debug ip pim atm

To log PIM ATM signalling activity, use the **debug ip pim atm** privileged EXEC command. The **no** form of this command disables debugging output.

debug ip pim atm

no debug ip pim atm

Syntax Description

This command has no arguments or keywords.

Examples

The following sample output shows a new group being created and the router toward the RP opening a new VC. Because there are now two groups on this router, there are two VCs open, as reflected by the "current count."

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

```
Router# debug ip pim atm

Jan 28 19:05:51: PIM-ATM: Max VCs 200, current count 1
Jan 28 19:05:51: PIM-ATM: Send SETUP on ATM2/0 for 239.254.254.253/171.69.214.43
Jan 28 19:05:51: PIM-ATM: Received CONNECT on ATM2/0 for 239.254.254.253, vcd 19
Jan 28 19:06:35: PIM-ATM: Max VCs 200, current count 2
```

Table 95 describes the significant fields in the output.

Table 95 debug ip pim atm Field Descriptions

Field	Description
Jan 28 19:05:51	Current date and time (in hours:minutes:seconds).
PIM-ATM	Indicates what PIM is doing to set up or monitor an ATM connection (vc).
current count	Current number of open virtual circuits.

The resulting **show ip mroute** output follows:

```
Router# show ip mroute 239.254.254.253

IP Multicast Routing Table
Flags: D - Dense, S - Sparse, C - Connected, L - Local, P - Pruned
       R - RP-bit set, F - Register flag, T - SPT-bit set, J - Join SPT
Timers: Uptime/Expires
Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 239.254.254.253), 00:00:04/00:02:53, RP 171.69.214.50, flags: S
  Incoming interface: Ethernet1/1, RPF nbr 171.69.214.50
  Outgoing interface list:
    ATM2/0, VCD 19, Forward/Sparse-Dense, 00:00:04/00:02:52
```

debug ip pim auto-rp

To display the contents of each Protocol Independent Multicast (PIM) packet used in the automatic discovery of group-to-rendezvous point (RP) mapping and the actions taken on the address-to-RP mapping database, use the **debug ip pim auto-rp** privileged EXEC command. The **no** form of this command disables debugging output.

debug ip pim auto-rp

no debug ip pim auto-rp

Syntax Description

This command has no arguments or keywords.

Examples

The following is sample output from the **debug ip pim auto-rp** command:

```
Router# debug ip pim auto-rp

Auto-RP: Received RP-announce, from 172.16.214.66, RP_cnt 1, holdtime 180 secs
Auto-RP:  update (192.168.248.0/24, RP:172.16.214.66)
Auto-RP: Build RP-Discovery packet
Auto-RP:  Build mapping (192.168.248.0/24, RP:172.16.214.66),
Auto-RP:  Build mapping (192.168.250.0/24, RP:172.16.214.26).
Auto-RP:  Build mapping (192.168.254.0/24, RP:172.16.214.2).
Auto-RP: Send RP-discovery packet (3 RP entries)
Auto-RP: Build RP-Announce packet for 172.16.214.2
Auto-RP:  Build announce entry for (192.168.254.0/24)
Auto-RP: Send RP-Announce packet, IP source 172.16.214.2, ttl 8
```

The first two lines show a packet received from 172.16.214.66 announcing that it is the RP for the groups in 192.168.248.0/24. This announcement contains one RP address and is valid for 180 seconds. The RP-mapping agent then updates its mapping database to include the new information.

```
Auto-RP: Received RP-announce, from 172.16.214.66, RP_cnt 1, holdtime 180 secs
Auto-RP:  update (192.168.248.0/24, RP:172.16.214.66)
```

In the next five lines, the router creates an RP-discovery packet containing three RP mapping entries. The packet is sent to the well-known CISCO-RP-DISCOVERY group address (224.0.1.40).

```
Auto-RP: Build RP-Discovery packet
Auto-RP:  Build mapping (192.168.248.0/24, RP:172.16.214.66),
Auto-RP:  Build mapping (192.168.250.0/24, RP:172.16.214.26).
Auto-RP:  Build mapping (192.168.254.0/24, RP:172.16.214.2).
Auto-RP: Send RP-discovery packet (3 RP entries)
```

The final three lines show the router announcing that it intends to be an RP for the groups in 192.168.254.0/24. Only routers inside the scope ttl 8 receive the advertisement and use the RP for these groups.

```
Auto-RP: Build RP-Announce packet for 172.16.214.2
Auto-RP:  Build announce entry for (192.168.254.0/24)
Auto-RP: Send RP-Announce packet, IP source 172.16.214.2, ttl 8
```

The following is sample output from the **debug ip pim auto-rp** command when a router receives an update. In this example, the packet contains three group-to-RP mappings, which are valid for 180 seconds. The RP-mapping agent then updates its mapping database to include the new information.

```
Router# debug ip pim auto-rp
```

```
Auto-RP: Received RP-discovery, from 172.16.214.17, RP_cnt 3, holdtime 180 secs
Auto-RP:  update (192.168.248.0/24, RP:172.16.214.66)
Auto-RP:  update (192.168.250.0/24, RP:172.16.214.26)
Auto-RP:  update (192.168.254.0/24, RP:172.16172.16.214.2)
```

debug ip policy

To display IP policy routing packet activity, use the **debug ip policy** privileged EXEC command. The **no** form of this command disables debugging output.

debug ip policy [*access-list-name*]

no debug ip policy [*access-list-name*]

Syntax Description

access-list-name	(Optional) The name of the access list. Displays packets permitted by the access list that are policy routed in process level, CEF, and DCEF (with NetFlow enabled or disabled). If no access list is specified, information about all policy-matched and policy-routed packets is displayed.

Command History

Release	Command
12.0(3)T	This command was introduced.

Usage Guidelines

After you configure IP policy routing with the **ip policy** and **route-map** commands, use the **debug ip policy** command to ensure that the IP policy is configured correctly.

Policy routing looks at various parts of the packet and then routes the packet based on certain user-defined attributes in the packet.

The **debug ip policy** command helps you determine what policy routing is following. It displays information about whether a packet matches the criteria, and if so, the resulting routing information for the packet.

Caution Because the **debug ip policy** command generates a substantial amount of output, use it only when traffic on the IP network is low, so other activity on the system is not adversely affected.

Examples

The following is sample output of the **debug ip policy** command:

```
Router# debug ip policy 3

IP: s=30.0.0.1 (Ethernet0/0/1), d=40.0.0.7, len 100,FIB flow policy match
IP: s=30.0.0.1 (Ethernet0/0/1), d=40.0.0.7, len 100,FIB PR flow accelerated!
IP: s=30.0.0.1 (Ethernet0/0/1), d=40.0.0.7, g=10.0.0.8, len 100, FIB policy routed
```

Table 96 describes the significant fields shown in the display.

Table 96 debug ip policy Field Descriptions

Field	Description
IP: s=	IP source address and interface of the packet being routed.
d=	IP destination address of the packet being routed.
len	Length of the packet.
g=	IP gateway address of the packet being routed.

debug ip rgmp

To log debug messages sent by an RGMP-enabled router, use the **debug ip rgmp** privileged EXEC command. To disable RGMP debugging, use the **no** form of this command.

debug ip rgmp [*group-name* | *group-address*]

no debug ip rgmp

Syntax Description

group-name	(Optional) The name of a specific IP multicast group.
group-address	(Optional) The IP address of a specific IP multicast group.

Defaults

Debugging for RGMP is not enabled. If the **debug ip rgmp** command is used without arguments, debugging is enabled for all RGMP message types.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(10)S	This command was introduced.
12.1(1)E	The command was integrated into Cisco IOS Release 12.1(1)E.
12.1(5)T	The command was integrated into Cisco IOS Release 12.1(5)T.

Examples

The following example shows output for the **debug ip rgmp** command:

```
Router# debug ip rgmp

   RGMP: Sending a Hello packet on Ethernet1/0

   RGMP: Sending a Join packet on Ethernet1/0 for group 224.1.2.3

   RGMP: Sending a Leave packet on Ethernet1/0 for group 224.1.2.3

   RGMP: Sending a Bye packet on Ethernet1/0
```

Related Commands

Command	Description
ip rgmp	Enables the RGMP on IEEE 802.3 Ethernet interfaces.
show ip igmp interface	Displays multicast-related information about an interface.

debug ip rip

To display information on RIP routing transactions, use the **debug ip rip** privileged EXEC command. The **no** form of this command disables debugging output.

debug ip rip

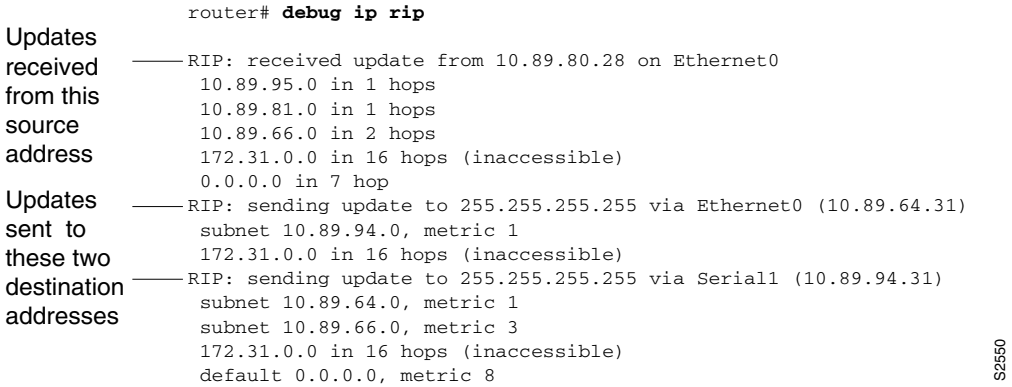
no debug ip rip

Syntax Description

This command has no arguments or keywords.

