53.682:SUCCESS:END of building

\*Mar 1 01:07:53.682:SGCP Packet sent --->

200 177780432 OK

I:6

v=0

c=IN IP4 5.0.0.1

m=audio 16538 RTP/AVP 0

<---

\*Mar 1 01:09:02.401:SUCCESS:Request ID string building is OK

\*Mar 1 01:09:02.401:SUCCESS:Building SGCP Parameter lines is OK

\*Mar 1 01:09:02.401:SUCCESS:SGCP message building OK

\*Mar 1 01:09:02.401:SUCCESS:END of building

\*Mar 1 01:09:02.401:SGCP Packet sent --->

NTFY 24 ds1-1/13@mc1 SGCP 1.1

X:519556004

O:hu

<---

\*Mar 1 01:09:02.401:NTFY Packet sent successfully.

\*Mar 1 01:09:02.437:Packet received -

200 24

\*Mar 1 01:09:02.441:SUCCESS:SGCP Header parsing was OK

## debug sgcp packet

```

*Mar 1 01:09:02.441:SUCCESS:END of Parsing
*Mar 1 01:09:02.541:Packet received -

DLCX 865375036 ds1-1/13@mc1 SGCP 1.1
C:6
S:hu

*Mar 1 01:09:02.541:SUCCESS:SGCP Header parsing was OK
*Mar 1 01:09:02.541:SUCCESS:Call ID string(6) parsing is OK
*Mar 1 01:09:02.541:SUCCESS:Signal Requests parsing is OK
*Mar 1 01:09:02.541:SUCCESS:END of Parsing
*Mar 1 01:09:02.545:SUCCESS:SGCP message building OK
*Mar 1 01:09:02.545:SUCCESS:END of building
*Mar 1 01:09:02.545:SGCP Packet sent --->
250 865375036 OK

<---

*Mar 1 01:09:02.577:Packet received -

RQNT 254959796 ds1-1/13@mc1 SGCP 1.1
X:358258758
R:hd

*Mar 1 01:09:02.577:SUCCESS:SGCP Header parsing was OK
*Mar 1 01:09:02.577:SUCCESS:Request ID string(358258758) parsing is OK
*Mar 1 01:09:02.577:SUCCESS:Requested Event parsing is OK
*Mar 1 01:09:02.581:SUCCESS:END of Parsing
*Mar 1 01:09:02.581:SUCCESS:SGCP message building OK
*Mar 1 01:09:02.581:SUCCESS:END of building
*Mar 1 01:09:02.581:SGCP Packet sent --->
200 254959796 OK

```

## Related Commands

Command	Description
<b>debug rtpspi all</b>	Debugs all RTP SPI errors, sessions, and in/out functions.
<b>debug rtpspi errors</b>	Debugs RTP SPI errors.
<b>debug rtpspi inout</b>	Debugs RTP SPI in/out functions.
<b>debug rtpspi send-nse</b>	Triggers the RTP SPI to send a triple redundant NSE.
<b>debug sgcp errors</b>	Debugs SGCP errors.
<b>debug sgcp events</b>	Debugs SGCP events.
<b>debug vtsp send-nse</b>	Sends and debugs a triple redundant NSE from the DSP to a remote gateway.

# debug shared-line

To display debugging information about SIP shared lines, use the **debug shared-line** command in privileged EXEC mode. To disable debugging messages, use the **no** form of this command.

```
debug shared-line {all | errors | events | info}
```

```
no debug shared-line {all | errors | events | info}
```

## Syntax Description

<b>all</b>	Displays all shared-line debugging messages.
<b>errors</b>	Displays shared-line error messages.
<b>events</b>	Displays shared-line event messages.
<b>info</b>	Displays general information about shared lines.

## Command Modes

Privileged EXEC (#)

## Command History

Release	Modification
12.4(22)YB	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.
12.4(24)T	This command was integrated into Cisco IOS Release 12.4(24)T.

