

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

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

Defaults

This command is disabled by default.

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.

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
show redundancy debug-log	Displays up to 256 debug entries.

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.

Command Modes Privileged EXEC

Defaults Disabled

Command History	Release	Modification
	12.0(4)XI	This command was introduced.

Usage Guidelines Enter the **debug resource-pool** command to see and trace resource pool management activity. [Table 209](#) describes the resource pooling states.

Table 209 Resource Pooling States

State	Description
RM_IDLE	No call activity.
RM_RES_AUTHOR	Call waiting for authorization, message sent to 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.
RM_DNIS_AUTHOR	An intermediate state before proceeding with RPM authorization.
RM_DNIS_AUTH_SUCCEEDED	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.

Table 209 Resource Pooling States (continued)

State	Description
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)
cpl (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 210 describes the significant fields shown in the command output.

Table 210 *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.
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

```

52559

The first line of output is an example of a RIF entry for an interface configured for SDLLC or Local-Ack. [Table 211](#) describes significant fields shown in the display.

Table 211 *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 211 *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"> • 0—Null entry • 1—This entry was learned from a particular Token Ring port (interface) • 2—Statically configured • 4—Statically configured for a remote interface • 8—This entry is to be aged • 16—This entry (which has been learned from a remote interface) is to be aged • 32—This entry is not to be aged • 64—This interface is to be used by LAN Network Manager (and is not to be aged)
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 211, Part 1 explains the other possible lines of **debug rif** Command output.

Table 211, Part 1 *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
debug list	Filters debugging information on a per-interface or per-access list basis.

debug route-map ipc

To display a summary of the one-way IPC messages set from the RP to the 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.

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 of 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 of 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 diagnostic reporting, use the **no** form of this command.

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

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

Syntax Description

all	Provides information for all calls.
h323	Provides information for H.323 calls.
sip	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.

Examples

The following example shows debug 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 212 describes the significant fields shown in the display.

Table 212 *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 rtr error

To enable logging of SA Agent 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 [*probe*]

no debug rtr error [*probe*]

Syntax Description: *probe* (Optional) Number of the probe in the range from 0 to 31.

Defaults Logging is off.

Command Modes Privileged EXEC

Command History	Release	Modification
	11.2	This command was introduced.
	12.0(5)T	This command was modified.

Usage Guidelines The **debug rtr error** command displays run-time errors. When a probe number other than 0 is specified, all run-time errors for that probe are displayed when the probe is active. When the probe number is 0 all run-time errors relating to the Response Time Reporter scheduler process are displayed. When no probe number is specified, all run-time errors for all active probes configured on the router and probe control 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 debug output.

Examples The following example shows 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 debug output for the Response Time Reporter (including the **debug rtr trace** command) has the format shown in [Table 213](#).

```
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 213 describes the significant fields shown in the display.

Table 213 *debug rtr error Field Descriptions*

Field	Description
RTR 1	Number of the probe 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
debug rtr trace	Traces the execution of an SA Agent operation.

debug rtr trace

To trace the execution of an SA Agent operation, use the **debug rtr trace** command in privileged EXEC mode. To disable trace debugging output (but not **debug rtr error** output), use the **no** form of this command.

debug rtr trace [*probe*]

no debug rtr trace [*probe*]

Syntax Description:

probe (Optional) Number of the probe in the range from 0 to 31.

Command Modes

Privileged EXEC

Command History

Release	Modification
11.2	This command was introduced.
12.0(5)T	This command was modified.

Usage Guidelines

When a probe number other than 0 is specified, execution for that probe is traced. When the probe number is 0, the Response Time Reporter scheduler process is traced. When no probe number is specified, all active probes and every probe control is traced.

The **debug rtr trace** command also enables **debug rtr error** command for the specified probe. 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 debug 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-probe basis.

Examples

The following output is from the **debug rtr trace** command. In this example, a probe 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
debug rtr error	Enables logging of SA Agent 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"> • all—(Optional) Displays debug output for all clients or servers. • api—(Optional) Displays debug output for the client or server API. • error—(Optional) Displays errors when they are errors otherwise no output is displayed. • pmh—(Optional) Displays debug output for the Protocol Message Handler (PMH). • session—(Optional) Displays debug output for the client or server session. • socket—(Optional) Displays debug output for the client or server socket data.
-------------	--

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.

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
debug rtsp api	Displays debug output for the RTSP client API.
debug rtsp client session	Displays debug output for the RTSP client data.
debug rtsp socket	Displays debug output for the RTSP client socket data.

debug rtsp api

To display information about the Real Time Streaming Protocol (RTSP) API messages passed down to the RTSP client, use the **debug rtsp api** command. 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 History

Release	Modification
12.1(3)T	This command was introduced.

Examples

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

```
router# debug rtsp api

RTSP client API debugging is on
router#
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
router#
Jan  1 00:23:15.775:rtsp_api_request:msg=0x61C29A4C
router#
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
Router#
```

Related Commands

Command	Description
debug rtsp client session	Displays debug output for the RTSP client data.
debug rtsp pmh	Displays debug messages for the PMH.
debug rtsp socket	Displays debug output for the RTSP client socket data.

debug rtsp client session

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

client	(Optional) Displays client information and stream information for the stream that is currently active.
session	(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.

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:CCAPI 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
debug rtsp api	Displays debug output for the RTSP client API.
debug rtsp client session	Displays debug output for the RTSP client data.
debug rtsp pmh	Displays debug messages for the PMH.
debug rtsp socket	Displays debug output for the RTSP client socket data.

debug rtsp error

To display error messages for failures encountered by the Real Time Streaming Protocol (RTSP) client, use the **debug rtsp error** command in privileged EXEC mode. To disable error debugging, use the **no** form of this command.

debug rtsp error

no debug rtsp error

Syntax Description This command has no arguments or keywords.

Defaults No default behavior or values.

Command Modes Privileged EXEC

Command History	Release	Modification
	12.2(11)T	This command was introduced on the Cisco 3640, Cisco 3660, Cisco AS5300, Cisco AS5350, and Cisco AS5400.

Examples The following examples show output from the **debug rtsp error** command.

```
Router# debug rtsp error
```

The following example shows the message that displays when the RTSP connection fails:

```
*Sep 25 15:02:32.052: //-1//RTSP:/rtsplib_connect_to_svr: Socket Connect failed:
172.19.140.31:554
```

The following example shows the message that displays when the RTSP client receives an incorrect response from the server:

```
*Sep 25 15:03:35.062: //-1//RTSP:/rtsp_process_single_svr_resp: Parse Server Response
failed, 172.19.140.31:554
```

The following example shows the message that displays when the codec configured on the IP side is not G.711:

```
*Sep 25 15:05:15.765: //-1//RTSP:/rtsplib_rtp_associate_done: Association mismatch
```

Related Commands

Command	Description
debug mrcp	Displays debug messages for MRCP operations.
debug rtsp session	Displays debug messages about the current RTSP client session.
debug rtsp socket	Displays debug messages about the packets received or sent on the TCP or User Datagram Protocol (UDP) sockets.
show mrcp client session active	Displays information about active MRCP sessions.

debug rtsp pmh

To display debug information about the Protocol Message Handler (PMH), use the **debug rtsp pmh** command. 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 History

Release	Modification
12.1(3)T	This command was introduced.