Examples

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



The output shows that the router being debugged has received updates from one router at source address 160.89.80.28. That router sent information about five destinations in the routing table update. Notice that the fourth destination address in the update—131.108.0.0—is inaccessible because it is more than 15 hops away from the router sending the update. The router being debugged also sent updates, in both cases to broadcast address 255.255.255.255 as the destination.

The second line is an example of a routing table update. It shows how many hops a given Internet address is from the router.

The entries show that the router is sending updates that are similar, except that the number in parentheses is the source address encapsulated into the IP header.

Examples of additional output that the **debug ip rip** command can generate follow.

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

```
RIP: broadcasting general request on Ethernet0
RIP: broadcasting general request on Ethernet1
```

An entry such as the following is most likely caused by a malformed packet from the sender:

```
RIP: bad version 128 from 160.89.80.43
```

debug ip routing

To display information on Routing Information Protocol (RIP) routing table updates and route cache updates, use the **debug ip routing** privileged EXEC command. The **no** form of this command disables debugging output.

debug ip routing

no debug ip routing

Syntax Description

This command has no arguments or keywords.

Examples

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

```
Router# debug ip routing

RT: add 172.25.168.0 255.255.255.0 via 172.24.76.30, igrp metric [100/3020]
RT: metric change to 172.25.168.0 via 172.24.76.30, igrp metric [100/3020]
        new metric [100/2930]
IP: cache invalidation from 0x115248 0x1378A, new version 5736
RT: add 172.26.219.0 255.255.255.0 via 172.24.76.30, igrp metric [100/16200]
RT: metric change to 172.26.219.0 via 172.24.76.30, igrp metric [100/16200]
        new metric [100/10816]
RT: delete route to 172.26.219.0 via 172.24.76.30, igrp metric [100/10816]
RT: no routes to 172.26.219.0, entering holddown
IP: cache invalidation from 0x115248 0x1378A, new version 5737
RT: 172.26.219.0 came out of holddown
RT: garbage collecting entry for 172.26.219.0
IP: cache invalidation from 0x115248 0x1378A, new version 5738
RT: add 172.26.219.0 255.255.255.0 via 172.24.76.30, igrp metric [100/10816]
RT: delete route to 172.26.219.0 via 172.24.76.30, igrp metric [100/10816]
RT: no routes to 172.26.219.0, entering holddown
IP: cache invalidation from 0x115248 0x1378A, new version 5739
RT: 172.26.219.0 came out of holddown
RT: garbage collecting entry for 172.26.219.0
IP: cache invalidation from 0x115248 0x1378A, new version 5740
RT: add 172.26.219.0 255.255.255.0 via 172.24.76.30, igrp metric [100/16200]
RT: metric change to 172.26.219.0 via 172.24.76.30, igrp metric [100/16200]
        new metric [100/10816]
RT: delete route to 172.26.219.0 via 172.24.76.30, igrp metric [100/10816]
RT: no routes to 172.26.219.0, entering holddown
IP: cache invalidation from 0x115248 0x1378A, new version 5741
```

In the following lines, a newly created entry has been added to the IP routing table. The “metric change” indicates that this entry existed previously, but its metric changed and the change was reported by means of IGRP. The metric could also be reported via RIP, OSPF, or another IP routing protocol. The numbers inside the brackets report the administrative distance and the actual metric.

```
RT: add 172.25.168.0 255.255.255.0 via 172.24.76.30, igrp metric [100/3020]
RT: metric change to 172.25.168.0 via 172.24.76.30, igrp metric [100/3020]
        new metric [100/2930]
IP: cache invalidation from 0x115248 0x1378A, new version 5736
```

“Cache invalidation” means that the fast-switching cache was invalidated due to a routing table change. “New version” is the version number of the routing table. When the routing table changes, this number is incriminated. The hexadecimal numbers are internal numbers that vary from version to version and software load to software load.

In the following output, the “holddown” and “cache invalidation” lines are displayed. Most of the distance vector routing protocols use “holddown” to avoid typical problems like counting to infinity and routing loops. If you look at the output of the **show ip protocols** command you will see the timer values for “holddown” and “cache invalidation.” “Cache invalidation” corresponds to “came out of holddown.” “Delete route” is triggered when a better path comes along. It removes the old inferior path.

```
RT: delete route to 172.26.219.0 via 172.24.76.30, igrp metric [100/10816]
RT: no routes to 172.26.219.0, entering holddown
IP: cache invalidation from 0x115248 0x1378A, new version 5737
RT: 172.26.219.0 came out of holddown
```

debug ip rsvp

To display information about Subnetwork Bandwidth Manager (SBM) message processing, the Designated Subnetwork Bandwidth Manager (DSBM) election process, and standard Resource Reservation Protocol (RSVP)-enabled message processing information, use the **debug ip rsvp** privileged EXEC command. To turn off debugging when you no longer want to display the output, use the **no** form of this command.

debug ip rsvp

no debug ip rsvp

Syntax Description

This command has no arguments or keywords.

Defaults

This command is disabled by default.

Usage Guidelines

The **debug ip rsvp** command provides information about messages received, minimal detail about the content of these messages, and information about state transitions. To obtain detailed information about RSVP and SBM, use the **debug ip rsvp detail** command.

Command History

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

Examples

The following example enables output of debug information about SBM message processing, the DSBM election process, and RSVP message processing information on router2:

```
Router# debug ip rsvp

RSVP debugging is on
router2#
*Dec 31 16:42:14.635: RSVP: send I_AM_DSBM message from 145.2.2.150
*Dec 31 16:42:14.635: RSVP: IP to 224.0.0.17 length=88 checksum=C788 Ethernet2)
*Dec 31 16:42:19.635: RSVP: send I_AM_DSBM message from 145.2.2.150
*Dec 31 16:42:19.635: RSVP: IP to 224.0.0.17 length=88 checksum=C788 (Ethernet2)
*Dec 31 16:42:20.823: RSVP: PATH message for 145.5.5.202(Ethernet2) from 145.2.2.1
*Dec 31 16:42:22.163: RSVP: send path multicast about 145.5.5.202 on Ethernet2
*Dec 31 16:42:22.163: RSVP: DSBM mgd segment - sending to ALLSBMADDRESS
*Dec 31 16:42:22.163: RSVP: IP to 224.0.0.17 length=212 checksum=DCAB (Ethernet2)
*Dec 31 16:42:23.955: RSVP: Sending RESV message for 145.5.5.202
*Dec 31 16:42:23.955: RSVP: send reservation to 145.2.2.1 about 145.5.5.202
*Dec 31 16:42:23.955: RSVP: IP to 145.2.2.1 length=108 checksum=1420 (Ethernet2)
*Dec 31 16:42:24.443: RSVP: RESV message for 145.5.5.202 (Ethernet2) from 145.2.2.2
*Dec 31 16:42:24.635: RSVP: send I_AM_DSBM message from 145.2.2.150
*Dec 31 16:42:24.635: RSVP: IP to 224.0.0.17 length=88 checksum=43AF (Ethernet2)
```

Related Commands

Command	Description
debug ip rsvp detail	Displays detailed information about RSVP and SBM.
debug ip rsvp sbm	Displays detailed information about the contents of SMB messages only, and SBM and DSBM state transitions.
ip rsvp dsbm-candidate	Configures an interface as a DSBM candidate.
show ip rsvp sbm	Displays information about SBM configured for a specific RSVP-enabled interface or all RSVP-enabled interfaces on the router.

debug ip rsvp detail

To display detailed information about Resource Reservation Protocol (RSVP)-enabled and Subnetwork Bandwidth Manager (SBM) message processing, use the **debug ip rsvp detail** privileged EXEC command. To turn off debugging when you no longer want to display the output, use the **no** form of this command.

debug ip rsvp detail

no debug ip rsvp detail

Syntax Description

This command has no arguments or keywords.

Defaults

This command is disabled by default.