## Examples

The following example shows output from the **debug shared-line all** command:

```
Router# debug shared-line all

Aug 21 21:56:56.949: //Shared-Line/EVENT/shrl_validate_newcall_outgoing:Outgoing call
validation request from AFW for user = 20143, usrContainer = 4A7CFBDC
Aug 21 21:56:56.949: //Shared-Line/INFO/shrl_find_ccb_by_dn:Searching Shared-Line table
for dn '20143'
Aug 21 21:56:56.949: //Shared-Line/INFO/shrl_find_ccb_by_dn:Entry not found for dn
'20143'
Aug 21 21:56:56.949: //Shared-Line/INFO/shrl_find_ccb_by_demote_dn:Demoted dn: 20143
Aug 21 21:56:56.949: //Shared-Line/INFO/shrl_validate_newcall_outgoing>User '20143'
doesn't exist in Shared-Line table
Aug 21 21:56:56.957: //Shared-Line/EVENT/shrl_validate_newcall_incoming:Incoming call
validation request from AFW for user = 20141
Aug 21 21:56:56.957: //Shared-Line/INFO/shrl_find_ccb_by_dn:Searching Shared-Line table
for dn '20141'
Aug 21 21:56:56.957: //Shared-Line/INFO/shrl_find_ccb_by_dn:Entry found [ccb = 4742EAD4]
for dn '20141'
Aug 21 21:56:56.957: //Shared-Line/INFO/shrl_validate_newcall_incoming>User '20141'
found: ccb = 4742EAD4, mem_count = 2
Aug 21 21:56:56.957: //Shared-Line/EVENT/shrl_validate_newcall_incoming:Obtained call
instance inst: 0 for incoming call, incoming leg (peer_callid): 5399)
Aug 21 21:56:56.957: //Shared-Line/INFO/shrl_update_barge_calltype:Updating shared-line
call -1 with calltype = 1
```

```

.Aug 21 21:56:56.961: //Shared-Line/INFO/shrl_find_ccb_by_dn:Searching Shared-Line table
for dn '20141'
.Aug 21 21:56:56.961: //Shared-Line/INFO/shrl_find_ccb_by_dn:Entry found [ccb = 4742EAD4]
for dn '20141'
.Aug 21 21:56:56.961: //Shared-Line/INFO/shrl_find_ccb_by_dn:Searching Shared-Line table
for dn '20141'
.Aug 21 21:56:56.961: //Shared-Line/INFO/shrl_find_ccb_by_dn:Entry found [ccb = 4742EAD4]
for dn '20141'
.Aug 21 21:57:01.689: %IPPHONE-6-REG_ALARM: 24: Name=SEP00141C48E126 Load=8.0(5.0)
Last=Phone-Reg-Rej
.Aug 21 21:57:04.261: //Shared-Line/EVENT/shrl_app_event_notify_handler:Event notification
received: event = 9, callID = 5401, dn = 20141
.Aug 21 21:57:04.261: //Shared-Line/INFO/shrl_find_ccb_by_dn:Searching Shared-Line table
for dn '20141'
.Aug 21 21:57:04.261: //Shared-Line/INFO/shrl_find_ccb_by_dn:Entry found [ccb = 4742EAD4]
for dn '20141'
.Aug 21 21:57:04.261: //Shared-Line/EVENT/shrl_process_connect:called with state = 3,
callID = 5401, peer callID = 5399, dn = 20141, usrContainer = 4A7CACA4
.Aug 21 21:57:04.261: //Shared-Line/INFO/shrl_connect_upd_callinfo:Parsed To:
20141@15.6.0.2, to-tag: 2ed5b927-6ad6
.Aug 21 21:57:04.261: //Shared-Line/INFO/shrl_connect_upd_callinfo:Parsed Contact:
20141@15.6.0.2 for sipCallId: E8583537-6F0211DD-96A69BA1-1228BEFB@15.10.0.1
.Aug 21 21:57:04.261: //Shared-Line/EVENT/shrl_connect_upd_callinfo:Obtained call instance
inst: 0
.Aug 21 21:57:04.261: //Shared-Line/INFO/shrl_connect_upd_callinfo:CONNECT from shared
line for incoming shared-line call.
.Aug 21 21:57:04.261: //Shared-Line/INFO/shrl_find_peer_by_ipaddr:Trying to match peer for
member 20141@15.6.0.2
.Aug 21 21:57:04.261: //Shared-Line/INFO/shrl_find_peer_by_ipaddr:Matching peer [40002]
session target parsed = 15.6.0.2
.Aug 21 21:57:04.261: //Shared-Line/INFO/shrl_connect_upd_callinfo:Matching member found:
20141@15.6.0.2
.Aug 21 21:57:04.261: //Shared-Line/INFO/shrl_update_remote_name:Updating shared-line call
dialog info 5401

.Aug 21 21:57:04.261: //Shared-Line/INFO/shrl_process_connect:Updated callinfo for callid:
5401, member: '20141@15.6.0.2', peer-tag: 40002
.Aug 21 21:57:04.261: //Shared-Line/INFO/shrl_process_connect:Notify remote users about
CALL-CONNECT.
.Aug 21 21:57:04.261: //Shared-Line/EVENT/shrl_send_dialog_notify:Sending NOTIFY to remote
user: 20141@15.6.0.1
.Aug 21 21:57:04.261: //Shared-Line/INFO/shrl_send_dialog_notify:Sending NOTIFY to remote
user: 20141@15.6.0.1 about state 3 on incoming call from 20141@15.6.0.2 privacy OFF
.Aug 21 21:57:04.261: //Shared-Line/INFO/shrl_send_dialog_notify:Dialog msg: dir: 1,
orient: 2, local_tag: 2ed5b927-6ad6, remote_tag: 89DCF0-139B, local_uri: 20141@15.6.0.2,
remote_uri: 20143@15.10.0.1
.Aug 21 21:57:04.261: //Shared-Line/INFO/shrl_send_dialog_notify:Dialog notify sent
successfully
.Aug 21 21:57:04.261: //Shared-Line/INFO/shrl_process_connect:Shared-Line '20141':
Successfully sent notify for callid: 5401
.Aug 21 21:57:04.265: //Shared-Line/INFO/shrl_find_ccb_by_dn:Searching Shared-Line table
for dn '20141'
.Aug 21 21:57:04.265: //Shared-Line/INFO/shrl_find_ccb_by_dn:Entry found [ccb = 4742EAD4]
for dn '20141'
.Aug 21 21:57:04.265: //Shared-Line/INFO/shrl_find_ccb_by_dn:Searching Shared-Line table
for dn '20143'
.Aug 21 21:57:04.265: //Shared-Line/INFO/shrl_find_ccb_by_dn:Entry not found for dn
'20143'
.Aug 21 21:57:04.269: //Shared-Line/INFO/shrl_find_ccb_by_demote_dn:Demoted dn: 20143
.Aug 21 21:57:04.269: //Shared-Line/INFO/shrl_update_totag:Shared-Line not enabled for
'20143'
.Aug 21 21:57:04.269: //Shared-Line/EVENT/shrl_app_event_notify_handler:Event notification
received: event = 21, callID = 5401, dn = 20141

```

```
.Aug 21 21:57:04.269: //Shared-Line/INFO/shrl_find_ccb_by_dn:Searching Shared-Line table
for dn '20141'
.Aug 21 21:57:04.269: //Shared-Line/INFO/shrl_find_ccb_by_dn:Entry found [ccb = 4742EAD4]
for dn '20141'
.Aug 21 21:57:04.269: //Shared-Line/EVENT/shrl_process_callerid_update:called with state =
7, callID = 5401, peer callID = 5399, dn = 20141
.Aug 21 21:57:04.269: //Shared-Line/INFO/shrl_process_callerid_update:Updated callinfo for
callid: 5401, member: '20141@15.6.0.2', peer-tag: 40002
.Aug 21 21:57:04.269: //Shared-Line/EVENT/shrl_is_outbound:Check for shared line call type
callid 5401for user = 20141
.Aug 21 21:57:04.269: //Shared-Line/INFO/shrl_find_ccb_by_dn:Searching Shared-Line table
for dn '20141'
.Aug 21 21:57:04.269: //Shared-Line/INFO/shrl_find_ccb_by_dn:Entry found [ccb = 4742EAD4]
for dn '20141'
.Aug 21 21:57:04.269: //Shared-Line/EVENT/shrl_barge_type:Check for shared line call type
callid 5401for user = 20141
.Aug 21 21:57:04.269: //Shared-Line/INFO/shrl_find_ccb_by_dn:Searching Shared-Line table
for dn '20141'
.Aug 21 21:57:04.269: //Shared-Line/INFO/shrl_find_ccb_by_dn:Entry found [ccb = 4742EAD4]
for dn '20141'
.Aug 21 21:57:04.273: //Shared-Line/INFO/shrl_find_ccb_by_dn:Searching Shared-Line table
for dn '20141'
.Aug 21 21:57:04.273: //Shared-Line/INFO/shrl_find_ccb_by_dn:Entry found [ccb = 4742EAD4]
for dn '20141'
.Aug 21 21:57:04.281: //Shared-Line/EVENT/shrl_notify_done_handler:NOTIFY_DONE received
for subID: 5 respCode: 17
.Aug 21 21:57:04.281: //Shared-Line/INFO/shrl_find_ccb_by_subid:Search ccb for subid: 5
.Aug 21 21:57:04.281: //Shared-Line/INFO/shrl_find_ccb_by_subid:Found the entry ccb:
4742EAD4 member: 20141@15.6.0.1
.Aug 21 21:57:04.281: //Shared-Line/INFO/shrl_free_spi_respinfo:Free ASNL resp info for
subID = 5
```

**Related Commands**

Command	Description
<b>shared-line</b>	Creates a directory number to be shared by multiple SIP phones.
<b>show shared-line</b>	Displays information about active calls using SIP shared lines.

# debug smrp all

To display information about Simple Multicast Routing Protocol (SMRP) activity, use the **debug smrp all** privileged EXEC command. The **no** form of this command disables debugging output.

**debug smrp all**

**no debug smrp all**

## Syntax Description

This command has no arguments or keywords.