Usage Guidelines

Use the **debug rtsp pmh** debug command for the following instances:

- To display packets sent by the gateway (Real Time Streaming Protocol [RTSP] client) to the RTSP server. For example:

```
Mar  1 02:25:11.447:SendBuf:DESCRIBE rtsp://rtsp-cisco.cisco.com/en_welcome.au
RTSP/1.0
CSeq:0
```

- To view packets sent by the RTSP server to the gateway. For example:

```
Mar  1 02:25:11.947:#####
Mar  1 02:25:11.947:Mesg_line           :RTSP/1.0 200 OK
Mar  1 02:25:11.951:Content_length      :459
Mar  1 02:25:11.951:Header list
Mar  1 02:25:11.951:Content-length:459
Mar  1 02:25:11.951:Content-type:application/sdp
Mar  1 02:25:11.951:Content-base:rtsp://rtsp-cisco.cisco.com/en_welcome.au/
Mar  1 02:25:11.951:X-TSPort:7802
Mar  1 02:25:11.951>Last-Modified:Thu, 07 Oct 1999 13:51:28 GMT
Mar  1 02:25:11.951>Date:Mon, 10 Jan 2000 16:40:59 GMT
Mar  1 02:25:11.951:CSeq:0
```

Examples

The following example output displays the result from entering the **debug rtsp pmh** command:

```
Router# debug rtsp pmh

RTSP client Protocol Message Handler debugging is on
Router#
Jan  1 00:22:34.087:rtsp_pmh_update_play_req_url:
Jan  1 00:22:34.087:rtsp_pmh_parse_url:
Jan  1 00:22:34.087:Input-Url:rtsp://rtsp-cisco.cisco.com:554/en_welcome.au
Jan  1 00:22:34.087:Hostname:rtsp-cisco.cisco.com
Jan  1 00:22:34.087:Port      :554
Jan  1 00:22:34.087:Path     :en_welcome.au
```

```

Jan 1 00:22:34.091:rtsp_pmh_build_desc_req:
Jan 1 00:22:34.091:rtsp_pmh_add_req_line:
Jan 1 00:22:34.091:RequestLine:(DESCRIBE rtsp://rtsp-cisco.cisco.com:554/en_welcome.au
RTSP/1.0
)
Jan 1 00:22:34.091:SendBuf:DESCRIBE rtsp://rtsp-cisco.cisco.com:554/en_welcome.au
RTSP/1.0
CSeq:0

Jan 1 00:22:34.091:last_req = 0
Jan 1 00:22:34.739:rtsp_pmh_parse_svr_response:
Jan 1 00:22:34.739:rtsp_pmh_create_mesg:
Jan 1 00:22:34.739:#####
Jan 1 00:22:34.739:Mesg_line           :RTSP/1.0 200 OK
Jan 1 00:22:34.739:Content_length      :482
Jan 1 00:22:34.739:Header list
Jan 1 00:22:34.739:Content-length:482
Jan 1 00:22:34.739:Content-type:application/sdp
Jan 1 00:22:34.739:Content-base:rtsp://rtsp-cisco.cisco.com:554/en_welcome.au/
Jan 1 00:22:34.739>Last-Modified:Thu, 07 Oct 1999 13:51:28 GMT
Jan 1 00:22:34.739:X-TSPort:7802
Jan 1
00:22:34.739:vsrc:http://rtsp-cisco.cisco.com:8080/viewsource/template.html?nuyhtgywkgz6mc
9AbhC4gn5gBsqq4eAlv1yeC3d4ngEt5o5gwuw4t6x05jbhcv66ngE8xg8f
Jan 1
00:22:34.739:Set-Cookie:cbid=ekeghhiljgekgihheoqohprrrjrktlufkegkioihgjfdlplnrqogpoqlrpsk
qnuffgjcml;path=/;expires=Thu,31-Dec-2037 23:59:59 GMT
Jan 1 00:22:34.739>Date:Mon, 10 Apr 2000 15:39:17 GMT
Jan 1 00:22:34.739:CSeq:0
Jan 1 00:22:34.739:Message Body
Jan 1 00:22:34.739:v=0
o=- 939300688 939300688 IN IP4 1.13.79.6
s=<No title>
i=<No author> <No copyright>
a=StreamCount:integer;1
t=0 0
m=audio 0 RTP/AVP 0
a=control:streamid=0
a=rtpmap:0 L8/8000/1
a=length:npt=3.249000
a=range:npt=0-3.249000
a=mimetype:string;"audio/x-pn-au"
a=StartTime:integer;0
a=AvgBitRate:integer;64000
a=AvgPacketSize:integer;320
a=Preroll:integer;0
a=MaxPacketSize:integer;320
a=MaxBitRate:integer;64000
a=OpaqueData:buffer;"AQABAEfAAA="
a=StreamName:string;"audio/x-pn-au"

Jan 1 00:22:34.739:#####
Jan 1 00:22:34.739:rtsp_pmh_process_resp_headers:
Jan 1 00:22:34.739:rtsp_pmh_get_header_value:
Jan 1 00:22:34.739:rtsp_pmh_process_resp_headers:Cseq=1
Jan 1 00:22:34.739:rtsp_pmh_get_resp_line:
Jan 1 00:22:34.739:rtsp_pmh_process_resp_headers:Response Status
Jan 1 00:22:34.739:rtsp_pmh_process_resp_headers:Status Code:200
Jan 1 00:22:34.739:rtsp_pmh_process_resp_headers:Reason Phrase:OK
Jan 1 00:22:34.743:rtsp_pmh_parse_mesg_body:
Jan 1 00:22:34.743:rtsp_pmh_process_resp_headers:Response
URL:rtsp://rtsp-cisco.cisco.com:554/en_welcome.au/streamid=0
Jan 1 00:22:34.743:rtsp_pmh_process_resp_headers:RealServer Duration