Command History

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

Examples

The following example shows the detailed debug information about RSVP and SBM that is available when you enable debug mode through the **debug ip rsvp detail** command:

```
Router# debug ip rsvp detail

RSVP debugging is on
router2#u
*Dec 31 16:44:29.651: RSVP: send I_AM_DSBM message from 145.2.2.150
*Dec 31 16:44:29.651: RSVP: IP to 224.0.0.17 length=88 checksum=43AF
(Ethernet2)
*Dec 31 16:44:29.651: RSVP: version:1 flags:0000 type:I_AM_DSBM cksum:43AF
                       ttl:254 reserved:0 length:88
*Dec 31 16:44:29.651:  DSBM_IP_ADDR       type 1 length 8 : 91020296
*Dec 31 16:44:29.651:  HOP_L2             type 1 length 12: 00E01ECE
*Dec 31 16:44:29.651:                                     : 0F760000
*Dec 31 16:44:29.651:  SBM_PRIORITY       type 1 length 8 : 00000064
*Dec 31 16:44:29.651:  DSBM_TIMERS        type 1 length 8 : 00000F05
*Dec 31 16:44:29.651:  SBM_INFO           type 1 length 44: 00000000
*Dec 31 16:44:29.651:                                     : 00240C02 00000007
*Dec 31 16:44:29.651:                                     : 01000006 7F000005
*Dec 31 16:44:29.651:                                     : 00000000 00000000
*Dec 31 16:44:29.655:                                     : 00000000 00000000
*Dec 31 16:44:29.655:                                     : 00000000
```

Related Commands

Command	Description
debug ip rsvp	Displays information about SBM message processing, the DSBM election process, and RSVP message processing.
debug ip rsvp sbm	Displays detailed information about the contents of SMB messages only, and SBM and DSBM state transitions.
ip rsvp dsbm-candidate	Configures an interface as a DSBM candidate.
show ip rsvp sbm	Displays information about SBM configured for a specific RSVP-enabled interface or all RSVP-enabled interfaces on the router.

debug ip rsvp policy

To display debug messages for RSVP policy processing, use the **debug ip rsvp policy** privileged EXEC command. Use the **no** form of this command to disable debugging output.

debug ip rsvp policy

no debug ip rsvp policy

Syntax Description

This command has no arguments or keywords.

Defaults

Debugging for RSVP policy processing is not enabled.

Command History

Release	Modification
12.1(1)T	This command was introduced.

Usage Guidelines

You might find it useful to enable the **debug cops** command when you are using the **debug ip rsvp policy** command. Together, these commands generate a complete record of the policy process.

Examples

The following example uses only the **debug ip rsvp policy** command:

```
router-1# debug ip rsvp policy

RSVP_POLICY debugging is on

02:02:14:RSVP-POLICY:Creating outbound policy IDB entry for Ethernet2/0 (61E6AB38)
02:02:14:RSVP-COPS:COPS query for Path message, 10.31.0.1_44->10.33.0.1_44
02:02:14:RSVP-POLICY:Building incoming Path context
02:02:14:RSVP-POLICY:Building outgoing Path context on Ethernet2/0
02:02:14:RSVP-POLICY:Build REQ message of 216 bytes
02:02:14:RSVP-POLICY:Message sent to PDP
02:02:14:RSVP-COPS:COPS engine called us with reason2, handle 6202A658
02:02:14:RSVP-COPS:Received decision message
02:02:14:RSVP-POLICY:Received decision for Path message
02:02:14:RSVP-POLICY:Accept incoming message
02:02:14:RSVP-POLICY:Send outgoing message to Ethernet2/0
02:02:14:RSVP-POLICY:Replacement policy object for path-in context
02:02:14:RSVP-POLICY:Replacement TSPEC object for path-in context
02:02:14:RSVP-COPS:COPS report for Path message, 10.31.0.1_44->10.33.0.1_44
02:02:14:RSVP-POLICY:Report sent to PDP
02:02:14:RSVP-COPS:COPS report for Path message, 10.31.0.1_44->10.33.0.1_44
```

The following example uses both the **debug ip rsvp policy** and the **debug cops** commands:

```
router-1# debug ip rsvp policy

RSVP_POLICY debugging is on

router-1# debug cops
```

```
COPS debugging is on

02:15:14:RSVP-POLICY:Creating outbound policy IDB entry for Ethernet2/0 (61E6AB38)
02:15:14:RSVP-COPS:COPS query for Path message, 10.31.0.1_44->10.33.0.1_44
02:15:14:RSVP-POLICY:Building incoming Path context
02:15:14:RSVP-POLICY:Building outgoing Path context on Ethernet2/0
02:15:14:RSVP-POLICY:Build REQ message of 216 bytes
02:15:14:COPS:** SENDING MESSAGE **
    COPS HEADER:Version 1, Flags 0, Opcode 1 (REQ), Client-type:1, Length:216
    HANDLE (1/1) object. Length:8.    00 00 22 01
    CONTEXT (2/1) object. Length:8.   R-type:5.     M-type:1
    IN_IF (3/1) object. Length:12.   Address:10.1.2.1.    If_index:4
    OUT_IF (4/1) object. Length:12.   Address:10.33.0.1.    If_index:3
    CLIENT SI (9/1) object. Length:168.   CSI data:
02:15:14: SESSION              type 1 length 12:
02:15:14:      Destination 10.33.0.1, Protocol_Id 17, Don't Police , DstPort 44
02:15:14: HOP                  type 1 length 12:0A010201
02:15:14:                                     :00000000
02:15:14: TIME_VALUES          type 1 length 8 :00007530
02:15:14: SENDER_TEMPLATE      type 1 length 12:
02:15:14:      Source 10.31.0.1, udp_source_port 44
02:15:14: SENDER_TSPEC         type 2 length 36:
02:15:14:      version=0, length in words=7
02:15:14:      Token bucket fragment (service_id=1, length=6 words
02:15:14:          parameter id=127, flags=0, parameter length=5
02:15:14:          average rate=1250 bytes/sec, burst depth=10000 bytes
02:15:14:          peak rate   =1250000 bytes/sec
02:15:14:          min unit=0 bytes, max unit=1514 bytes
02:15:14: ADSPEC               type 2 length 84:
02:15:14: version=0  length in words=19
02:15:14: General Parameters  break bit=0  service length=8
02:15:14:                                          IS Hops:1
02:15:14:                 Minimum Path Bandwidth (bytes/sec):1250000
02:15:14:                        Path Latency (microseconds):0
02:15:14:                                         Path MTU:1500
02:15:14: Guaranteed Service  break bit=0  service length=8
02:15:14:                           Path Delay (microseconds):192000
02:15:14:                          Path Jitter (microseconds):1200
02:15:14:            Path delay since shaping (microseconds):192000
02:15:14:           Path Jitter since shaping (microseconds):1200
02:15:14: Controlled Load Service  break bit=0  service length=0
02:15:14:COPS:Sent 216 bytes on socket,
02:15:14:RSVP-POLICY:Message sent to PDP
02:15:14:COPS:Message event!
02:15:14:COPS:State of TCP is 4
02:15:14:In read function
02:15:14:COPS:Read block of 96 bytes, num=104 (len=104)
02:15:14:COPS:** RECEIVED MESSAGE **
    COPS HEADER:Version 1, Flags 1, Opcode 2 (DEC), Client-type:1, Length:104
    HANDLE (1/1) object. Length:8.    00 00 22 01
    CONTEXT (2/1) object. Length:8.   R-type:1.     M-type:1
    DECISION (6/1) object. Length:8.   COMMAND cmd:1, flags:0
    DECISION (6/3) object. Length:56.   REPLACEMENT  00 10 0E 01 61 62 63 64 65 66 67
    68 69 6A 6B 6C 00 24 0C 02 00
    00 00 07 01 00 00 06 7F 00 00 05 44 9C 40 00 46 1C 40 00 49 98
    96 80 00 00 00 C8 00 00 01 C8
    CONTEXT (2/1) object. Length:8.   R-type:4.     M-type:1
    DECISION (6/1) object. Length:8.   COMMAND cmd:1, flags:0

02:15:14:Notifying client (callback code 2)
02:15:14:RSVP-COPS:COPS engine called us with reason2, handle 6202A104
02:15:14:RSVP-COPS:Received decision message
02:15:14:RSVP-POLICY:Received decision for Path message
02:15:14:RSVP-POLICY:Accept incoming message
```

```
02:15:14:RSVP-POLICY:Send outgoing message to Ethernet2/0
02:15:14:RSVP-POLICY:Replacement policy object for path-in context
02:15:14:RSVP-POLICY:Replacement TSPEC object for path-in context
02:15:14:RSVP-COPS:COPS report for Path message, 10.31.0.1_44->10.33.0.1_44
02:15:14:COPS:** SENDING MESSAGE **
    COPS HEADER:Version 1, Flags 1, Opcode 3 (RPT), Client-type:1, Length:24
    HANDLE (1/1) object. Length:8.    00 00 22 01
    REPORT (12/1) object. Length:8.   REPORT type COMMIT (1)

02:15:14:COPS:Sent 24 bytes on socket,
02:15:14:RSVP-POLICY:Report sent to PDP
02:15:14:Timer for connection entry is zero
02:15:14:RSVP-COPS:COPS report for Path message, 10.31.0.1_44->10.33.0.1_44
```

Related Commands

Command	Description
debug cops	Displays debug messages for COPS processing.

debug ip rsvp sbm

To display detailed information about Subnetwork Bandwidth Manager (SBM) messages only, and SBM and Designated Subnetwork Bandwidth Manager (DSBM) state transitions, use the **debug ip rsvp sbm** privileged EXEC command. To turn off debugging when you no longer want to display the output, use the **no** form of this command.

debug ip rsvp sbm

no debug ip rsvp sbm

Syntax Description

This command has no arguments or keywords.