## Command History

10.0	This command was introduced.
12.2(13)T	This command is no longer supported in Cisco IOS Mainline releases or in Technology-based (T-train) releases. It might continue to appear in 12.2S-family releases.
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

Because the **debug smrp all** command displays all SMRP debugging output, it is processor intensive and should not be enabled when memory is scarce or in very high traffic situations.

For general debugging, use the **debug smrp all** command and turn off excessive transactions with the **no debug smrp transaction** command. This combination of commands will display various state changes and events without displaying every transaction packet. For debugging a specific feature such as a routing problem, use the **debug smrp route** and **debug smrp transaction** commands to learn if packets are sent and received and which specific routes are affected. The **show smrp traffic** EXEC command is highly recommended as a troubleshooting method because it displays the SMRP counters.

For examples of the type of output you may see, refer to each of the commands listed in the “Related Commands” section.

## Related Commands

Command	Description
<b>debug smrp group</b>	Displays information about SMRP group activity.
<b>debug smrp mcache</b>	Displays information about SMRP multicast fast-switching cache entries.
<b>debug smrp neighbor</b>	Displays information about SMRP neighbor activity.
<b>debug smrp port</b>	Displays information about SMRP port activity.
<b>debug smrp route</b>	Displays information about SMRP routing activity.
<b>debug smrp transaction</b>	Displays information about SMRP transactions.

# debug smrp group

To display information about SMRP group activity, use the **debug smrp group** privileged EXEC command. The **no** form of this command disables debugging output.

**debug smrp group**

**no debug smrp group**

## Syntax Description

This command has no arguments or keywords.

## Command History

10.0	This command was introduced.
12.2(13)T	This command is no longer supported in Cisco IOS Mainline releases or in Technology-based (T-train) releases. It might continue to appear in 12.2S-family releases.
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

The **debug smrp group** command displays information when a group is created or deleted and when a forwarding entry for a group is created, changed, or deleted. For more information, refer to the **show smrp group** command described in the *Cisco IOS AppleTalk and Novell IPX Command Reference*.

## Examples

The following is sample output from the **debug smrp group** command showing a port being created and deleted on group AT 20.34. (AT signifies that this is an AppleTalk network group.)

```
Router# debug smrp group

SMRP: Group AT 20.34, created on port 20.1 by 20.2
SMRP: Group AT 20.34, deleted on port 20.1
```

[Table 315](#) lists the messages that may be generated with the **debug smrp group** command concerning the forwarding table.

**Table 315** *debug smrp group Message Descriptions*

Messages	Descriptions
Group <address>, deleted on port <address>	Group entry was deleted from the group table for the specified port.
Group <address>, forward state changed from <i>state</i> to <i>state</i>	State of the group changed. States are join, forward, and leave.
Group <address>, deleted forward entry	Group was deleted from the forwarding table.
Group <address>, created on port <address> by <address>	Group entry was created in the table for the specified port.

**Table 315** *debug smrp group Message Descriptions (continued)*

Messages	Descriptions
Group <address>, added by <address> to the group	Secondary router has added this group to its group table.
Group <address>, discard join request from <address>, not responsible	Discard Join Group request if the router is not the primary router on the local connected network or if it is not the port parent of the route.
Group <address>, join request from <address>	Request to join the group was received.
Group <address>, forward is found	Forward entry for the group was found in the forwarding table.
Group <address>, forward state is already joining, ignored	Request to join the group is in progress, so the second request was discarded.
Group <address>, no forward found	Forward entry for the group was not found in the forwarding table.
Group <address>, join request discarded, fw discarded, fwd parent port not operational	Request to join the group was discarded because the parent port is not available.
Group <address>, created forward entry - parent <address> child <address>	Forward entry was created in the forwarding table for the parent and child address.
Group <address>, creator no longer up on <address>	Group creator has not been heard from for a specified time and is deemed no longer available.
Group <address>, pruning duplicate path on <address>	Duplicate path was removed. If we are forwarding and we are a child port, and our port parent address is not pointing to our own port address, we are in a duplicate path.
Group <address>, member no longer up on <address>	Group member has not been heard from for a specified time and is deemed no longer available.
Group <address>, no more child ports in forward entry	Forward entry for group no longer has any child ports. As a result, the forward entry is no longer necessary.

**Related Commands**

Command	Description
<b>debug sgbp dial-bids</b>	Displays large-scale dial-out negotiations between the primary NAS and alternate NASs.

# debug smrp mcache

To display information about SMRP multicast fast-switching cache entries, use the **debug smrp mcache** privileged EXEC command. The **no** form of this command disables debugging output.

**debug smrp mcache**

**no debug smrp mcache**

## Syntax Description

This command has no arguments or keywords.

## Command History

10.0	This command was introduced.
12.2(13)T	This command is no longer supported in Cisco IOS Mainline releases or in Technology-based (T-train) releases. It might continue to appear in 12.2S-family releases.
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 **show smrp mcache** EXEC command (described in the *Cisco IOS AppleTalk and Novell IPX Command Reference* to display the entries in the SMRP multicast cache, and use the **debug smrp mcache** command to learn whether the cache is being populated and invalidated.

## Examples

The following is sample output from the **debug smrp mcache** command. In this example, the cache is created and populated for group AT 11.124. (AT signifies that this is an AppleTalk network group.)

```
Router# debug smrp mcache

SMRP: Cache created
SMRP: Cache populated for group AT 11.124
      mac - 090007400b7c00000c1740d9
      net - 001fef7500000014ff020a0a0a
SMRP: Forward cache entry created for group AT 11.124
SMRP: Forward cache entry validated for group AT 11.124
SMRP: Forward cache entry invalidated for group AT 11.124
SMRP: Forward cache entry deleted for group AT 11.124
```

Table 316 lists all the messages that can be generated with the **debug smrp mcache** command concerning the multicast cache.

**Table 316** *debug smrp mcache Message Descriptions*

Messages	Descriptions
Cache populated for group <address>	SMRP packet was received on a parent port that has fast switching enabled. As a result, the cache was created and the MAC and network headers were stored for all child ports that have fast switching enabled. Use the <b>show smrp port appletalk EXEC</b> command with the optional interface type and number to display the switching path.
Cache memory allocated	Memory was allocated for the multicast cache.
Forward cache entry created/deleted for group <address>	Forward cache entry for the group was added to or deleted from the cache.
Forward cache entry validated for group <address>	Forward cache entry is validated and is now ready for fast switching.
Forward cache entry invalidated for group <address>	Cache entry is invalidated because some change (such as port was shut down) occurred to one of the ports.

#### Related Commands

Command	Description
<b>debug sgbp dial-bids</b>	Displays large-scale dial-out negotiations between the primary NAS and alternate NASs.

# debug smrp neighbor

To display information about SMRP neighbor activity, use the **debug smrp neighbor** privileged EXEC command. The **no** form of this command disables debugging output.

**debug smrp neighbor**

**no debug smrp neighbor**

## Syntax Description

This command has no arguments or keywords.

## Command History

10.0	This command was introduced.
12.2(13)T	This command is no longer supported in Cisco IOS Mainline releases or in Technology-based (T-train) releases. It might continue to appear in 12.2S-family releases.
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

The **debug smrp neighbor** command displays information when a neighbor operating state changes. A neighbor is an adjacent router. For more information, refer to the **show smrp neighbor** EXEC command described in the *Cisco IOS AppleTalk and Novell IPX Command Reference*.