```

```

Jan 1 00:22:34.743:rtsp_pmh_process_resp_headers:IP/TV Duration
Jan 1 00:22:34.743:rtsp_pmh_get_range_from_npt:
Jan 1 00:22:34.743:rtsp_pmh_get_range_from_npt:Duration:3249 msec
Jan 1 00:22:34.743:rtsp_pmh_update_resp_status:
Jan 1 00:22:34.743:rtsp_pmh_update_resp_status:Control Not active
Jan 1 00:22:34.743:#####
Jan 1 00:22:34.743:Mesg_line           :RTSP/1.0 200 OK
Jan 1 00:22:34.743:Content_length      :482
Jan 1 00:22:34.743:Header list
Jan 1 00:22:34.743:Content-length:482
Jan 1 00:22:34.743:Content-type:application/sdp
Jan 1 00:22:34.743:Content-base:rtsp://rtsp-cisco.cisco.com:554/en_welcome.au/
Jan 1 00:22:34.743>Last-Modified:Thu, 07 Oct 1999 13:51:28 GMT
Jan 1 00:22:34.743:X-TSPort:7802
Jan 1
00:22:34.743:vsrc:http://rtsp-cisco.cisco.com:8080/viewsource/template.html?nuyhtgywkgz6mc
9AbhC4gn5gBsqp4eA1vlveC3d4ngEt5o5gwuw4t6x05jbhcv66ngE8xg8f
Jan 1
00:22:34.743:Set-Cookie:cbid=ekeghhiljgekgihheoqhpptrrjrktlufkegkioihgjfdlplrnqogpoqlrpsk
qnuffgjcml;path=/;expires=Thu,31-Dec-2037 23:59:59 GMT
Jan 1 00:22:34.743>Date:Mon, 10 Apr 2000 15:39:17 GMT
Jan 1 00:22:34.743:CSeq:0
Jan 1 00:22:34.743:Message Body
Jan 1 00:22:34.743:v=0
o=- 939300688 939300688 IN IP4 1.13.79.6
s=<No title>
i=<No author> <No copyright>
a=StreamCount:integer;1
t=0 0
m=audio 0 RTP/AVP 0
a=control:streamid=0
a=rtptime:0 L8/8000/1
a=length:npt=3.249000
a=range:npt=0-3.249000
a=mimetype:string;"audio/x-pn-au"
a=StartTime:integer;0
a=AvgBitRate:integer;64000
a=AvgPacketSize:integer;320
a=Preroll:integer;0
a=MaxPacketSize:integer;320
a=MaxBitRate:integer;64000
a=OpaqueData:buffer;"AQABAEfAAA="
a=StreamName:string;"audio/x-pn-au"

Jan 1 00:22:34.743:#####
Jan 1 00:22:34.743:rtsp_pmh_free_mesg:
Jan 1 00:22:34.743:rtsp_pmh_build_setup_req:
Jan 1 00:22:34.743:rtsp_pmh_add_req_line:
Jan 1 00:22:34.743:RequestLine:(SETUP
rtsp://rtsp-cisco.cisco.com:554/en_welcome.au/streamid=0 RTSP/1.0
)
Jan 1 00:22:34.747:rtsp_pmh_build_setup_req:SendBuf:SETUP
rtsp://rtsp-cisco.cisco.com:554/en_welcome.au/streamid=0 RTSP/1.0
CSeq:1
Transport:rtp/avp;unicast;client_port=18084

Jan 1 00:22:35.243:rtsp_pmh_parse_svr_response:
Jan 1 00:22:35.243:rtsp_pmh_create_mesg:
Jan 1 00:22:35.243:#####
Jan 1 00:22:35.243:Mesg_line           :RTSP/1.0 200 OK
Jan 1 00:22:35.243:Content_length      :0
Jan 1 00:22:35.243:Header list

```

```

Jan 1
00:22:35.243:Transport:rtp/avp;unicast;client_port=18084-18085;server_port=23192-23193
Jan 1 00:22:35.243:Session:24457-1
Jan 1 00:22:35.243>Date:Mon, 10 Apr 2000 15:39:17 GMT
Jan 1 00:22:35.243:CSeq:1
Jan 1 00:22:35.243:Message Body
Jan 1 00:22:35.243:#####
Jan 1 00:22:35.243:rtsp_pmh_process_resp_headers:
Jan 1 00:22:35.243:rtsp_pmh_get_header_value:
Jan 1 00:22:35.243:rtsp_pmh_process_resp_headers:Cseq=2
Jan 1 00:22:35.243:rtsp_pmh_get_resp_line:
Jan 1 00:22:35.243:rtsp_pmh_process_resp_headers:Response Status
Jan 1 00:22:35.243:rtsp_pmh_process_resp_headers:Status Code:200
Jan 1 00:22:35.243:rtsp_pmh_process_resp_headers:Reason Phrase:OK
Jan 1 00:22:35.243:rtsp_pmh_get_header_value:
Jan 1 00:22:35.247:rtsp_pmh_get_header_value:
Jan 1 00:22:35.247:rtsp_pmh_process_resp_headers:RTP PORT= 23192
Jan 1 00:22:35.247:rtsp_pmh_process_resp_headers:RTP PORT= 23192
Jan 1 00:22:35.247:rtsp_pmh_update_resp_status:
Jan 1 00:22:35.247:rtsp_pmh_update_resp_status:Control Not active
Jan 1 00:22:35.247:#####
Jan 1 00:22:35.247:Mesg_line           :RTSP/1.0 200 OK
Jan 1 00:22:35.247:Content_length       :0
Jan 1 00:22:35.247:Header list
Jan 1
00:22:35.247:Transport:rtp/avp;unicast;client_port=18084-18085;server_port=23192-23193
Jan 1 00:22:35.247:Session:24457-1
Jan 1 00:22:35.247>Date:Mon, 10 Apr 2000 15:39:17 GMT
Jan 1 00:22:35.247:CSeq:1
Jan 1 00:22:35.247:Message Body
Jan 1 00:22:35.247:#####
Jan 1 00:22:35.247:rtsp_pmh_free_mesg:
Jan 1 00:22:35.247:rtsp_pmh_build_play_req:
Jan 1 00:22:35.247:rtsp_pmh_add_req_line:
Jan 1 00:22:35.247:RequestLine:(PLAY
rtsp://rtsp-cisco.cisco.com:554/en_welcome.au/streamid=0 RTSP/1.0
)
Jan 1 00:22:35.247:rtsp_pmh_build_play_req:SendBuf:PLAY
rtsp://rtsp-cisco.cisco.com:554/en_welcome.au/streamid=0 RTSP/1.0
Session:24457-1
CSeq:2

Jan 1 00:22:35.735:rtsp_pmh_parse_svr_response:
Jan 1 00:22:35.735:rtsp_pmh_create_mesg:
Jan 1 00:22:35.739:#####
Jan 1 00:22:35.739:Mesg_line           :RTSP/1.0 200 OK
Jan 1 00:22:35.739:Content_length       :0
Jan 1 00:22:35.739:Header list
Jan 1 00:22:35.739>Date:Mon, 10 Apr 2000 15:39:18 GMT
Jan 1 00:22:35.739:CSeq:2
Jan 1 00:22:35.739:Message Body
Jan 1 00:22:35.739:#####
Jan 1 00:22:35.739:rtsp_pmh_process_resp_headers:
Jan 1 00:22:35.739:rtsp_pmh_get_header_value:
Jan 1 00:22:35.739:rtsp_pmh_process_resp_headers:Cseq=3
Jan 1 00:22:35.739:rtsp_pmh_get_resp_line:
Jan 1 00:22:35.739:rtsp_pmh_process_resp_headers:Response Status
Jan 1 00:22:35.739:rtsp_pmh_process_resp_headers:Status Code:200
Jan 1 00:22:35.739:rtsp_pmh_process_resp_headers:Reason Phrase:OK
Jan 1 00:22:35.739:rtsp_pmh_update_resp_status:
Jan 1 00:22:35.739:rtsp_pmh_update_resp_status:Control Not active
Jan 1 00:22:35.739:#####
Jan 1 00:22:35.739:Mesg_line           :RTSP/1.0 200 OK