Defaults

This command is disabled by default.

Usage Guidelines

The **debug ip rsvp sbm** command provides information about messages received, minimal detail about the content of these messages, and information about state transitions.

Command History

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

Examples

The following example shows the detailed debug information about SBM and the SBM and DSBM state transitions that is available when you enable debug mode through the **debug ip rsvp sbm** command:

```
Router# debug ip rsvp sbm

RSVP debugging is on
router2#
*Dec 31 16:45:34.659: RSVP: send I_AM_DSBM message from 145.2.2.150
*Dec 31 16:45:34.659: RSVP: IP to 224.0.0.17 length=88 checksum=9385 (Ethernet2)
*Dec 31 16:45:34.659:  RSVP: version:1 flags:0000 type:I_AM_DSBM cksum:9385
                        ttl:254    reserved:0 length:88
*Dec 31 16:45:34.659:  DSBM_IP_ADDR       type 1 length 8 : 91020296
*Dec 31 16:45:34.659:  HOP_L2             type 1 length 12: 00E01ECE
*Dec 31 16:45:34.659:                                     : 0F760000
*Dec 31 16:45:34.659:  SBM_PRIORITY       type 1 length 8 : 0029B064
*Dec 31 16:45:34.659:  DSBM_TIMERS        type 1 length 8 : 00000F05
*Dec 31 16:45:34.659:  SBM_INFO           type 1 length 44: 00000000
*Dec 31 16:45:34.659:                                     : 00240C02 00000007
*Dec 31 16:45:34.659:                                     : 01000006 7F000005
*Dec 31 16:45:34.659:                                     : 00000000 00000000
*Dec 31 16:45:34.663:                                     : 00000000 00000000
*Dec 31 16:45:34.663:                                     : 00000000
*Dec 31 16:45:34.663:
```

Related Commands

Command	Description
debug ip rsvp	Displays information about SBM message processing, the DSBM election process, and RSVP message processing.
debug ip rsvp detail	Displays detailed information about RSVP and SBM
ip rsvp dsbm-candidate	Configures an interface as a DSBM candidate.

debug ip rsvp traffic-control

To display debug messages for traffic control, use the **debug ip rsvp traffic-control** EXEC command. To disable the **debug ip rsvp traffic-control** command, use the **no** form of this command.

debug ip rsvp traffic-control

no debug ip rsvp traffic-control

Syntax Description

This command has no arguments or keywords.

Defaults

No default behavior or values.

Command History

Release	Modification
12.0	This command was introduced.

Examples

The following is an example of output from the **debug ip rsvp traffic-control** command:

```
Router# debug ip rsvp traffic-control

RSVP debugging is on

Router# show debugging

IP RSVP debugging is on
IP RSVP debugging (Traffic Control events) is on
Router#
03:03:56:RSVP-TC:Attempting to remove QoS for rsb 6268A538
03:03:56:RSVP-TC:tcsb 00001A01 found for rsb 6268A538
03:03:56:RSVP-TC:Deleting tcsb 00001A01
03:04:15:RSVP-TC:Attempting to install QoS for rsb 6268A538
03:04:15:RSVP-TC:Adding new tcsb 00001E01 for rsb 6268A538
03:04:15:RSVP-TC:Assigning WFQ QoS to tcsb 00001E01
03:04:15:RSVP-TC:Consulting policy for tcsb 00001E01
03:04:15:RSVP-TC:Policy granted QoS for tcsb 00001E01
03:04:15:RSVP-TC:Requesting QoS for tcsb 00001E01
03:04:15:RSVP-TC:    ( r = 12500      bytes/s   M = 1514       bytes
03:04:15:RSVP-TC:      b = 1000       bytes     m = 0          bytes )
03:04:15:RSVP-TC:      p = 12500      bytes/s   Service Level = non-priority
03:04:15:RSVP-TC:Allocation succeeded for tcsb 00001E01
```

Related Commands

Command	Description
show debug	Displays active debug output.

debug ip rsvp wfq

To display debug messages for the weighted fair queue (WFQ), use the **debug ip rsvp wfq** EXEC command. To disable the command, use the **no** form of this command.

debug ip rsvp wfq

no debug ip rsvp wfq

Syntax Description

This command has no arguments or keywords.

Defaults

No default behavior or values.

Command History

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

Examples

The following is an example of output from the **debug ip rsvp wfq** command:

```
Router# debug ip rsvp wfq

RSVP debugging is on

Router# show debugging

IP RSVP debugging is on
IP RSVP debugging (Traffic Control events) is on
IP RSVP debugging (WFQ events) is on
Router#
03:03:23:RSVP-TC:Attempting to install QoS for rsb 6268A538
03:03:23:RSVP-TC:Adding new tcsb 00001A01 for rsb 6268A538
03:03:23:RSVP-TC:Assigning WFQ QoS to tcsb 00001A01
03:03:23:RSVP-TC:Consulting policy for tcsb 00001A01
03:03:23:RSVP-TC:Policy granted QoS for tcsb 00001A01
03:03:23:RSVP-TC:Requesting QoS for tcsb 00001A01
03:03:23:RSVP-TC:    ( r = 12500      bytes/s   M = 1514       bytes
03:03:23:RSVP-TC:      b = 1000       bytes     m = 0          bytes )
03:03:23:RSVP-TC:      p = 12500      bytes/s   Service Level = non-priority
03:03:23:RSVP-WFQ:Requesting a RESERVED queue on Et0/1 for tcsb 00001A01
03:03:23:RSVP-WFQ:Queue 265 allocated for tcsb 00001A01
03:03:23:RSVP-TC:Allocation succeeded for tcsb 00001A01
Router#

Router# no debug ip rsvp

RSVP debugging is off
```

Related Commands

Command	Description
show debug	Displays active debug output.

debug ip rtp header-compression

To display events specific to RTP header compression, use the **debug ip rtp header-compression** privileged EXEC command. Use the **no** form of this command to disable debugging output.

debug ip rtp header-compression

no debug ip rtp header-compression

Syntax Description

This command has no arguments or keywords.

Examples

The following is sample output from the **debug ip rtp header-compression** command:

```
Router# debug ip rtp header-compression

RHC BRI0: rcv compressed rtp packet
RHC BRI0: context0: expected sequence 0, received sequence 0
RHC BRI0: rcv compressed rtp packet
RHC BRI0: context0: expected sequence 1, received sequence 1
RHC BRI0: rcv compressed rtp packet
RHC BRI0: context0: expected sequence 2, received sequence 2
RHC BRI0: rcv compressed rtp packet
RHC BRI0: context0: expected sequence 3, received sequence 3
```

Table 97 describes the significant fields shown in the output.

Table 97 debug ip rtp header-compression Field Descriptions

Field	Description
context0	Compression state for a connection 0.
expected sequence	RTP header compression link sequence (expected).
received sequence	RTP header compression link sequence (actually received).

Related Commands

Command	Description
debug ip rtp packets	Displays a detailed dump of packets specific to RTP header compression.

debug ip rtp packets

To display a detailed dump of packets specific to RTP header compression, use the **debug ip rtp packets** privileged EXEC command. Use the **no** form of this command to disable debugging output.

debug ip rtp packets

no debug ip rtp packets

Syntax Description

This command has no arguments or keywords.

Examples

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

```
Router# debug ip rtp packets

RTP packet dump:
  IP:  source: 171.68.8.10, destination: 224.2.197.169, id: 0x249B, ttl: 9,
       TOS: 0 prot: 17,
  UDP: source port: 1034, destination port: 27404, checksum: 0xB429,len: 152
  RTP: version: 2, padding: 0, extension: 0, marker: 0,
       payload: 3, ssrc 2369713968,
       sequence: 2468, timestamp: 85187180, csrc count: 0
```

Table 98 describes the significant fields shown in the output.

Table 98 debug ip rtp packets Field Descriptions

Field	Description
id	IP identification.
ttl	IP time to live (TTL).
len	Total UDP length.

Related Commands

Command	Description
debug ip rtp header-compression	Displays events specific to RTP header compression.

debug ip sd

To display all session directory (SD) announcements received, use the **debug ip sd** privileged EXEC command. The **no** form of this command disables debugging output.

debug ip sd

no debug ip sd

Syntax Description

This command has no arguments or keywords.

Usage Guidelines

This command shows session directory announcements for multicast IP. Use it to observe multicast activity.

Examples

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

```
Router# debug ip sd

SD: Announcement from 172.16.58.81 on Serial0.1, 146 bytes
    s=*cisco: CBONE Audio
    i=cisco internal-only audio conference
    o=dino@dino-ss20.cisco.com
    c=224.0.255.1 16 2891478496 2892688096
    m=audio 31372 1700
SD: Announcement from 172.22.246.68 on Serial0.1, 147 bytes
    s=IMS: U.S. Senate
    i=U.S. Senate at http://town.hall.org/radio/live.html
    o=carl@also.radio.com
    c=224.2.252.231 95 0 0
    m=audio 36572 2642
    a=fmt:gsm
```

Table 99 describes the significant fields in the output.