## Examples

The following is sample output from the **debug smrp neighbor** command. In this example, the neighbor on port 30.02 has changed state from normal operation to secondary operation.

```
Router# debug smrp neighbor
```

```
SMRP: Neighbor 30.2, state changed from "normal op" to "secondary op"
```

[Table 317](#) lists all the messages that can be generated with the **debug smrp neighbor** command concerning the neighbor table.

**Table 317** *debug smrp neighbor Message Descriptions*

Messages	Descriptions
Neighbor <address>, state changed from <i>state</i> to <i>state</i>	State of the neighbor changed. States are primary operation, secondary operation, normal operation, primary negotiation, secondary negotiation, and down.
Neighbor <address>, neighbor added/deleted	Neighbor was added to or removed from the neighbor table.
SMRP neighbor up/down	Neighbor is available for service or unavailable.
Neighbor <address>, no longer up	Neighbor is unavailable because it has not been heard from for a specified duration.

■ `debug smrp neighbor`

---

**Related Commands**

<b>Command</b>	<b>Description</b>
<code>debug sgbp dial-bids</code>	Displays large-scale dial-out negotiations between the primary NAS and alternate NASs.

---

# debug smrp port

To display information about SMRP port activity, use the **debug smrp port** privileged EXEC command. The **no** form of this command disables debugging output.

**debug smrp port**

**no debug smrp port**

## Syntax Description

This command has no arguments or keywords.

## Command History

10.0	This command was introduced.
12.2(13)T	This command is no longer supported in Cisco IOS Mainline releases or in Technology-based (T-train) releases. It might continue to appear in 12.2S-family releases.
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

The **debug smrp port** command displays information when a port operating state changes. For more information, refer to the **show smrp port** command described in the *Cisco IOS AppleTalk and Novell IPX Command Reference*.

## Examples

The following is sample output from the **debug smrp port** command. In this example, port 30.1 has changed state from secondary negative to secondary operation to primary negative:

```
Router# debug smrp port
```

```
SMRP: Port 30.1, state changed from "secondary neg" to "secondary op"
SMRP: Port 30.1, secondary router changed from 0.0 to 30.1
SMRP: Port 30.1, state changed from "secondary op" to "primary neg"
```

[Table 318](#) lists all the messages that can be generated with the **debug smrp port** command concerning the port table.

**Table 318** *debug smrp port Message Descriptions*

Messages	Descriptions
Port <address>, port created/deleted	Port entry was added to or removed from the port table.
Port <address>, line protocol changed to <i>state</i>	Line protocol for the port is up or down.

**Table 318** *debug smrp port Message Descriptions (continued)*

<b>Messages</b>	<b>Descriptions</b>
Port <address>, state changed from <i>state</i> to <i>state</i>	State of the port changed. States are primary operation, secondary operation, normal operation, primary negotiation, secondary negotiation, and down.
Port <address>, primary/secondary router changed from <address> to <address>	Primary or secondary port address of the router changed.

**Related Commands**

<b>Command</b>	<b>Description</b>
<b>debug sgbp dial-bids</b>	Displays large-scale dial-out negotiations between the primary NAS and alternate NASs.

# debug smrp route

To display information about SMRP routing activity, use the **debug smrp route** privileged EXEC command. The **no** form of this command disables debugging output.

**debug smrp route**

**no debug smrp route**

## Syntax Description

This command has no arguments or keywords.

## Command History

10.0	This command was introduced.
12.2(13)T	This command is no longer supported in Cisco IOS Mainline releases or in Technology-based (T-train) releases. It might continue to appear in 12.2S-family releases.
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

For more information, refer to the **show smrp route** EXEC command described in the *Cisco IOS AppleTalk and Novell IPX Command Reference*.

## Examples

The following is sample output from the **debug smrp route** command. In this example, poison notification is received from port 30.2. Poison notification is the receipt of a poisoned route on a nonparent port.

```
Router# debug smrp route
```

```
SMRP: Route AT 20-20, poison notification from 30.2
```

```
SMRP: Route AT 30-30, poison notification from 30.2
```

[Table 319](#) lists all the messages that can be generated with the **debug smrp route** command concerning the routing table. In [Table 319](#), the term *route* does not refer to an address but rather to a network range.

**Table 319** *debug smrp route Message Descriptions*

Messages	Descriptions
Route address, deleted/created as local network	Route entry was removed from or added to the routing table.
Route address, from address has invalid distance value	Route entry from the specified address has an incorrect distance value and was ignored.
Route address, unknown route poisoned by address ignored	Route entry received from the specified address is bad and was ignored.
Route address, created via address - hop number tunnel number	New route entry added to the routing table with the specified number of hops and tunnels.

**Table 319** *debug smrp route Message Descriptions (continued)*

<b>Messages</b>	<b>Descriptions</b>
Route address, from address - overlaps existing route	Route entry received from the specified address overlaps an existing route and was ignored.
Route address, poisoned by address	Route entry has been poisoned by neighbor. Poisoned routes have distance of 255.
Route address, poison notification from address	Poisoned route is received from a nonparent port.
Route address, worsened by parent address	Distance to the route has worsened (become higher), received from the parent neighbor.
Route address, improved via address - number -> number hop, number -> number tunnel	Distance to the route has improved (become lower), received from a neighbor.
Route address, switched to address - higher address than address	Tie condition exists, and because this router had the highest network address, it was used to forward the packet.
Route address, parent port changed address -> address	Parent port address change occurred. The parent port address of a physical network segment determines which router should handle Join Group and Leave Group requests.
SMRP bad distance vector	Packet has an invalid distance vector and was ignored.
Route address, has been poisoned	Route has been poisoned. Poisoned routes are purged from the routing table after a specified time.

**Related Commands**

<b>Command</b>	<b>Description</b>
<b>debug sgbp dial-bids</b>	Displays large-scale dial-out negotiations between the primary NAS and alternate NASs.

# debug smrp transaction

To display information about SMRP transactions, use the **debug smrp transaction** privileged EXEC command. The **no** form of this command disables debugging output.

**debug smrp transaction**

**no debug smrp transaction**

## Syntax Description

This command has no arguments or keywords.

## Examples

The following is sample output from the **debug smrp transaction** command. In this example, a secondary node request is sent out to all routers on port 30.1.

```
Router# debug smrp transaction
```

```
SMRP: Transaction for port 30.1, secondary node request (seq 8435) sent to all routers
SMRP: Transaction for port 30.1, secondary node request (seq 8435) sent to all routers
SMRP: Transaction for port 30.1, secondary node request (seq 8435) sent to all routers
SMRP: Transaction for port 30.1, secondary node request (seq 8435) sent to all routers
```

[Table 320](#) lists all the messages that can be generated with the **debug smrp route** command.