```

```

Jan 1 00:22:35.739:Content_length      :0
Jan 1 00:22:35.739:Header list
Jan 1 00:22:35.739>Date:Mon, 10 Apr 2000 15:39:18 GMT
Jan 1 00:22:35.739:CSeq:2
Jan 1 00:22:35.739:Message Body
Jan 1 00:22:35.739:#####
Jan 1 00:22:35.739:rtsp_pmh_free_mesg:
Jan 1 00:22:40.011:rtsp_pmh_build_teardown_req:
Jan 1 00:22:40.011:rtsp_pmh_add_req_line:
Jan 1 00:22:40.011:RequestLine:(TEARDOWN
rtsp://rtsp-cisco.cisco.com:554/en_welcome.au/streamid=0 RTSP/1.0
)
Jan 1 00:22:40.011:SendBuf:TEARDOWN
rtsp://rtsp-cisco.cisco.com:554/en_welcome.au/streamid=0 RTSP/1.0
Session:24457-1
CSeq:3

Jan 1 00:22:40.443:rtsp_pmh_parse_svr_response:
Jan 1 00:22:40.443:rtsp_pmh_create_mesg:
Jan 1 00:22:40.443:#####
Jan 1 00:22:40.443:Mesg_line          :RTSP/1.0 200 OK
Jan 1 00:22:40.443:Content_length      :0
Jan 1 00:22:40.443:Header list
Jan 1 00:22:40.443>Date:Mon, 10 Apr 2000 15:39:23 GMT
Jan 1 00:22:40.443:CSeq:3
Jan 1 00:22:40.443:Message Body
Jan 1 00:22:40.443:#####
Jan 1 00:22:40.443:rtsp_pmh_process_resp_headers:
Jan 1 00:22:40.443:rtsp_pmh_get_header_value:
Jan 1 00:22:40.443:rtsp_pmh_process_resp_headers:Cseq=4
Jan 1 00:22:40.443:rtsp_pmh_get_resp_line:
Jan 1 00:22:40.443:rtsp_pmh_process_resp_headers:Response Status
Jan 1 00:22:40.443:rtsp_pmh_process_resp_headers:Status Code:200
Jan 1 00:22:40.443:rtsp_pmh_process_resp_headers:Reason Phrase:OK
Jan 1 00:22:40.443:rtsp_pmh_update_resp_status:
Jan 1 00:22:40.443:rtsp_pmh_update_resp_status:Control Not active
Jan 1 00:22:40.443:#####
Jan 1 00:22:40.447:Mesg_line          :RTSP/1.0 200 OK
Jan 1 00:22:40.447:Content_length      :0
Jan 1 00:22:40.447:Header list
Jan 1 00:22:40.447>Date:Mon, 10 Apr 2000 15:39:23 GMT
Jan 1 00:22:40.447:CSeq:3
Jan 1 00:22:40.447:Message Body
Jan 1 00:22:40.447:#####
Jan 1 00:22:40.447:rtsp_pmh_free_mesg:
Router#

Jan 1 00:14:20.483:rtsp_tcp_socket_connect:
Jan 1 00:14:20.483:rtsp_tcp_socket_connect:Socket = 0
Jan 1 00:14:20.483:      Dest_addr = 1.13.79.6  Dest_Port=554
Jan 1 00:14:20.487:rtsp_send_req_to_svr:Socket = 0 send_buf = DESCRIBE
rtsp://rtsp-cisco.cisco.com:554/en_welcome.au RTSP/1.0
CSeq:0

len = 76
Jan 1 00:14:20.491:rtsp_send_req_to_svr:bytes_sent = 76

Jan 1 00:14:20.491:rtsp_read_svr_resp:Socket = 0
Jan 1 00:14:20.491:rtsp_read_svr_resp:NBYTES = -1
Jan 1 00:14:21.155:rtsp_read_svr_resp:Socket = 0
Jan 1 00:14:21.159:rtsp_read_svr_resp:NBYTES = 996
Jan 1 00:14:21.223:rtsp_read_svr_resp:rtsp_pmh_parse_svr_response complete

```

```
Jan 1 00:14:21.227:rtsp_read_svr_resp:RESP received OK
Jan 1 00:14:21.227:rtsp_send_req_to_svr:Socket = 0 send_buf = SETUP
rtsp://rtsp-cisco.cisco.com:554/en_welcome.au/streamid=0 RTSP/1.0
CSeq:1
Transport:rtp/avp;unicast;client_port=18074

len = 130
Jan 1 00:14:21.227:rtsp_send_req_to_svr:bytes_sent = 130

Jan 1 00:14:21.663:rtsp_read_svr_resp:Socket = 0
Jan 1 00:14:21.663:rtsp_read_svr_resp:NBYTES = 159
Jan 1 00:14:21.663:rtsp_read_svr_resp:rcv_buf = RTSP/1.0 200 OK
CSeq:1
Date:Mon, 10 Apr 2000 15:31:04 GMT
Session:24455-1
Transport:rtp/avp;unicast;client_port=18074-18075;server_port=15562-15563

Jan 1 00:14:21.663:rtsp_read_svr_resp:rtsp_pmh_parse_svr_response complete
Jan 1 00:14:21.663:rtsp_read_svr_resp:RESP received OK
Jan 1 00:14:21.663:rtsp_send_req_to_svr:Socket = 0 send_buf = PLAY
rtsp://rtsp-cisco.cisco.com:554/en_welcome.au/streamid=0 RTSP/1.0
Session:24455-1
CSeq:2

len = 101
Jan 1 00:14:21.667:rtsp_send_req_to_svr:bytes_sent = 101

Jan 1 00:14:22.155:rtsp_read_svr_resp:Socket = 0
Jan 1 00:14:22.155:rtsp_read_svr_resp:NBYTES = 65
Jan 1 00:14:22.155:rtsp_read_svr_resp:rcv_buf = RTSP/1.0 200 OK
CSeq:2
Date:Mon, 10 Apr 2000 15:31:04 GMT

Jan 1 00:14:22.155:rtsp_read_svr_resp:rtsp_pmh_parse_svr_response complete
Jan 1 00:14:22.155:rtsp_read_svr_resp:RESP received OK
rtsp-5#
Jan 1 00:14:26.411:rtsp_send_req_to_svr:Socket = 0 send_buf = TEARDOWN
rtsp://rtsp-cisco.cisco.com:554/en_welcome.au/streamid=0 RTSP/1.0
Session:24455-1
CSeq:3

len = 105
Jan 1 00:14:26.411:rtsp_send_req_to_svr:bytes_sent = 105

Jan 1 00:14:26.863:rtsp_read_svr_resp:Socket = 0
Jan 1 00:14:26.863:rtsp_read_svr_resp:NBYTES = 65
Jan 1 00:14:26.863:rtsp_read_svr_resp:rcv_buf = RTSP/1.0 200 OK
CSeq:3
Date:Mon, 10 Apr 2000 15:31:09 GMT

Jan 1 00:14:26.863:rtsp_read_svr_resp:rtsp_pmh_parse_svr_response complete
Jan 1 00:14:26.863:rtsp_read_svr_resp:RESP received OK
Jan 1 00:14:26.863:rtsp_close_svr_connection:closing socket 0
Router#
```

Related Commands

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

debug rtsp socket

To display debug 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

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	Messages for speech recognition and speech synthesis operations were added.