Table 99 debug ip sd Output Descriptions

Field	Description
SD	Session directory event.
Announcement from	Address sending the SD announcement.
on Serial0.1	Interface receiving the announcement.
146 bytes	Size of the announcement event.
s=	Session name being advertised.
i=	Information providing a descriptive name for the session.
o=	Origin of the session, either an IP address or a name.
c=	Connect description showing address and number of hops.
m=	Media description that includes media type, port number, and ID.

Related Commands

Command	Description
debug ip dvmrp	Displays information on DVMRP packets received and sent.
debug ip igmp	Displays IGMP packets received and sent, and IGMP host-related events.
debug ip mbgp dampening	Logs route flap dampening activity related to MBGP.
debug ip mrouting	Displays changes to the IP multicast routing table.
debug ip pim	Displays PIM packets received and sent, and PIM-related events.

debug ip security

To display IP security option processing, use the **debug ip security** privileged EXEC command. The **no** form of this command disables debugging output.

debug ip security

no debug ip security

Syntax Description

This command has no arguments or keywords.

Usage Guidelines

The **debug ip security** command displays information for both basic and extended IP security options. For interfaces where **ip security** is configured, each IP packet processed for that interface results in debugging output regardless of whether the packet contains IP security options. IP packets processed for other interfaces that also contain IP security information also trigger debugging output. Some additional IP security debugging information is also controlled by the **debug ip packet** privileged EXEC command.

Caution Because the **debug ip security** command generates a substantial amount of output for every IP packet processed, use it only when traffic on the IP network is low, so other activity on the system is not adversely affected.

Examples

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

```
Router# debug ip security

IP Security: src 172.24.72.52 dst 172.24.72.53, number of BSO 1
     idb: NULL
     pak: insert (0xFF) 0x0
IP Security: BSO postroute: SECINSERT changed to secret (0x5A) 0x10
IP Security: src 172.24.72.53 dst 172.24.72.52, number of BSO 1
     idb: secret (0x6) 0x10 to secret (0x6) 0x10, no implicit
          def secret (0x6) 0x10
     pak: secret (0x5A) 0x10
IP Security: checking BSO 0x10 against [0x10 0x10]
IP Security: classified BSO as secret (0x5A) 0x10
```

Table 100 describes significant fields shown in the output.

Table 100 debug ip security Field Descriptions

Field	Description
number of BSO	Indicates the number of basic security options found in the packet.
idb	Provides information on the security configuration for the incoming interface.
pak	Provides information on the security classification of the incoming packet.
src	Indicates the source IP address.
dst	Indicates the destination IP address.

The following line indicates that the packet was locally generated, and it has been classified with the internally significant security level “insert” (0xff) and authority information of 0x0:

```
idb: NULL
pak: insert (0xff) 0x0
```

The following line indicates that the packet was received via an interface with dedicated IP security configured. Specifically, the interface is configured at security level “secret” and with authority information of 0x0. The packet itself was classified at level “secret” (0x5a) and authority information of 0x10.

```
idb: secret (0x6) 0x10 to secret (0x6) 0x10, no implicit
     def secret (0x6) 0x10
pak: secret (0x5A) 0x10
```

debug ip slb

To display debug messages for the Cisco IOS Server Load Balancing (SLB) feature, use the **debug ip slb** EXEC command. To stop debug output, use the **no** form of this command.

debug ip slb {**conns** | **dfp** | **icmp** | **reals** | **all**}

no debug ip slb {**conns** | **dfp** | **icmp** | **reals** | **all**}

Syntax Description

conns	Displays debug messages for all connections being handled by Cisco IOS SLB.
dfp	Displays debug messages for the Cisco IOS SLB DFP and DFP agents.
icmp	Displays all Internet Control Message Protocol (ICMP) debug messages for Cisco IOS SLB.
reals	Displays debug messages for all real servers defined to Cisco IOS SLB.
all	Displays all debug messages for Cisco IOS SLB.

Defaults

No default behavior or values.

Command History

Release	Modification
12.0(7)XE	This command was introduced.
12.1(5)T	This command was integrated into Cisco IOS Release 12.1(5)T.

Usage Guidelines

See the following caution before using **debug** commands.

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

Examples

The following example configures a debug session to check all IP IOS SLB parameters:

```
Router# debug ip slb all

SLB All debugging is on
Router#
```

The following example stops all debugging:

```
Router# no debug all

All possible debugging has been turned off
Router#
```

The following example shows Cisco IOS SLB DFP debug output:

```
router# debug ip slb dfp

SLB DFP debugging is on
router#
022048 SLB DFP Queue to main queue - type 2 for Agent 161.44.2.3458229
022048 SLB DFP              select_rc = -1   readset = 0
022048 SLB DFP       Sleeping ...
022049 SLB DFP              readset = 0
022049 SLB DFP              select_rc = -1   readset = 0
022049 SLB DFP Processing Q event for Agent 161.44.2.3458229 - OPEN
022049 SLB DFP Queue to conn_proc_q - type 2 for Agent 161.44.2.3458229
022049 SLB DFP              readset = 0
022049 SLB DFP Set SLB_DFP_SIDE_QUEUE
022049 SLB DFP Processing Conn Q event for Agent 161.44.2.3458229 - OPEN
022049 SLB DFP Open to Agent 161.44.2.3458229 succeeded, socket = 0
022049 SLB DFP Agent 161.44.2.3458229 start connect
022049 SLB DFP Connect to Agent 161.44.2.3458229 successful - socket 0


022049 SLB DFP Queue to main queue - type 6 for Agent 161.44.2.3458229
022049 SLB DFP Processing Conn Q unknown MAJOR 80
022049 SLB DFP Reset SLB_DFP_SIDE_QUEUE
022049 SLB DFP              select_rc = -1   readset = 0
022049 SLB DFP       Sleeping ...
022050 SLB DFP              readset = 1
022050 SLB DFP              select_rc = 1   readset = 1
022050 SLB DFP Agent 161.44.2.3458229 fd = 0 readset = 1
022050 SLB DFP Message length 44 from Agent 161.44.2.3458229
022050 SLB DFP Agent 161.44.2.3458229 setting Host 17.17.17.17, Bind ID 1 Weight 1
022050 SLB DFP Agent 161.44.2.3458229 setting Host 34.34.34.34, Bind ID 2 Weight 2
022050 SLB DFP Agent 161.44.2.3458229 setting Host 51.51.51.51, Bind ID 3 Weight 3
022050 SLB DFP Processing Q event for Agent 161.44.2.3458229 - WAKEUP


022050 SLB DFP              readset = 1
022050 SLB DFP              select_rc = 1   readset = 1
022050 SLB DFP Agent 161.44.2.3458229 fd = 0 readset = 1
022050 SLB DFP Message length 64 from Agent 161.44.2.3458229
022050 SLB DFP Agent 161.44.2.3458229 setting Host 17.17.17.17, Bind ID 1 Weight 1
022050 SLB DFP Agent 161.44.2.3458229 setting Host 68.68.68.68, Bind ID 4 Weight 4
022050 SLB DFP Agent 161.44.2.3458229 setting Host 85.85.85.85, Bind ID 5 Weight 5
022050 SLB DFP Agent 161.44.2.3458229 setting Host 17.17.17.17, Bind ID 111 Weight 111
022050 SLB DFP              readset = 1


022115 SLB DFP Queue to main queue - type 5 for Agent 161.44.2.3458229
022115 SLB DFP              select_rc = -1   readset = 0
022115 SLB DFP       Sleeping ...
022116 SLB DFP              readset = 1
022116 SLB DFP              select_rc = -1   readset = 0
022116 SLB DFP Processing Q event for Agent 161.44.2.3458229 - DELETE


022116 SLB DFP Queue to conn_proc_q - type 5 for Agent 161.44.2.3458229
022116 SLB DFP              readset = 1
022116 SLB DFP Set SLB_DFP_SIDE_QUEUE
```

```
022116 SLB DFP Processing Conn Q event for Agent 161.44.2.3458229 - DELETE
022116 SLB DFP Connection to Agent 161.44.2.3458229 closed
022116 SLB DFP Agent 161.44.2.3458229 deleted
022116 SLB DFP Processing Conn Q unknown MAJOR 80
022116 SLB DFP Reset SLB_DFP_SIDE_QUEUE
022116 SLB DFP Set SLB_DFP_SIDE_QUEUE
022116 SLB DFP Reset SLB_DFP_SIDE_QUEUE
```

debug ip socket

To display all state change information for all sockets, use the **debug ip socket** privileged EXEC command. Use the **no** form of this command to disable debugging output.

debug ip socket

no debug ip socket

Syntax Description

This command has no arguments or keywords.

Usage Guidelines

Use this command to collect information on the socket interface. To get more complete information on a socket/TCP port pair, use this command in conjunction with the **debug ip tcp transactions** command.

Because the socket debugging information is state change oriented, you will not see the debug message on a per-packet basis. However, if the connections normally have very short lives (few packet exchanges during the life cycle of a connection), then socket debugging could become expensive because of the state changes involved during connection setup and teardown.