**Table 320** *debug smrp Transaction Message Descriptions*

Messages	Descriptions
Transaction for port address, packet-type command-type (grp/sec number) sent to/received from address	Port message concerning a packet or command was sent to or received from the specified address.
Transaction for group address on port address, (seq number) sent to/received from address	Group message for a specified port was sent to or received from the specified address.
Unrecognized transaction for port address	Unrecognized message was received and ignored by the port.
Discarded incomplete request	Incomplete message was received and ignored.
Response in wrong state in HandleRequest	Message was received with the wrong state and was ignored.
SMRP bad packet type	SMRP packet was received with a bad packet type and was ignored.
Packet discarded, Bad Port ID	Packet was received with a bad port ID and was ignored.
Packet discarded, Check Packet failed	Packet was received with a failed check packet and was ignored.

## Related Commands

Command	Description
<b>debug sgbp dial-bids</b>	Displays large-scale dial-out negotiations between the primary NAS and alternate NASs.

# debug snasw dlc

To display frame information entering and leaving the Systems Network Architecture (SNA) switch in real time to the console, use the **debug snasw dlc** command in privileged EXEC mode.

## debug snasw dlc detail

### Syntax Description

<b>detail</b>	Indicates that in addition to a one-line description of the frame being displayed, an entire hexadecimal dump of the frame will follow.
---------------	---

### Defaults

By default, a one-line description of the frame is displayed.

### Command Modes

Privileged EXEC

### Command History

Release	Modification
12.0(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.

### Usage Guidelines



#### Caution

The **debug snasw dlc** command displays the same trace information available via the **snasw dlctrace** command. The **snasw dlctrace** command is the preferred method for gathering this trace information because it is written to a capture buffer instead of directly to the console. The **debug snasw dlc** command should only be used when it is certain that the output will not cause excessive data to be output to the console.

### Examples

The following shows sample output from the **debug snasw dlc** command:

```
Router# debug snasw dlc
```

```
Sequence
Number      Link          Size of ISR/
           SNA BTU HPR  Description of frame
343  MVSD      In  sz:134  ISR fmh5 DLUR Rq ActPU NETA.APPNRA29
344  MVSD      Out sz:12   ISR +Rsp IPM      slctd nws:0008
345  @I000002 Out sz:18   ISR Rq ActPU
346  MVSD      Out sz:273  ISR fmh5 TOPOLOGY UPDATE
347  @I000002 In  sz:9     ISR +Rsp Data
348  @I000002 In  sz:12   ISR +Rsp IPM      slctd nws:0002
349  @I000002 In  sz:29   ISR +Rsp ActPU
```

```

350  MVSD      Out sz:115  ISR fmh5 DLUR +Rsp ActPU
351  MVSD      In  sz:12   ISR +Rsp IPM      slctd nws:0007
352  MVSD      In  sz:88   ISR fmh5 DLUR Rq ActLU NETA.MARTLU1
353  MVSD      Out sz:108  ISR fmh5 REGISTER
354  @I000002 Out sz:27   ISR Rq ActLU NETA.MARTLU1

```

**Related Commands**

Command	Description
<b>snasw dlcfilter</b>	Filters frames traced by the <b>snasw dlctrace</b> or <b>debug snasw dlc</b> command.
<b>snasw dlctrace</b>	Captures trace frames entering and leaving the SNA Switching Services feature.

# debug snasw ips

To display internal signal information between the Systems Network Architecture (SNA) switch and the console in real time, use the **debug snasw ips** command in privileged EXEC mode.

## debug snasw dlc

**Syntax Description** This command has no arguments or keywords.

**Defaults** By default, a one-line description of the interprocess signal is displayed.

**Command Modes** Privileged EXEC

Command History	Release	Modification
	12.0(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.

## Usage Guidelines



### Caution

The **debug snasw ips** command displays the same trace information available via the **snasw ipstrace** command. Output from this **debug** command can be large. The **snasw ipstrace** command is the preferred method for gathering this trace information because it is written to a capture buffer instead of directly to the console. The **debug snasw ips** command should only be used when it is certain that the output will not cause excessive data to be output to the console. The **debug snasw dlc** command displays the same trace information available via the **snasw dlctrace** command.

## Examples

The following is an example of the **debug snasw ips** command output:

```
Router# debug snasw ips
```

```
Sequence
Number      Signal Name      Sending      Receiving
              Process          Process      Queue

11257 : DEALLOCATE_RCB : --(0) -> RM(2130000) Q 4
11258 : RCB_DEALLOCATED : RM(2130000) -> PS(22E0000) Q 2
11259 : RCB_DEALLOCATED : --(0) -> PS(22E0000) Q 2
11260 : VERB_SIGNAL : PS(22E0000) -> DR(20F0000) Q 2
11261 : FREE_SESSION : --(0) -> RM(2130000) Q 2
11262 : BRACKET_FREED : RM(2130000) -> HS(22FB0001) Q 2
11263 : BRACKET_FREED : --(0) -> HS(22FB0001) Q 2
11264 : VERB_SIGNAL : --(0) -> DR(20F0000) Q 2
```

```
11265 : DLC_MU : DLC(2340000) -> PC(22DD0001) Q 2
11266 : DLC_MU : --(0) -> PC(22DD0001) Q 2
```

---

**Related Commands**

<b>Command</b>	<b>Description</b>
<b>snasw ipstrace</b>	Captures interprocess signal information between Switching Services components.

---

# debug snmp bulkstat

To enable debugging messages for the Simple Network Management Protocol (SNMP) bulk statistics, use the **debug snmp bulkstat** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug snmp bulkstat**

**no debug snmp bulkstat**

**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.
	12.3(2)T	This command was integrated into Cisco IOS Release 12.3(2)T.
	12.2(25)S	This command was integrated into Cisco IOS Release 12.2(25)S.
	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 is intended primarily for Cisco support personnel. Debugging output for the Periodic MIB Data Collection and Transfer Mechanism (Bulk Statistics feature) includes messages for data collection, local file generation, and transfer attempts.