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). The **debug rtsp socket** command enables the user to debug the message exchanges being done on the TCP control connection.

Examples

The following examples display output from the **debug rtsp socket** command:

```
Router# debug rtsp socket

Jan  1 00:14:20.483:rtsp_tcp_socket_connect:
Jan  1 00:14:20.483:rtsp_tcp_socket_connect:Socket = 0
Jan  1 00:14:20.483:                Dest_addr = 1.13.79.6  Dest_Port=554
Jan  1 00:14:20.487:rtsp_send_req_to_svr:Socket = 0 send_buf = DESCRIBE
rtsp://rtsp-cisco.cisco.com:554/en_welcome.au RTSP/1.0
CSeq:0

len = 76
Jan  1 00:14:20.491:rtsp_send_req_to_svr:bytes_sent = 76

Jan  1 00:14:20.491:rtsp_read_svr_resp:Socket = 0
Jan  1 00:14:20.491:rtsp_read_svr_resp:NBYTES = -1
Jan  1 00:14:21.155:rtsp_read_svr_resp:Socket = 0
Jan  1 00:14:21.159:rtsp_read_svr_resp:NBYTES = 996
Jan  1 00:14:21.223:rtsp_read_svr_resp:rtsp_pmh_parse_svr_response complete
Jan  1 00:14:21.227:rtsp_read_svr_resp:RESP received OK
```

```

Jan  1 00:14:21.227:rtsp_send_req_to_svr:Socket = 0 send_buf = SETUP
rtsp://rtsp-cisco.cisco.com:554/en_welcome.au/streamid=0 RTSP/1.0
CSeq:1
Transport:rtp/avp;unicast;client_port=18074

len = 130
Jan  1 00:14:21.227:rtsp_send_req_to_svr:bytes_sent = 130

Jan  1 00:14:21.663:rtsp_read_svr_resp:Socket = 0
Jan  1 00:14:21.663:rtsp_read_svr_resp:NBYTES = 159
Jan  1 00:14:21.663:rtsp_read_svr_resp:rcv_buf = RTSP/1.0 200 OK
CSeq:1
Date:Mon, 10 Apr 2000 15:31:04 GMT
Session:24455-1
Transport:rtp/avp;unicast;client_port=18074-18075;server_port=15562-15563

Jan  1 00:14:21.663:rtsp_read_svr_resp:rtsp_pmh_parse_svr_response complete
Jan  1 00:14:21.663:rtsp_read_svr_resp:RESP received OK
Jan  1 00:14:21.663:rtsp_send_req_to_svr:Socket = 0 send_buf = PLAY
rtsp://rtsp-cisco.cisco.com:554/en_welcome.au/streamid=0 RTSP/1.0
Session:24455-1
CSeq:2

len = 101
Jan  1 00:14:21.667:rtsp_send_req_to_svr:bytes_sent = 101

Jan  1 00:14:22.155:rtsp_read_svr_resp:Socket = 0
Jan  1 00:14:22.155:rtsp_read_svr_resp:NBYTES = 65
Jan  1 00:14:22.155:rtsp_read_svr_resp:rcv_buf = RTSP/1.0 200 OK
CSeq:2
Date:Mon, 10 Apr 2000 15:31:04 GMT

Jan  1 00:14:22.155:rtsp_read_svr_resp:rtsp_pmh_parse_svr_response complete
Jan  1 00:14:22.155:rtsp_read_svr_resp:RESP received OK
rtsp-5#
Jan  1 00:14:26.411:rtsp_send_req_to_svr:Socket = 0 send_buf = TEARDOWN
rtsp://rtsp-cisco.cisco.com:554/en_welcome.au/streamid=0 RTSP/1.0
Session:24455-1
CSeq:3

len = 105
Jan  1 00:14:26.411:rtsp_send_req_to_svr:bytes_sent = 105

Jan  1 00:14:26.863:rtsp_read_svr_resp:Socket = 0
Jan  1 00:14:26.863:rtsp_read_svr_resp:NBYTES = 65
Jan  1 00:14:26.863:rtsp_read_svr_resp:rcv_buf = RTSP/1.0 200 OK
CSeq:3
Date:Mon, 10 Apr 2000 15:31:09 GMT

Jan  1 00:14:26.863:rtsp_read_svr_resp:rtsp_pmh_parse_svr_response complete
Jan  1 00:14:26.863:rtsp_read_svr_resp:RESP received OK
Jan  1 00:14:26.863:rtsp_close_svr_connection:closing socket 0

```

The following example shows output from the **debug rtsp socket** command when doing speech recognition and synthesis:

```

*Sep 25 15:01:43.824: //-1//RTSP:/rtsp_tcp_socket_connect:
*Sep 25 15:01:43.824: //-1//RTSP:/rtsp_tcp_socket_connect: Socket = 0, Dest_addr =
172.19.140.31, Dest_Port=554
*Sep 25 15:01:43.832: //-1//RTSP:/rtsp_read_svr_resp: Socket = 0
*Sep 25 15:01:43.832: //-1//RTSP:/rtsp_read_svr_resp: NBYTES = -1
*Sep 25 15:01:43.832: //-1//RTSP:/rtsp_partial_socket_send:
*Sep 25 15:01:43.832: rtsp_partial_socket_send: (fd:0 len:116) 400 bytes of data:

```

```

SETUP rtsp://nuance-asr/recognizer RTSP/1.0
CSeq: 0
Transport: rtp/avp;unicast;client_port=19206;mode=record

*Sep 25 15:01:43.832: //-1//RTSP:/rtsp_partial_socket_send:
*Sep 25 15:01:43.832: (socket:0) (bytes-sent:116)
*Sep 25 15:01:43.896: //-1//RTSP:/rtsp_read_svr_resp: Socket = 0
*Sep 25 15:01:43.896: //-1//RTSP:/rtsp_read_svr_resp: NBYTES = 410
*Sep 25 15:01:43.896: //-1//RTSP:/rtsp_process_single_svr_resp:
*Sep 25 15:01:43.896: rtsp_process_single_svr_resp: 400 bytes of data:
RTSP/1.0 200 OK
CSeq: 0
Session: ac138c1f_0000017c_3d4817ce_0018_0000
Transport: RTP/AVP;unicast;client_port=19206;server_port=4732-4733;mode=record
Content-Length: 203
Content-Type: application/sdp

v=0
o=- 3237123662 3237123662 IN IP4 172.19.140.31
s=Nuance Media Server
c=IN IP4 0.0.0.0
t=0 0
m=audio 4732 RTP/AVP 0 101
a=rtpmap:0 pcmu/8000
a=rtpmap:101 telephone-event/8000
a=fmtp:aQ
*Sep 25 15:01:43.900: //-1//RTSP:/rtsp_process_single_svr_resp: complete
*Sep 25 15:01:43.900: //-1//RTSP:/rtsp_process_single_svr_resp: Response sent
*Sep 25 15:01:43.900: //-1//RTSP:/rtsp_partial_socket_send:
*Sep 25 15:01:43.900: rtsp_partial_socket_send: (fd:0 len:158) 400 bytes of data:
SETUP rtsp://nuance-asr/synthesizer RTSP/1.0
CSeq: 1
Session: ac138c1f_0000017c_3d4817ce_0018_0000
Transport: rtp/avp;unicast;client_port=19206-19207

*Sep 25 15:01:43.900: //-1//RTSP:/rtsp_partial_socket_send:
*Sep 25 15:01:43.900: (socket:0) (bytes-sent:158)
*Sep 25 15:01:43.900: //-1//RTSP:/rtsp_partial_socket_send:
*Sep 25 15:01:43.900: rtsp_partial_socket_send: (fd:0 len:163) 400 bytes of data:
ANNOUNCE rtsp://nuance-asr/recognizer RTSP/1.0
CSeq: 2
Session: ac138c1f_0000017c_3d4817ce_0018_0000
Content-Type: application/mrcp
Content-Length: 1702

*Sep 25 15:01:43.900: //-1//RTSP:/rtsp_partial_socket_send:
*Sep 25 15:01:43.904: (socket:0) (bytes-sent:163)
*Sep 25 15:01:43.904: //-1//RTSP:/rtsp_partial_socket_send:
*Sep 25 15:01:43.904: rtsp_partial_socket_send: (fd:0 len:93) 400 bytes of data:
DEFINE-GRAMMAR 45 MRCP/1.0
Speech-Language: en-US
Content-Base: file:///z:/test-content/

*Sep 25 15:01:43.904: //-1//RTSP:/rtsp_partial_socket_send:
*Sep 25 15:01:43.904: (socket:0) (bytes-sent:93)
*Sep 25 15:01:43.904: //-1//RTSP:/rtsp_partial_socket_send:
*Sep 25 15:01:43.904: rtsp_partial_socket_send: (fd:0 len:97) 400 bytes of data:
Content-Type: application/grammar+xml
Content-Id: form7@dialog.grammar
Content-Length: 1512

```

```

*Sep 25 15:01:43.904: //-1//RTSP:/rtsp_partial_socket_send:
*Sep 25 15:01:43.904: (socket:0) (bytes-sent:97)
*Sep 25 15:01:43.904: //-1//RTSP:/rtsp_partial_socket_send:
*Sep 25 15:01:43.904: rtsp_partial_socket_send: (fd:0 len:1512) 400 bytes of data:

*Sep 25 15:01:43.956: //-1//RTSP:/rtsp_partial_socket_send:
*Sep 25 15:01:43.956: (socket:0) (bytes-sent:1512)
*Sep 25 15:01:43.960: //-1//RTSP:/rtsp_read_svr_resp: Socket = 0
*Sep 25 15:01:43.960: //-1//RTSP:/rtsp_read_svr_resp: NBYTES = 143
*Sep 25 15:01:43.960: //-1//RTSP:/rtsp_process_single_svr_resp:
*Sep 25 15:01:43.960: rtsp_process_single_svr_resp: 400 bytes of data:
RTSP/1.0 200 OK
CSeq: 1
Session: ac138c1f_0000017c_3d4817ce_0018_0000
Transport: RTP/AVP;unicast;client_port=19206;server_port=4732-4733

*Sep 25 15:01:43.960: //-1//RTSP:/rtsp_process_single_svr_resp: complete
*Sep 25 15:01:43.960: //-1//RTSP:/rtsp_process_single_svr_resp: Response sent
*Sep 25 15:01:43.968: //-1//RTSP:/rtsp_partial_socket_send:
*Sep 25 15:01:43.968: rtsp_partial_socket_send: (fd:0 len:162) 400 bytes of data:
ANNOUNCE rtsp://nuance-asr/recognizer RTSP/1.0
CSeq: 3
Session: ac138c1f_0000017c_3d4817ce_0018_0000
Content-Type: application/mrcp
Content-Length: 430

*Sep 25 15:01:43.968: //-1//RTSP:/rtsp_partial_socket_send:
*Sep 25 15:01:43.968: (socket:0) (bytes-sent:162)
*Sep 25 15:01:43.968: //-1//RTSP:/rtsp_partial_socket_send:
*Sep 25 15:01:43.968: rtsp_partial_socket_send: (fd:0 len:349) 400 bytes of data:
RECOGNIZE 46 MRCP/1.0
Confidence-Threshold: 50
Sensitivity-Level: 50
Speed-Vs-Accuracy: 50
Speech-Complete-Timeout: 300
Speech-Incomplete-Timeout: 3000
Dtmf-Interdigit-Timeout: 10000
Dtmf-Term-Timeout: 0
Dtmf-Term-Char: #
No-Input-Timeout: 20000
Logging-Tag: 33:33
Content-Base: file:///z:/test-content/
Recognizer-Start-Timers: false

```

Related Commands

Command	Description
debug rtsp api	Displays debug output for the RTSP client API.
debug rtsp client session	Displays debug output for the RTSP session.
debug rtsp error	Displays error messages for failures encountered by the RTSP client.
debug rtsp pmh	Displays debug messages for the PMH.

debug rtpspi all

To debug all RTP SPI errors, sessions, and in/out functions, use the **debug rtpspi all** EXEC command. 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 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) in a private release that was not generally available.

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

debug rtpspi errors

To debug RTP SPI errors, use the **debug rtpspi errors EXEC** command. 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 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) in a private release that was not generally available.

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:
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

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

debug rtpspi inout

To debug RTP SPI in/out functions, use the **debug rtpspi inout** EXEC command. 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 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) in a private release that was not generally available.

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_addr = 0x5000001,remote_ip_addr = 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
debug rtpspi all	Debugs all RTP SPI errors, sessions, and in/out functions.
debug rtpspi errors	Debugs RTP SPI errors.
debug rtpspi send-nse	Triggers the RTP SPI to send a triple redundant NSE.
debug sgcp errors	Debugs SGCP errors.
debug sgcp events	Debugs SGCP events.
debug sgcp packet	Debugs SGCP packets.
debug vtsp send-nse	Sends and debugs a triple redundant NSE from the DSP to a remote gateway.

debug rtpspi send-nse

To trigger the RTP SPI software module to send a triple redundant NSE, use the **debug rtpspi send-nse EXEC** command. 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	Parameter	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 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) in a private release that was not generally available.

Examples The following example shows the RTP SPI software module set to send an NSE:

```
router# debug rtpspi send-nse
```

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

debug rtpspi session

To debug all RTP SPI sessions, use the **debug rtpspi session** EXEC command. To disable debugging, use the **no** form of this command.

no debug rtpspi session

Syntax Description

This command has no arguments or keywords.

Defaults

No default behavior or values.