Examples

The following is sample output from the **debug ip socket** output from a server process:

```
Router# debug ip socket

Added socket 0x60B86228 to process 40
SOCKET: set TCP property TCP_PID, socket 0x60B86228, TCB 0x60B85E38
Accepted new socket fd 1, TCB 0x60B85E38
Added socket 0x60B86798 to process 40
SOCKET: set TCP property TCP_PID, socket 0x60B86798, TCB 0x60B877C0
SOCKET: set TCP property TCP_BIT_NOTIFY, socket 0x60B86798, TCB 0x60B877C0
SOCKET: created new socket to TCP, fd 2, TCB 0x60B877C0
SOCKET: bound socket fd 2 to TCB 0x60B877C0
SOCKET: set TCP property TCP_WINDOW_SIZE, socket 0x60B86798, TCB 0x60B877C0
SOCKET: listen on socket fd 2, TCB 0x60B877C0
SOCKET: closing socket 0x60B86228, TCB 0x60B85E38
SOCKET: socket event process: socket 0x60B86228, TCB new state --> FINWAIT1
socket state: SS_ISCONNECTED SS_CANTSENDMORE SS_ISDISCONNECTING
SOCKET: Removed socket 0x60B86228 from process 40 socket list
```

The following is sample output from the **debug ip socket** command from a client process:

```
Router# debug ip socket

Added socket 0x60B70220 to process 2
SOCKET: set TCP property TCP_PID, socket 0x60B70220, TCB 0x60B6CFDC
SOCKET: set TCP property TCP_BIT_NOTIFY, socket 0x60B70220, TCB 0x60B6CFDC
SOCKET: created new socket to TCP, fd 0, TCB 0x60B6CFDC
SOCKET: socket event process: socket 0x60B70220, TCB new state --> SYNSENT
socket state: SS_ISCONNECTING
SOCKET: socket event process: socket 0x60B70220, TCB new state --> ESTAB
socket state: SS_ISCONNECTING
SOCKET: closing socket 0x60B70220, TCB 0x60B6CFDC
SOCKET: socket event process: socket 0x60B70220, TCB new state --> FINWAIT1
socket state: SS_ISCONNECTED SS_CANTSENDMORE SS_ISDISCONNECTING
SOCKET: Removed socket 0x60B70220 from process 2 socket list
```

Table 101 describes the significant fields shown in the display.

Table 101 debug ip socket Field Descriptions

Field	Description
Added socket 0x60B86228 process 40	New socket is opened for process 40.
SOCKET	Indicates that this is a SOCKET transaction.
set TCP property TCP_PID	Sets the process ID to the TCP associated with the socket.
socket 0x60B86228, TCB 0x60B85E38	Address for the socket/TCP pair.
set TCP property TCP_BIT_NOTIFY	Sets the method for how the socket wants to be notified for an event.
created new socket to TCP, fd 2	Opened a new socket referenced by file descriptor 2 to TCP.
bound socket fd 2 to TCB	Bound the socket referenced by file descriptor 2 to TCP.
listen on socket fd 2	Indicates which file descriptor the application is listening to.
closing socket	Indicates that the socket is being closed.
socket event process	Processed a state change event occurred in the transport layer.
TCB new state --> FINWAIT1	TCP state machine changed to FINWAIT1. (See the **debug ip tcp transaction** command for more information on TCP state machines.)

Table 101 debug ip socket Field Descriptions (continued)

Field	Description
socket state: SS_ISCONNECTED SS_CANTSENDMORE SS_ISDISCONNECTING	New SOCKET state flags after the transport event processing. This socket is still connected, but disconnecting is in progress, and it will not send more data to peer. Possible SOCKET state flags follow: • SS_NOFDREF No file descriptor reference for this socket. • SS_ISCONNECTING Socket connecting is in progress. • SS_ISBOUND Socket is bound to TCP. • SS_ISCONNECTED Socket is connected to peer. • SS_ISDISCONNECTING Socket disconnecting is in progress. • SS_CANTSENDMORE Can't send more data to peer. • SS_CANTRCVMORE Can't receive more data from peer. • SS_ISDISCONNECTED Socket is disconnected. Connection is fully closed.
Removed socket 0x60B86228 from process 40 socket list	Connection is closed, and the socket is removed from the process socket list.

Related Commands

Command	Description
debug ip tcp transactions	Displays information on significant TCP transactions such as state changes, retransmissions, and duplicate packets.

debug ip ssh

To display debug messages for Secure Shell (SSH), use the **debug ip ssh** EXEC command. To disable debugging output, use the **no** form of the command.

debug ip ssh

no debug ip ssh

Syntax Description

This command has no arguments or keywords.

Defaults

Debugging for SSH is not enabled.

Command History

Release	Modification
12.0(5)S	This command was introduced.
12.1(1)T	This command was integrated into Cisco IOS Release 12.1 T.

Usage Guidelines

Use the **debug ssh** command to ensure normal operation of the SSH server.

Examples

The following example shows the SSH debugging output:

```
Router# debug ssh

00:53:46: SSH0: starting SSH control process
00:53:46: SSH0: Exchanging versions - SSH-1.5-Cisco-1.25

00:53:46: SSH0: client version is - SSH-1.5-1.2.25
00:53:46: SSH0: SSH_SMSG_PUBLIC_KEY message sent
00:53:46: SSH0: SSH_CMSG_SESSION_KEY message received
00:53:47: SSH0: keys exchanged and encryption on
00:53:47: SSH0: authentication request for userid guest
00:53:47: SSH0: authentication successful for jcisco
00:53:47: SSH0: starting exec shell
```

debug ip tcp driver

To display information on TCP driver events; for example, connections opening or closing, or packets being dropped because of full queues, use the **debug ip tcp driver** privileged EXEC command. The **no** form of this command disables debugging output.

debug ip tcp driver

no debug ip tcp driver

Syntax Description

This command has no arguments or keywords.

Usage Guidelines

The TCP driver is the process that the router software uses to send packet data over a TCP connection. Remote source-route bridging (RSRB), serial tunneling (STUN), and X.25 switching currently use the TCP driver.

Using the **debug ip tcp driver** command together with the **debug ip tcp driver-pak** command provides the most verbose debugging output concerning TCP driver activity.

Examples

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

```
Router# debug ip tcp driver

TCPDRV359CD8: Active open 172.21.80.26:0 --> 172.21.80.25:1996 OK, lport 36628
TCPDRV359CD8: enable tcp timeouts
TCPDRV359CD8: 172.21.80.26:36628 --> 172.21.80.25:1996 Abort
TCPDRV359CD8: 172.21.80.26:36628 --> 172.21.80.25:1996 DoClose tcp abort
```

Table 102 describes the significant fields shown in the display.

Table 102 debug ip tcp driver Field Descriptions

Field	Description
TCPDRV359CD8:	Unique identifier for this instance of TCP driver activity.
Active open 172.21.80.26	Indication that the router at IP address 172.21.80.26 has initiated a connection to another router.
:0	TCP port number the initiator of the connection uses to indicate that any port number can be used to set up a connection.
--> 172.21.80.25	IP address of the remote router to which the connection has been initiated.
:1996	TCP port number that the initiator of the connection is requesting that the remote router use for the connection. (1996 is a private TCP port number reserved in this implementation for RSRB.)
OK,	Indication that the connection has been established. If the connection has not been established, this field and the following field do not appear in this line of output.
lport 36628	TCP port number that has actually been assigned for the initiator to use for this connection.

The following line indicates that the TCP driver user (RSRB, in this case) will allow TCP to drop the connection if excessive retransmissions occur:

```
TCPDRV359CD8: enable tcp timeouts
```

The following line indicates that the TCP driver user (in this case, RSRB) at IP address 172.21.80.26 (and using TCP port number 36628) is requesting that the connection to IP address 172.21.80.25 using TCP port number 1996 be aborted:

```
TCPDRV359CD8: 172.21.80.26:36628 --> 172.21.80.25:1996 Abort
```

The following line indicates that this connection was in fact closed due to an abnormal termination:

```
TCPDRV359CD8: 172.21.80.26:36628 --> 172.21.80.25:1996 DoClose tcp abort
```

debug ip tcp driver-pak

To display information on every operation that the TCP driver performs, use the **debug ip tcp driver-pak** privileged EXEC command. The **no** form of this command disables debugging output.

debug ip tcp driver-pak

no debug ip tcp driver-pak

Syntax Description

This command has no arguments or keywords.

Usage Guidelines

This command turns on a verbose debugging by logging at least one debugging message for every packet sent or received on the TCP driver connection.

The TCP driver is the process that the router software uses to send packet data over a TCP connection. RSRB, serial tunneling (STUN), and X.25 switching currently use the TCP driver.

To observe the context within which certain **debug ip tcp driver-pak** messages occur, turn on this command in conjunction with the **debug ip tcp driver** command.

Caution Because the **debug ip tcp driver-pak** command generates so many messages, use it only on lightly loaded systems. This command not only places a substantial load on the system processor, it also may change the symptoms of any unexpected behavior that occurs.