**Examples** In the following example, debugging command output is enabled for the Periodic MIB Data Collection and Transfer Mechanism (Bulk Statistics feature). Note that the references to a VFile indicate a local bulk statistics file, usually followed by the filename. The filename uses the format *specified-filename\_device-name\_date\_time-stamp*.

```
Router# debug snmp

00:17:38:BULKSTAT-DC:Poll timer fired for ifmib
00:17:38:BULKSTAT-DC:In pollDataGroup
00:17:38:BULKSTAT-DC:creating new file
vfile:IfMIB_objects_ios108_030307_101119739
00:17:38:BULKSTAT-DC:Too small state buffer for ifmib
102
00:17:38:BULKSTAT-DC:Increased buffer state to 1024
00:17:38:BULKSTAT-DC:Interface type data group
00:17:38:BULKSTAT-DC:polling done
00:18:38:BULKSTAT-DC:Poll timer fired for ifmib
```

```
00:18:38:BULKSTAT-DC:In pollDataGroup
00:18:38:BULKSTAT-DC:Interface type data group
00:18:38:BULKSTAT-DC:polling done
00:19:26:
BULKSTAT-DC:Collection timer fired for IfMIB_objects
00:19:26:BULKSTAT-TP:Transfer request for
vfile:IfMIB_objects_ios108_030307_101119739
00:19:30:BULKSTAT-TP:written vfile
IfMIB_objects_ios108_030307_101119739
00:19:30:BULKSTAT-TP:retained vfile
vfile:IfMIB_objects_ios108_030307_101119739
00:19:38:BULKSTAT-DC:Poll timer fired for ifmib
00:19:38:BULKSTAT-DC:In pollDataGroup
00:19:38:BULKSTAT-DC:creating new file
vfile:IfMIB_objects_ios108_030307_101319739
00:19:38:BULKSTAT-DC:Interface type data group
00:19:38:BULKSTAT-DC:polling done
00:20:38:BULKSTAT-DC:Poll timer fired for ifmib
00:20:38:BULKSTAT-DC:In pollDataGroup
00:20:38:BULKSTAT-DC:Interface type data group
00:20:38:BULKSTAT-DC:polling done
00:21:38:BULKSTAT-DC:Poll timer fired for ifmib
00:21:38:BULKSTAT-DC:In pollDataGroup
00:21:38:BULKSTAT-DC:Interface type data group
00:21:38:BULKSTAT-DC:polling done
00:22:26:
BULKSTAT-DC:Collection timer fired for IfMIB_objects
00:22:26:BULKSTAT-TP:Transfer request for
vfile:IfMIB_objects_ios108_030307_101319739
00:22:26:BULKSTAT-TP:written vfile
IfMIB_objects_ios108_030307_101319739
00:22:26:BULKSTAT-TP:retained vfile
vfile:IfMIB_objects_ios108_030307_101319739
00:22:38:BULKSTAT-DC:Poll timer fired for ifmib
00:22:38:BULKSTAT-DC:In pollDataGroup
00:22:38:BULKSTAT-DC:creating new file
vfile:IfMIB_objects_ios108_030307_101619739
00:22:38:BULKSTAT-DC:Interface type data group
00:22:38:BULKSTAT-DC:polling done
00:23:38:BULKSTAT-DC:Poll timer fired for ifmib
00:23:38:BULKSTAT-DC:In pollDataGroup
00:23:38:BULKSTAT-DC:Interface type data group
00:23:38:BULKSTAT-DC:polling done
00:24:38:BULKSTAT-DC:Poll timer fired for ifmib
00:24:38:BULKSTAT-DC:In pollDataGroup
00:24:38:BULKSTAT-DC:Interface type data group
00:24:38:BULKSTAT-DC:polling done
00:25:26:
BULKSTAT-DC:Collection timer fired for IfMIB_objects
00:25:26:BULKSTAT-TP:Transfer request for
vfile:IfMIB_objects_ios108_030307_101619739
00:25:26:BULKSTAT-TP:written vfile
IfMIB_objects_ios108_030307_101619739
00:25:26:BULKSTAT-TP:retained vfile
vfile:IfMIB_objects_ios108_030307_101619739
00:25:38:BULKSTAT-DC:Poll timer fired for ifmib
00:25:38:BULKSTAT-DC:In pollDataGroup
00:25:38:BULKSTAT-DC:creating new file
vfile:IfMIB_objects_ios108_030307_101919739
00:25:38:BULKSTAT-DC:Interface type data group
00:25:38:BULKSTAT-DC:polling done
00:26:38:BULKSTAT-DC:Poll timer fired for ifmib
```

## ■ debug snmp bulkstat

```
00:26:38:BULKSTAT-DC:In pollDataGroup
00:26:38:BULKSTAT-DC:Interface type data group
00:26:38:BULKSTAT-DC:polling done
```

---

**Related Commands**

<b>Command</b>	<b>Description</b>
<b>show snmp mib bulkstat transfer</b>	Displays the transfer status of files generated by the Periodic MIB Data Collection and Transfer Mechanism.
<b>snmp mib bulkstat transfer</b>	Names a bulk statistics transfer configuration and enters Bulk Statistics Transfer configuration mode.

# debug snmp detail

To display Simple Network Management Protocol (SNMP) debug messages, use the **debug snmp detail** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug snmp detail**

**no debug snmp detail**

**Syntax Description** This command has no arguments or keywords.

**Command Default** SNMP debug messages are not displayed.

**Command Modes** Privileged EXEC (#)

Command History	Release	Modification
	12.4(20)T	This command was introduced.
	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.
	12.2(50)SY	This command was integrated into Cisco IOS Release 12.2(50)SY.

**Usage Guidelines** Before running the **debug snmp detail** command, connect the device to the Network Management System (NMS). The command output displays debug messages for errors occurred during SNMP operations. The debug messages help in identifying and debugging errors.

**Examples** The following is sample output from the **debug snmp detail** command:

```
Router# debug snmp detail

SNMP Detail Debugs debugging is on
process_mgmt_req_int: UDP packet being de-queued
findContextInfo: Authentication failure, bad community string
SrDoSnmp: Bad Community name.

process_mgmt_req_int: UDP packet being de-queued
SrParseV3SnmpMessage: No matching Engine ID.
SrParseV3SnmpMessage: Failed.
SrDoSnmp: authentication failure, Unknown Engine ID

process_mgmt_req_int: UDP packet being de-queued
ParseSequence, Unexpected type: 4
SrParseV3SnmpMessage: ParseSequence:
SrParseV3SnmpMessage: Failed.
SrDoSnmp: authentication failure, Unsupported security modelQ:
```

---

**Related Commands**

<b>Command</b>	<b>Description</b>
<b>debug snmp packet</b>	Displays information about every SNMP packet sent or received by the router.

---

# debug snmp mib nhrp

To display messages about Simple Network Management Protocol (SNMP) Next Hop Resolution Protocol (NHRP) MIB, use the **debug snmp mib nhrp** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

```
debug snmp mib nhrp {error | events | internal | notif [detail]}
```

```
no debug snmp mib nhrp {error | events | internal | notif [detail]}
```

## Syntax Description

<b>error</b>	Displays messages about SNMP NHRP MIB error events, including error information about packet processing or MIB special events.
<b>events</b>	Displays messages about SNMP NHRP MIB events, from the NHRP MIB tree data-structures and SNMP query-related events.
<b>internal</b>	Displays messages about SNMP NHRP MIB engineering events.
<b>notif</b>	Displays debug messages related to SNMP NHRP MIB notification events.
<b>detail</b>	(Optional) Displays detailed messages related to SNMP NHRP MIB notification events.

## Command Modes

Privileged EXEC (#)

## Command History

Release	Modification
12.4(20)T	This command was introduced.
15.0(1)M	This command was modified. The <b>notif</b> and <b>detail</b> keywords were added.

## Usage Guidelines

The **debug snmp mib nhrp internal** command can generate many output messages. Due to the increased command processing and its effect on system usage, the use of this command is not advisable under normal circumstances.