Command Modes

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) in a private release that was not generally available.

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
```

debug rtpspi session

```

*Mar 1 01:01:51.822:rtpspi_do_call_modify:Remote IP addrs changed. New IP addrs=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
debug rtpspi all	Debugs all RTP SPI errors, sessions, and in/out functions.
debug rtpspi errors	Debugs RTP SPI errors.
debug rtpspi inout	Debugs RTP SPI in/out functions.
debug rtpspi send-nse	Triggers the RTP SPI to send a triple redundant NSE.
debug sgcp errors	Debugs SGCP errors.
debug sgcp events	Debugs SGCP events.
debug sgcp packet	Debugs SGCP packets.
sgcp	Starts and allocates resources for the SCGP daemon.
debug vtsp send-nse	Sends and debugs a triple redundant NSE from the DSP to a remote gateway.

debug rtsp a11

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	The new debug header was added to the following Cisco routers: Cisco 2600 series, Cisco 3620, Cisco 3640, and Cisco 3660; on the following universal gateways: Cisco AS5350, Cisco AS5400, and Cisco AS5850; on the following universal access servers: Cisco AS5300, and Cisco AS5800; and, on the Cisco MC3810 multiservice access concentrators.

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 214 describes the significant fields shown in the display.

Table 214 *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.
rtsp_ <i>function name</i>	Identifies the function name.

Related Commands

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

debug rtsp api

To display information about the Real Time Streaming Protocol (RTSP) 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	The new debug header was added to the following Cisco routers: Cisco 2600 series, Cisco 3620, Cisco 3640, and Cisco 3660; on the following universal gateways: Cisco AS5350, Cisco AS5400, and Cisco AS5850; on the following universal access servers: Cisco AS5300, and Cisco AS5800; and, on the Cisco MC3810 multiservice access concentrators.

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 debugging output for 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 215 describes the significant fields shown in the display.

Table 215 *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.
<i>rtsp_function name</i>	Identifies the function name.

Related Commands

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

debug rtsp client

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.
12.2(11)T	The new debug header was added to the following Cisco routers: Cisco 2600 series, Cisco 3620, Cisco 3640, and Cisco 3660; on the following universal gateways: Cisco AS5350, Cisco AS5400, and Cisco AS5850; on the following universal access servers: Cisco AS5300, and Cisco AS5800; and, on the Cisco MC3810 multiservice access concentrators.

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

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	The new debug header was added to the following Cisco routers: Cisco 2600 series, Cisco 3620, Cisco 3640, and Cisco 3660; on the following universal gateways: Cisco AS5350, Cisco AS5400, and Cisco AS5850; on the following universal access servers: Cisco AS5300, and Cisco AS5800; and, on the Cisco MC3810 multiservice access concentrators.

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
debug rtsp api	Displays debug output for the RTSP client API.
debug rtsp client	Displays debug output for the RTSP client data.
debug rtsp pmh	Displays debug messages for the PMH.
debug rtsp socket	Displays debug output for the RTSP client socket data.
voice call debug	Allows configuration of the voice call debug 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	The new debug header was added to the following Cisco routers: Cisco 2600 series, Cisco 3620, Cisco 3640, and Cisco 3660; on the following universal gateways: Cisco AS5350, Cisco AS5400, and Cisco AS5850; on the following universal access servers: Cisco AS5300, and Cisco AS5800; and, on the Cisco MC3810 multiservice access concentrators.

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
debug rtsp api	Displays debug output for the RTSP client API.
debug rtsp client	Displays debug output for the RTSP client data.
debug rtsp error	Displays error message for RTSP data.
debug rtsp socket	Displays debug output for the RTSP client socket data.
voice call debug	Allows configuration of the voice call debug output.

debug rtsp session

To display 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 about the current Real-Time Streaming Protocol (RTSP) session, 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	The new debug header was added to the following Cisco routers: Cisco 2600 series, Cisco 3620, Cisco 3640, and Cisco 3660; on the following universal gateways: Cisco AS5350, Cisco AS5400, and Cisco AS5850; on the following universal access servers: Cisco AS5300, and Cisco AS5800; and, on the Cisco MC3810 multiservice access concentrators.

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 216 describes the significant fields shown in the display.

Table 216 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
debug rtsp api	Displays debug output for the RTSP client API.
debug rtsp error	Displays error message for RTSP data.
debug rtsp pmh	Displays debug messages for the PMH.
debug rtsp socket	Displays debug output for the RTSP client socket data.
voice call debug	Allows configuration of the voice call debug 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	The new debug header was added to the following Cisco routers: Cisco 2600 series, Cisco 3620, Cisco 3640, and Cisco 3660; on the following universal gateways: Cisco AS5350, Cisco AS5400, and Cisco AS5850; on the following universal access servers: Cisco AS5300, and Cisco AS5800; and, on the Cisco MC3810 multiservice access concentrators.

Usage Guidelines

Each 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
debug rtsp api	Displays debug output for the RTSP client API.
debug rtsp client	Displays debug output for the RTSP client data.
debug rtsp error	Displays error message for RTSP data.
debug rtsp pmh	Displays debug messages for the PMH.
voice call debug	Allows configuration of the voice call debug 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}
```



Caution

Use this command only during times of low traffic.

Syntax Description

application	Application debugging.
performance	Performance debugging.
retransmit	Retransmit/soft reset debugging.
segment	Segment debugging.
signal	Signals sent to applications.
state	State transitions.
timer	Timer debugging.
transfer	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). This command is not supported on the access servers in this release.
12.2(11)T	This command was integrated into Cisco IOS Release 12.2(11)T and implemented on Cisco AS5350, Cisco AS5400, and Cisco AS5850 platforms.

Examples

The following is output for 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 output for 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 output for 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 output for 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 output for 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 output for 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 output for 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 output for 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

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

debug saa apm

To enable debugging output for the SA Agent Application Performance Monitor (APM), 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.

Defaults No default behavior or values.

Command Modes Privileged EXEC

Command History	Release	Modification
	12.2(2)T	This command was introduced.

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 operation 29
```

```
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

To enable the output of detailed event messages for Service Assurance Agent (SAA) ATM Service Level Monitoring (SLM), use the **debug saa slm** command in privileged EXEC mode. To disable debugging message output, use the **no** form of this command.

debug saa slm

no debug saa slm

Syntax Description This command has no arguments or keywords.

Defaults Debug message output is disabled.

Command Modes Privileged EXEC

Command History	Release	Modification
	12.2(11)T	This command was introduced.

Usage Guidelines This command may generate a large amount of debugging messages.

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

```
Router# debug saa slm
Router# debug saa xml
```

Related Commands	Command	Description
	show rtr operational-state	Displays the accumulated monitoring statistics for the specified SAA operation.
	type atm-slm	Configures an ATM service level monitoring SAA operation.
	type slm	Configures an SLM PVC SAA operation.
	type t1-slm	Configures a T1-SLM SAA operation.

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

all	All SCCP debug-trace information.
errors	SCCP errors.
events	SCCP events.
packets	SCCP packets.
parser	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.

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

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 input event:

```
Serial3: SDLC O (12495952) C2 CONNECT (2) RR P/F 6
```

Table 217 describes the significant fields shown in the display.

Table 217 debug sdlc Field Descriptions for a Frame Output Event

Field	Description
SDLC	Protocol providing the information.
Serial3	Interface type and unit number reporting the frame event.
O	Command mode of frame event. Possible values are as follows: <ul style="list-style-type: none"> • I—Frame input • O—Frame output • T—T1 timer expired
(12495952)	Current timer value.
C2	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"> • CONNECT • DISCONNECT • DISCSENT (disconnect sent) • ERROR (FRMR frame sent) • REJSENT (reject frame sent) • SNRMSSENT (SNRM frame sent) • USBUSY • THEMBUSY • BOTHBUSY
(2)	Size of the frame (in bytes).
RR	Frame type name. Possible values are as follows: <ul style="list-style-type: none"> • DISC—Disconnect • DM—Disconnect mode • FRMR—Frame reject • IFRAME—Information frame • REJ—Reject • RNR—Receiver not ready • RR—Receiver ready • SIM—Set Initialization mode command • SNRM—Set Normal Response Mode • TEST—Test frame • UA—Unnumbered acknowledgment • XID—EXchange ID

Table 217 *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"> F—Final (printed for Response frames) P—Poll (printed for Command frames) P/F—Poll/Final (printed for RR, RNR, and REJ frames, which can be either Command or Response frames)
6	Receive count; range: 0 to 7.

The following line of output describes a frame input event:

```
Serial3: SDLC I (12495964) [C2] CONNECT (2) RR P/F 0 (R) [VR: 6 VS: 0] rfp: P
```

In addition to the fields described in [Table 217](#), output for a frame input event includes the fields described in [Table 218](#).

Table 218 *debug sdlc Field Descriptions Unique to a Frame Input Event*

Field	Description
(R)	Frame Type, as follows: <ul style="list-style-type: none"> C—Command R—Response
VR: 6	Receive count; range: 0 to 7.
VS: 0	Send count; range: 0 to 7.
rfp: P	Ready for poll, as follows: <ul style="list-style-type: none"> P—Idle poll (keepalive) timer is on. T—Data acknowledgment timer is on. These timers are based on the T1 timer.
VS: 0	Send count; range: 0 to 7.

The following line of output describes a frame timer event:

```
Serial3: SDLC T [C2] 12496064 CONNECT 12496064 0
```

[Table 219](#) describes the significant fields shown in the display.

Table 219 *debug sdlc Field Descriptions for a Timer Event*

Field	Description
Serial3:	Interface type and unit number reporting the frame event.
SDLC	Protocol providing the information.
T	Timer has expired.
[C2]	SDLC address of this SDLC connection.
12496064	System clock.

Table 219 *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"> • BOTHBUSY • CONNECT • DISCONNECT • DISCSENT (disconnect sent) • ERROR (FRMR frame sent) • REJSENT (reject frame sent) • SNRMSSENT (SNRM frame sent) • THEMBUSY • BOTHBUSY
12496064	Top timer.
0	Retry count; default: 0.

Related Commands

Command	Description
debug list	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 220](#) defines these bit flags.

Table 220 *debug sdlc local-ack Debugging Levels*

Debug Command	Meaning
debug sdlc local-ack 1	Only U-Frame events
debug sdlc local-ack 2	Only I-Frame events
debug sdlc local-ack 4	Only S-Frame events
debug sdlc local-ack 7	All 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:

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

Group of associated operations

```
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
```

S2560

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

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

[Table 221](#) describes the significant fields shown in the display.

Table 221 *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 sdllc

To display information about data link-layer frames transferred between a device on a Token Ring and a device on a serial line via a router configured with the SDLLC feature, use the **debug sdllc** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug sdllc

no debug sdllc

Syntax Description This command has no arguments or keywords.

Command Modes Privileged EXEC

Usage Guidelines The SDLLC feature translates between the SDLC link-layer protocol used to communicate with devices on a serial line and the LLC2 link-layer protocol used to communicate with devices on a Token Ring. The router configured with the SDLLC feature must be attached to the serial line. The router sends and receives frames on behalf of the serial device on the attached serial line but acts as an SDLC station. The topology between the router configured with the SDLLC feature and the Token Ring is network dependent and is not limited by the SDLLC feature.

Examples The following is sample output from the **debug sdllc** command between link-layer peers from the perspective of the SDLLC-configured router:

```
Router# debug sdllc

SDLLC: rx explorer rsp, da 4000.2000.1001, sa C000.1020.1000, rif
      8840.0011.00A1.0050
SDLLC: tx short xid, sa 4000.2000.1001, da C000.1020.1000, rif
      88C0.0011.00A1.0050, dsap 4 ssap 4
SDLLC: tx long xid, sa 4000.2000.1001, da C000.1020.1000, rif
      88C0.0011.00A1.0050, dsap 4 ssap 4
Rcvd SABME/LINKUP_REQ pak from TR host
SDLLCERR: not from our partner, pak dropped, da 4000.2000.1001,
sa C000.1020.1000, rif 8840.0011.00A1.0050, partner = 5000.1040.1003
```

[Table 222](#) describes the significant fields shown in the display.

Table 222 *debug sdllc Field Descriptions*

Field	Description
rx	Router receives message from the FEP.
explorer rsp	Response to an explorer (TEST) frame previously sent by the router to the FEP.
da	Destination address. This is the address of the router receiving the response.

Table 222 *debug sdllc Field Descriptions (continued)*

Field	Description
sa	Source address. This is the address of the FEP sending the response to the router.
rif	Routing information field (RIF).
tx	Router sent message to the FEP.
short xid	Router sent the null XID to the FEP.
dsap	Destination service access point
ssap	Source service access point.
tx long xid	Router sent the XID type 2 to the FEP.
Rcvd	Router received Layer 2 message from the FEP.
SABME/LINKUP_REQ	A set asynchronous Balanced Mode Extended command.
partner =	Partner address.

The following line indicates that an explorer frame response was received by the router at address 4000.2000.1001 from the FEP at address C000.1020.1000 with the specified RIF. The original explorer sent to the FEP from the router is not monitored as part of the **debug sdllc** command.

```
SDLLC: rx explorer rsp, da 4000.2000.1001, sa C000.1020.1000, rif
8840.0011.00A1.0050
```

The following line indicates that the router sent the null XID (Type 0) to the FEP. The debugging information does not include the response to the XID message sent by the FEP to the router.

```
SDLLC: tx short xid, sa 4000.2000.1001, da C000.1020.1000, rif
88C0.0011.00A1.0050, dsap 4 ssap 4
```

The following line indicates that the router sent the XID command (Format 0 Type 2) to the FEP:

```
SDLLC: tx long xid, sa 4000.2000.1001, da C000.1020.1000, rif
88C0.0011.00A1.0050, dsap 4 ssap 4
```

The following line is the SABME response to the XID command previously sent by the router to the FEP:

```
Rcvd SABME/LINKUP_REQ pak from TR host
```