Examples

The following is sample output from the **debug ip tcp driver-pak** command:

```
Router# debug ip tcp driver-pak

TCPDRV359CD8: send 2E8CD8 (len 26) queued
TCPDRV359CD8: output pak 2E8CD8 (len 26) (26)
TCPDRV359CD8: readf 42 bytes (Thresh 16)
TCPDRV359CD8: readf 26 bytes (Thresh 16)
TCPDRV359CD8: readf 10 bytes (Thresh 10)
TCPDRV359CD8: send 327E40 (len 4502) queued
TCPDRV359CD8: output pak 327E40 (len 4502) (4502)
```

Table 103 describes the significant fields shown in the display.

Table 103 debug ip tcp driver-pak Field Descriptions

Field	Description
TCPDRV359CD8	Unique identifier for this instance of TCP driver activity.
send	Indicates that this event involves the TCP driver sending data.
2E8CD8	Address in memory of the data the TCP driver is sending.
(len 26)	Length of the data (in bytes).
queued	Indicates that the TCP driver user process (in this case, RSRB) has transferred the data to the TCP driver to send.

The following line indicates that the TCP driver has sent the data that it had received from the TCP driver user, as shown in the previous line of output. The last field in the line (26) indicates that the 26 bytes of data were sent out as a single unit.

```
TCPDRV359CD8: output pak 2E8CD8 (len 26) (26)
```

The following line indicates that the TCP driver has received 42 bytes of data from the remote IP address. The TCP driver user (in this case, remote source-route bridging) has established an input threshold of 16 bytes for this connection. (The input threshold instructs the TCP driver to transfer data to the TCP driver user only when at least 16 bytes are present.)

```
TCPDRV359CD8: readf 42 bytes (Thresh 16)
```

debug ip tcp intercept

To display TCP intercept statistics, use the **debug ip tcp intercept** privileged EXEC command. The **no** form of this command disables debugging output.

debug ip tcp intercept

no debug ip tcp intercept

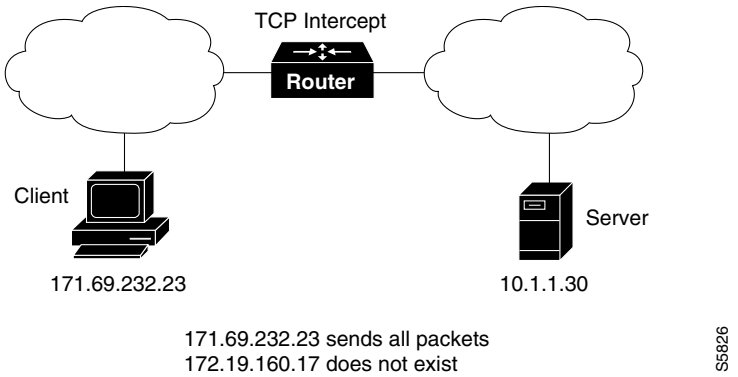
Syntax Description

This command has no arguments or keywords.

Examples

Figure 4 illustrates a scenario in which a router configured with TCP intercept operates between a client and a server.

Figure 4 Example TCP Intercept Environment



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

```
Router# debug ip tcp intercept
```

A connection attempt arrives:

```
INTERCEPT: new connection (172.19.160.17:61774) => (10.1.1.30:23)
INTERCEPT: 172.19.160.17:61774 <- ACK+SYN (10.1.1.30:61774)
```

A second connection attempt arrives:

```
INTERCEPT: new connection (172.19.160.17:62030) => (10.1.1.30:23)
INTERCEPT: 172.19.160.17:62030 <- ACK+SYN (10.1.1.30:62030)
```

The router re-sends to both apparent clients:

```
INTERCEPT: retransmit 2 (172.19.160.17:61774) <- (10.1.1.30:23) SYNRCVD
INTERCEPT: retransmit 2 (172.19.160.17:62030) <- (10.1.1.30:23) SYNRCVD
```

A third connection attempt arrives:

```
INTERCEPT: new connection (171.69.232.23:1048) => (10.1.1.30:23)
INTERCEPT: 171.69.232.23:1048 <- ACK+SYN (10.1.1.30:1048)
```

The router sends more retransmissions trying to establish connections with the apparent clients:

```
INTERCEPT: retransmit 4 (172.19.160.17:61774) <- (10.1.1.30:23) SYNRCVD
INTERCEPT: retransmit 4 (172.19.160.17:62030) <- (10.1.1.30:23) SYNRCVD
INTERCEPT: retransmit 2 (171.69.232.23:1048) <- (10.1.1.30:23) SYNRCVD
```

The router establishes the connection with the third client and re-sends to the server:

```
INTERCEPT: 1st half of connection is established (171.69.232.23:1048) => (10.1.1.30:23)
INTERCEPT: (171.69.232.23:1048) SYN -> 10.1.1.30:23
INTERCEPT: retransmit 2 (171.69.232.23:1048) -> (10.1.1.30:23) SYNSENT
```

The server responds; the connection is established:

```
INTERCEPT: 2nd half of connection established (171.69.232.23:1048) => (10.1.1.30:23)
INTERCEPT: (171.69.232.23:1048) ACK -> 10.1.1.30:23
```

The router re-sends to the first two apparent clients, times out, and sends resets:

```
INTERCEPT: retransmit 8 (172.19.160.17:61774) <- (10.1.1.30:23) SYNRCVD
INTERCEPT: retransmit 8 (172.19.160.17:62030) <- (10.1.1.30:23) SYNRCVD
INTERCEPT: retransmit 16 (172.19.160.17:61774) <- (10.1.1.30:23) SYNRCVD
INTERCEPT: retransmit 16 (172.19.160.17:62030) <- (10.1.1.30:23) SYNRCVD
INTERCEPT: retransmitting too long (172.19.160.17:61774) => (10.1.1.30:23) SYNRCVD
INTERCEPT: 172.19.160.17:61774 <- RST (10.1.1.30:23)
INTERCEPT: retransmitting too long (172.19.160.17:62030) => (10.1.1.30:23) SYNRCVD
INTERCEPT: 172.19.160.17:62030 <- RST (10.1.1.30:23)
```

debug ip tcp transactions

To display information on significant TCP transactions such as state changes, retransmissions, and duplicate packets, use the **debug ip tcp transactions** privileged EXEC command. The **no** form of this command disables debugging output.

debug ip tcp transactions

no debug ip tcp transactions

Syntax Description

This command has no arguments or keywords.

Usage Guidelines

This command is particularly useful for debugging a performance problem on a TCP/IP network that you have isolated above the data link layer.

The **debug ip tcp transactions** command displays output for packets the router sends and receives, but does not display output for packets it forwards.

Examples

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

```
Router# debug ip tcp transactions

TCP: sending SYN, seq 168108, ack 88655553
TCP0: Connection to 10.9.0.13:22530, advertising MSS 966
TCP0: state was LISTEN -> SYNRCVD [23 -> 10.9.0.13(22530)]
TCP0: state was SYNSENT -> SYNRCVD [23 -> 10.9.0.13(22530)]
TCP0: Connection to 10.9.0.13:22530, received MSS 956
TCP0: restart retransmission in 5996
TCP0: state was SYNRCVD -> ESTAB [23 -> 10.9.0.13(22530)]
TCP2: restart retransmission in 10689
TCP2: restart retransmission in 10641
TCP2: restart retransmission in 10633
TCP2: restart retransmission in 13384 -> 10.0.0.13(16151)]
TCP0: restart retransmission in 5996 [23 -> 10.0.0.13(16151)]
```

Table 104 describes the significant fields shown in the display.

Table 104 debug ip tcp transactions Field Descriptions

Field	Description
TCP:	Indicates that this is a TCP transaction.
sending SYN	Indicates that a synchronize packet is being sent.
seq 168108	Indicates the sequence number of the data being sent.
ack 88655553	Indicates the sequence number of the data being acknowledged.
TCP0:	Indicates the TTY number (0, in this case) with which this TCP connection is associated.
Connection to 10.9.0.13:22530	Indicates the remote address with which a connection has been established.

Table 104 debug ip tcp transactions Field Descriptions (continued)

Field	Description
advertising MSS 966	Indicates the maximum segment size this side of the TCP connection is offering to the other side.
state was LISTEN -> SYNRCVD	Indicates that the TCP state machine changed state from LISTEN to SYNSENT. Possible TCP states follow: • CLOSED—Connection closed. • CLOSEWAIT—Received a FIN segment. • CLOSING—Received a FIN/ACK segment. • ESTAB—Connection established. • FINWAIT 1—Sent a FIN segment to start closing the connection. • FINWAIT 2—Waiting for a FIN segment. • LASTACK—Sent a FIN segment in response to a received FIN segment. • LISTEN—Listening for a connection request. • SYNRCVD—Received a SYN segment, and responded. • SYNSENT—Sent a SYN segment to start connection negotiation. • TIMEWAIT—Waiting for network to clear segments for this connection before the network no longer recognizes the connection as valid. This must occur before a new connection can be set up.
[23 -> 10.9.0.13(22530)]	The element within these brackets are as follows: • The first field (23) indicates local TCP port. • The second field (10.9.0.13) indicates the destination IP address. • The third field (22530) indicates the destination TCP port.
restart retransmission in 5996	Indicates the number of milliseconds until the next retransmission takes place.

debug ip trigger-authentication

To display information related to automated double authentication, use the **debug ip trigger-authentication** privileged EXEC command. The **no** form of this command disables debugging output.

debug ip trigger-authentication [**verbose**]

no debug ip trigger-authentication [**verbose**]

Syntax Description

verbose	(Optional) Specifies that the complete debugging output be displayed, including information about packets that are blocked before authentication is complete.