## Examples

The following is sample output from the **debug snmp mib nhrp notif** command:

```
*May 10 12:52:01.245: NHRP_SNMP-NOTIF[1488]: Retrieved values from instrumentation
*May 10 12:52:01.245: NHRP_SNMP-NOTIF[1646]: Varbind list created
*May 10 12:52:01.245: NHRP_SNMP-NOTIF[1665]: NHRP trap queued: cneNotifNextHopRegClientUp
```

The following is sample output from the **debug snmp mib nhrp notif detail** command:

```
*May 10 12:52:44.461: NHRP_SNMP-NOTIF[695]: Address parameters'
extraction for local and remote endpoints successful
*May 10 12:52:44.461: NHRP_SNMP-NOTIF[1488]: Retrieved values from instrumentation
*May 10 12:52:44.461: NHRP_SNMP-NOTIF[1589]: Instance OIDs populated
*May 10 12:52:44.461: NHRP_SNMP-NOTIF[1608]: Value types and values populated
*May 10 12:52:44.461: NHRP_SNMP-NOTIF[1625]: Varbind created for
nhrpServerInternetNetworkAddrType
*May 10 12:52:44.461: NHRP_SNMP-NOTIF[1643]: Varbind created for
nhrpServerInternetNetworkAddr
*May 10 12:52:44.461: NHRP_SNMP-NOTIF[1643]: Varbind created for nhrpServerNbmaAddrType
```

```
*May 10 12:52:44.461: NHRP_SNMP-NOTIF[1643]: Varbind created for nhrpServerNbmaAddr
*May 10 12:52:44.461: NHRP_SNMP-NOTIF[1643]: Varbind created for nhrpServerNbmaSubaddr
*May 10 12:52:44.461: NHRP_SNMP-NOTIF[1643]: Varbind created for
nhrpServerNhcInternetNetworkAddrType
*May 10 12:52:44.461: NHRP_SNMP-NOTIF[1643]: Varbind created for
nhrpServerNhcInternetNetworkAddr
*May 10 12:52:44.461: NHRP_SNMP-NOTIF[1643]: Varbind created for nhrpServerNhcNbmaAddrType
*May 10 12:52:44.461: NHRP_SNMP-NOTIF[1643]: Varbind created for nhrpServerNhcNbmaAddr
*May 10 12:52:44.461: NHRP_SNMP-NOTIF[1643]: Varbind created for nhrpServerNhcNbmaSubaddr
*May 10 12:52:44.461: NHRP_SNMP-NOTIF[1643]: Varbind created for nhrpServerNhcPrefixLength
*May 10 12:52:44.461: NHRP_SNMP-NOTIF[1643]: Varbind created for nhrpServerNhcInUse
*May 10 12:52:44.461: NHRP_SNMP-NOTIF[1643]: Varbind created for nhrpServerCacheUniqueness
*May 10 12:52:44.461: NHRP_SNMP-NOTIF[1646]: Varbind list created
*May 10 12:52:44.461: NHRP_SNMP-NOTIF[1665]: NHRP trap queued: cneNotifNextHopRegClientUp
```

The following is sample output from the **debug snmp mib nhrp events** command:

```
Router# debug snmp mib nhrp events

*Apr 10 13:34:46.175: NHRP_SNMP-EVE[2097]: In Get nhrpClientEntry for VRFID [0]
ClientIndex [0] NHS [0] Req [1]
*Apr 10 13:34:46.175: NHRP_SNMP-EVE[2148]: In here as expected.
*Apr 10 13:34:46.175: NHRP_SNMP-EVE[1050]: In Extract Client Entry Info
*Apr 10 13:34:46.223: NHRP_SNMP-EVE[2097]: In Get nhrpClientEntry for VRFID [0]
ClientIndex [2] NHS [0] Req [1]
*Apr 10 13:34:46.223: NHRP_SNMP-EVE[2140]: Could not find the Node
*Apr 10 13:34:46.223: NHRP_SNMP-EVE[2097]: In Get nhrpClientEntry for VRFID [0]
ClientIndex [0] NHS [0] Req [1]
*Apr 10 13:34:46.223: NHRP_SNMP-EVE[2148]: In here as expected.
*Apr 10 13:34:46.223: NHRP_SNMP-EVE[1050]: In Extract Client Entry Info
```

The following is sample output from the **debug snmp mib nhrp internal** command:

```
Router# debug snmp mib nhrp internal

*Apr 10 13:36:33.267: NHRP_SNMP-INTR[2089]: In nhrpClientEntry
*Apr 10 13:36:33.323: NHRP_SNMP-INTR[2089]: In nhrpClientEntry
*Apr 10 13:36:33.323: NHRP_SNMP-INTR[2089]: In nhrpClientEntry
```

[Table 321](#) describes the significant fields shown in the displays.

**Table 321 debug snmp mib nhrp Field Descriptions**

Field	Description
NHRP_SNMP-ERR[ ]	Indicates output from the <b>debug snmp mib nhrp error</b> command.
NHRP_SNMP-EVE[2097 ]	Indicates output from the <b>debug snmp mib nhrp events</b> command.
NHRP_SNMP-INTR[2089 ]	Indicates output from the <b>debug snmp mib nhrp internal</b> command.
NHRP_SNMP-NOTIF[1488]	Indicates output from the <b>debug snmp mib nhrp notif</b> command.

#### Related Commands

Command	Description
<b>show snmp mib nhrp status</b>	Indicates the status of the NHRP MIB and whether the NHRP MIB is enabled or disabled.

# debug snmp overhead

To display the list of Simple Network Management Protocol (SNMP) MIBs that take more than the threshold time to perform an SNMP get or get-next operation, use the **debug snmp overhead** command in privileged EXEC mode. To disable debugging, use the **no** form of this command.

**debug snmp overhead**

**no debug snmp overhead**

## Syntax Description

This command has no arguments or keywords.

## Command Default

SNMP debug messages are not displayed.

## Command Modes

Privileged EXEC (#)

## Command History

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

## Examples

The following is sample output from the **debug snmp overhead** command:

```
Router# debug snmp overhead

SNMP overhead debugging is on

*Nov 11 16:35:02.579 PDT: Process exceeds 1000ms threshold (200ms IOS quantum)
*Nov 11 16:35:02.579 PDT: GETNEXT of ciscoFlashFileEntry.2.1.1.1--result
ciscoFlashFileEntry.2.1.1.2
```

[Table 322](#) describes the significant fields shown in the display.

**Table 322** *debug snmp overhead Field Descriptions*

Field	Description
Process exceeds 1000ms threshold	Processing time for the SNMP get-next operation is more than 1000 milliseconds.
200ms IOS quantum	Threshold time in milliseconds.
GETNEXT of ciscoFlashFileEntry.2.1.1.1	The OID ciscoFlashFileEntry.2.1.1.1 is queried using the get-next operation.
result ciscoFlashFileEntry.2.1.1.2	The result of the get-next operation is ciscoFlashFileEntry.2.1.1.2, which is the next value of the OID being queried.

# debug snmp packet

To display information about every Simple Network Management Protocol (SNMP) packet sent or received by the router, use the **debug snmp packet** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug snmp packet**

**no debug snmp packet**

**Syntax Description** This command has no arguments or keywords.

**Command Default** The command is disabled by default.

**Command Modes** Privileged EXEC (#)

## Command History

Release	Modification
12.0(24)S	This command was introduced.
12.3(2)T	This command was integrated into Cisco IOS Release 12.3(2)T.
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 was integrated into Cisco IOS Release 12.2(33)SB.
Cisco IOS XE Release 2.5	This command was implemented on Cisco ASR 1000 series routers.

## Examples

The following is sample output from the **debug snmp packet** command. In this example, the router receives a get-next request from the host at 192.10.2.10 and responds with the requested information.

```
Router# debug snmp packet

SNMP: Packet received via UDP from 192.10.2.10 on Ethernet0
SNMP: Get-next request, reqid 23584, errstat 0, erridx 0
  sysUpTime = NULL TYPE/VALUE
  system.1 = NULL TYPE/VALUE
  system.6 = NULL TYPE/VALUE
SNMP: Response, reqid 23584, errstat 0, erridx 0
  sysUpTime.0 = 2217027
  system.1.0 = Cisco Internetwork Operating System Software
  system.6.0 =
SNMP: Packet sent via UDP to 192.10.2.10
```

Based on the kind of packet sent or received, the output may vary. For get-bulk requests, a line similar to the following is displayed:

```
SNMP: Get-bulk request, reqid 23584, nonrptr 10, maxreps 20
```

For traps, a line similar to the following is displayed:

SNMP: V1 Trap, ent 1.3.6.1.4.1.9.1.13, gentrap 3, spectrap 0

Table 323 describes the significant fields shown in the display.

**Table 323** *debug snmp packet Field Descriptions*

Field	Description
Get-next request	<p>Indicates what type of SNMP protocol data unit (PDU) the packet is. Possible types are as follows:</p> <ul style="list-style-type: none"> <li>• Get request</li> <li>• Get-next request</li> <li>• Response</li> <li>• Set request</li> <li>• V1 Trap</li> <li>• Get-bulk request</li> <li>• Inform request</li> <li>• V2 Trap</li> </ul> <p>Depending on the type of PDU, the rest of this line displays different fields. The indented lines following this line list the MIB object names and corresponding values.</p>
reqid	Request identification number. This number is used by the SNMP manager to match responses with requests.
errstat	Error status. All PDU types other than response will have an errstat of 0. If the agent encounters an error while processing the request, it will set errstat in the response PDU to indicate the type of error.
erridx	Error index. This value will always be 0 in all PDUs other than responses. If the agent encounters an error, the erridx will be set to indicate which varbind in the request caused the error. For example, if the agent had an error on the second varbind in the request PDU, the response PDU will have an erridx equal to 2.
nonrptr	Nonrepeater value. This value and the maximum repetition value are used to determine how many varbinds are returned. Refer to RFC 1905 for details.
maxreps	Maximum repetition value. This value and the nonrepeater value are used to determine how many varbinds are returned. Refer to RFC 1905 for details.
ent	Enterprise object identifier. Refer to RFC 1215 for details.
gentrap	Generic trap value. Refer to RFC 1215 for details.
spectrap	Specific trap value. Refer to RFC 1215 for details.

# debug snmp requests

To display information about every Simple Network Management Protocol (SNMP) request made by the SNMP manager, use the **debug snmp requests** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug snmp requests**

**no debug snmp requests**

## Syntax Description

This command has no arguments or keywords.

## Command Modes

Privileged EXEC

## Examples

The following is sample output from the **debug snmp requests** command:

```
Router# debug snmp requests

SNMP Manager API: request
  dest: 171.69.58.33.161, community: public
  retries: 3, timeout: 30, mult: 2, use session rtt
  userdata: 0x0
```

[Table 324](#) describes the significant fields shown in the display.

**Table 324** *debug snmp requests* Field Descriptions

Field	Description
SNMP Manager API	Indicates that the router sent an SNMP request.
dest	Destination of the request.
community	Community string sent with the request.
retries	Number of times the request has been re-sent.
timeout	Request timeout, or how long the router will wait before resending the request.
mult	Timeout multiplier. The timeout for a re-sent request will be equal to the previous timeout multiplied by the timeout multiplier.
use session rtt	Indicates that the average round-trip time of the session should be used in calculating the timeout value.
userdata	Internal Cisco IOS software data.

# debug snmp sync

To debug Simple Network Management Protocol (SNMP) synchronization and faults in synchronization, use the **debug snmp sync** command in privileged EXEC mode. To disable the display of debugging output, use the **no** form of this command.

**debug snmp sync**

**no debug snmp sync**

**Syntax Description** This command has no arguments or keywords.

**Defaults** Disabled.

**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(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.
	12.2SX	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** The **debug snmp sync** command can be used to debug SNMP synchronization and faults in synchronization. The standby Route Processor (RP) may sometimes reset as a result of synchronization faults. If the fault occurs when SNMP activities such as SNMP sets are in progress, enter the **debug snmp sync** command to identify whether a synchronization fault caused the reset.

SNMP synchronizations (dynamic and bulk) are performed only if the router is configured to be in stateful switchover (SSO) mode.

**Examples** The following example enables debugging of SNMP synchronization activity:

```
Router# debug snmp sync
```

Related Commands	Command	Description
	<b>debug snmp packets</b>	Displays information about every SNMP packet sent or received by the networking device.
	<b>mode</b>	Configures the redundancy mode of operation.

# debug snmp tunnel-mib

To enable the debugging for configuring the IP Tunnel Management Information Base (MIB) through Simple Network Management Protocol (SNMP), use the **debug snmp tunnel-mib** command in privileged EXEC mode. To disable debugging, use the **no** form of this command.

**debug snmp tunnel-mib**

**no debug snmp tunnel-mib**

**Syntax Description** This command has no arguments or keywords.

**Command Modes** Privileged EXEC (#)

Command History	Release	Modification
	12.2(33)SRB	This command was introduced.
	12.4(15)T	This command was integrated into Cisco IOS Release 12.4(15)T.
	12.2(33)SB1	This command was integrated into Cisco IOS Release 12.2(33)SB1.
	12.2(44)SG	This command was integrated into Cisco IOS Release 12.2(44)SG.
	Cisco IOS Release XE 2.1	This command was integrated into Cisco IOS Release XE 2.1.

**Usage Guidelines** Use the **debug snmp tunnel-mib** command to verify whether a tunnel is created or deleted.

**Examples** The following is sample output from the **debug snmp tunnel-mib** command. The output shows that a tunnel is created through SNMP.

```
Router# debug snmp tunnel-mib

SNMP TUNNEL-MIB debugging is on

k_tunnelInetConfigEntry_get: Entering
k_tunnelInetConfigEntry_get: Exact search
tim_client_tunnel_endpoint_data_get: Entering
tim_client_tunnel_endpoint_data_get: Exact search
tim_client_tunnel_endpoint_data_get: No element found
k_tunnelInetConfigEntry_get: Client service failed
k_tunnelInetConfigEntry_test: Entering
k_tunnelInetConfigEntry_test: Completed
k_tunnelInetConfigEntry_set: Entering
tim_client_tunnel_endpoint_data_get: Entering
tim_client_tunnel_endpoint_data_get: Exact search
tim_client_tunnel_endpoint_data_get: No element found
k_tunnelInetConfigEntry_set: Calling tunnel create
tim_client_tunnel_create: Entering
tim_client_tunnel_create: Completed
```