Usage Guidelines

Use this command when troubleshooting automated double authentication.

This command displays information about the remote host table. Whenever entries are added, updated, or removed, a new debugging message is displayed.

What is the remote host table? Whenever a remote user needs to be user-authenticated in the second stage of automated double authentication, the local device sends a UDP packet to the host of the remote user. Whenever such a UDP packet is sent, the host IP address of the user is added to a table. If additional UDP packets are sent to the same remote host, a new table entry is not created; instead, the existing entry is updated with a new time stamp. This remote host table contains a cumulative list of host entries; entries are deleted after a timeout period or after you manually clear the table using the **clear ip trigger-authentication** command.

If you include the **verbose** keyword, the debugging output also includes information about packet activity.

Examples

The following is sample output from the **debug ip trigger-authentication** command. In this example, the local device at 172.21.127.186 sends a UDP packet to the remote host at 172.21.127.114. The UDP packet is sent to request the remote user's username and password (or PIN). (The output indicates "New entry added.")

After a timeout period, the local device has not received a valid response from the remote host, so the local device sends another UDP packet. (The output indicates "Time stamp updated.")

Then the remote user is authenticated, and after a length of time (the timeout period) the entry is removed from the remote host table. (The output indicates "remove obsolete entry.")

```
myfirewall# debug ip trigger-authentication

TRIGGER_AUTH: UDP sent from 172.21.127.186 to 172.21.127.114, qdata=7C2504
              New entry added, timestamp=2940514234
TRIGGER_AUTH: UDP sent from 172.21.127.186 to 172.21.127.114, qdata=7C2504
              Time stamp updated, timestamp=2940514307
TRIGGER_AUTH: remove obsolete entry, remote host=172.21.127.114
```

The following is sample output from the **debug ip trigger-authentication verbose** command. In this example, messages about packet activity are included because of the use of the **verbose** keyword.

You can see many packets that are being blocked at the interface because the user has not yet been double authenticated. These packets will be permitted through the interface only after the user has been double authenticated. (You can see packets being blocked when the output indicates “packet enqueued” then “packet ignored.”)

```
TRIGGER_AUTH: packet enqueued, qdata=69FEEC
              remote host=172.21.127.113, local host=172.21.127.186 (if: 0.0.0.0)
TRIGGER_AUTH: UDP sent from 172.21.127.186 to 172.21.127.113, qdata=69FEEC
              Time stamp updated
TRIGGER_AUTH: packet enqueued, qdata=69FEEC
              remote host=172.21.127.113, local host=172.21.127.186 (if: 0.0.0.0)
TRIGGER_AUTH: packet ignored, qdata=69FEEC
TRIGGER_AUTH: packet enqueued, qdata=69FEEC
              remote host=172.21.127.113, local host=172.21.127.186 (if: 0.0.0.0)
TRIGGER_AUTH: packet ignored, qdata=69FEEC
TRIGGER_AUTH: packet enqueued, qdata=69FEEC
              remote host=172.21.127.113, local host=172.21.127.186 (if: 0.0.0.0)
TRIGGER_AUTH: UDP sent from 172.21.127.186 to 172.21.127.113, qdata=69FEEC
              Time stamp updated
TRIGGER_AUTH: packet enqueued, qdata=69FEEC
              remote host=172.21.127.113, local host=172.21.127.186 (if: 0.0.0.0)
TRIGGER_AUTH: packet ignored, qdata=69FEEC
TRIGGER_AUTH: packet enqueued, qdata=69FEEC
              remote host=172.21.127.113, local host=172.21.127.186 (if: 0.0.0.0)
TRIGGER_AUTH: packet ignored, qdata=69FEEC
```

debug ip udp

To enable logging of User Datagram Protocol (UDP) packets sent and received, use the **debug ip udp** privileged EXEC command. To disable debugging output, use the **no** form of this command.

debug ip udp

no debug ip udp

Syntax Description

This command has no arguments or keywords.

Usage Guidelines

Enter the **debug ip udp** command on the device that should be receiving packets from the host. Check the debugging output to see whether packets are being received from the host.

Caution The **debug ip udp** command can use considerable CPU cycles on the device. Do not enable it if your network is heavily congested.

Examples

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

```
Router# debug ip udp
UDP packet debugging is on
Router#

00:18:48: UDP: rcvd src=0.0.0.0(68), dst=255.255.255.255(67), length=584
00:18:48: UDP: sent src=10.1.1.10(67), dst=172.17.110.136(67), length=604
00:18:48: UDP: rcvd src=172.17.110.136(67), dst=10.1.1.10(67), length=308
00:18:48: UDP: sent src=0.0.0.0(67), dst=255.255.255.255(68), length=328
00:18:48: UDP: rcvd src=0.0.0.0(68), dst=255.255.255.255(67), length=584
00:18:48: UDP: sent src=10.1.1.10(67), dst=172.17.110.136(67), length=604
00:18:48: UDP: rcvd src=172.17.110.136(67), dst=10.1.1.10(67), length=308
00:18:50: UDP: sent src=0.0.0.0(67), dst=255.255.255.255(68), length=328
```

debug ip urd

To display debug messages for URL Rendezvous Directory (URD) channel subscription report processing, use the **debug ip urd** EXEC command. To disable debugging of URD reports, use the **no** form of this command.

debug ip urd [*hostname* | *ip-address*]

no debug ip urd

Syntax Description

hostname	(Optional) The domain Name System (DNS) name.
ip-address	(Optional) The IP address.

Defaults

If no host name or IP address is specified, all URD reports are debugged.

Command History

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

Examples

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

```
Router# debug ip urd

13:36:25 pdt:URD:Data intercepted from 171.71.225.103
13:36:25 pdt:URD:Enqueued string:
'/cgi-bin/error.pl?group=232.16.16.16&port=32620&source=171.69.214.1&li'
13:36:25 pdt:URD:Matched token:group
13:36:25 pdt:URD:Parsed value:232.16.16.16
13:36:25 pdt:URD:Creating IGMP source state for group 232.16.16.16
```

debug ip wccp events

To display information about significant Web Cache Control Protocol (WCCP) events, use the **debug ip wccp events** privileged EXEC command. The **no** form of this command disables debugging output.

debug ip wccp events

no debug ip wccp events

Syntax Description

This command has no arguments or keywords.

Examples

The following is sample output from the **debug ip wccp events** command when a Cisco Cache Engine is added to the list of available Web caches:

```
Router# debug ip wccp events

WCCP-EVNT: Built I_See_You msg body w/1 usable web caches, change # 0000000A
WCCP-EVNT: Web Cache 192.168.25.3 added
WCCP-EVNT: Built I_See_You msg body w/2 usable web caches, change # 0000000B
WCCP-EVNT: Built I_See_You msg body w/2 usable web caches, change # 0000000C
```

debug ip wccp packets

To display information about every Web Cache Control Protocol (WCCP) packet received or sent by the router, use the **debug ip wccp packets** privileged EXEC command. The **no** form of this command disables debugging output.

debug ip wccp packets

no debug ip wccp packets

Syntax Description

This command has no arguments or keywords.

Examples

The following is sample output from the **debug ip wccp packets** command. The router is sending keepalive packets to the Cisco Cache Engines at 192.168.25.4 and 192.168.25.3. Each keepalive packet has an identification number associated with it. When the Cisco Cache Engine receives a keepalive packet from the router, it sends a reply with the identification number back to the router.

```
Router# debug ip wccp packets

WCCP-PKT: Received valid Here_I_Am packet from 192.168.25.4 w/rcvd_id 00003532
WCCP-PKT: Sending I_See_You packet to 192.168.25.4 w/ rcvd_id 00003534
WCCP-PKT: Received valid Here_I_Am packet from 192.168.25.3 w/rcvd_id 00003533
WCCP-PKT: Sending I_See_You packet to 192.168.25.3 w/ rcvd_id 00003535
WCCP-PKT: Received valid Here_I_Am packet from 192.168.25.4 w/rcvd_id 00003534
WCCP-PKT: Sending I_See_You packet to 192.168.25.4 w/ rcvd_id 00003536
WCCP-PKT: Received valid Here_I_Am packet from 192.168.25.3 w/rcvd_id 00003535
WCCP-PKT: Sending I_See_You packet to 192.168.25.3 w/ rcvd_id 00003537
WCCP-PKT: Received valid Here_I_Am packet from 192.168.25.4 w/rcvd_id 00003536
WCCP-PKT: Sending I_See_You packet to 192.168.25.4 w/ rcvd_id 00003538
WCCP-PKT: Received valid Here_I_Am packet from 192.168.25.3 w/rcvd_id 00003537
WCCP-PKT: Sending I_See_You packet to 192.168.25.3 w/ rcvd_id 00003539
```