

debug ip sctp api

To provide diagnostic information about Stream Control Transmission Protocol (SCTP) application programming interfaces (APIs), use the **debug ip sctp api** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip sctp api

no debug ip sctp api

Syntax Description This command has no arguments or keywords.

Defaults No default behavior or values.

Command Modes Privileged EXEC

Command History	Release	Modification
	12.2(4)T	This command was introduced.

Usage Guidelines In a live system, the debug messages for performance, state, signal, and warnings are the most useful. These show any association or destination address failures and can be used to monitor the stability of any established associations.



Caution

The **debug ip sctp api** command should not be used in a live system that has any significant amount of traffic running because it can generate a lot of traffic, which can cause associations to fail.

Examples The following example shows SCTP calls to the API that are being executed and the parameters associated with these calls:

```
Router# debug ip sctp api

*Mar 1 00:31:14.211: SCTP: sctp_send: Assoc ID: 1
*Mar 1 00:31:14.211: SCTP:          stream num: 10
*Mar 1 00:31:14.211: SCTP:          bptr: 62EE332C, dptr: 4F7B598
*Mar 1 00:31:14.211: SCTP:          datalen: 100
*Mar 1 00:31:14.211: SCTP:          context: 1
*Mar 1 00:31:14.211: SCTP:          lifetime: 0
*Mar 1 00:31:14.211: SCTP:          unorder flag: FALSE
*Mar 1 00:31:14.211: SCTP:          bundle flag: TRUE
*Mar 1 00:31:14.211: SCTP: sctp_send successful return
*Mar 1 00:31:14.211: SCTP: sctp_receive: Assoc ID: 1
*Mar 1 00:31:14.215: SCTP:          max data len: 100
*Mar 1 00:31:14.215: SCTP: sctp_receive successful return
```

```
*Mar 1 00:31:14.215: SCTP: Process Send Request
*Mar 1 00:31:14.951: SCTP: sctp_receive: Assoc ID: 0
*Mar 1 00:31:14.951: SCTP:                               max data len: 100
*Mar 1 00:31:14.951: SCTP: sctp_receive successful return
.
.
.
```

Table 113 describes the significant fields shown in the display.

Table 113 *debug ip sctp api Command Field Descriptions*

Field	Description
Assoc ID	Association identifier.
stream num	SCTP stream number.
bptr, dptr	Address of the buffer that contains the data, and address of the start of the data.
datalen	Length of the data that the application is sending (the datagram).
context	A value that is meaningful to the application. Returned with the datagram if the datagram ever needs to be retrieved.
lifetime	Not used.
unordered flag	Specifies that the datagram should be sent as unordered data.
bundle flag	Indicates whether the application wants the datagram to be delayed slightly, trying to bundle it with other data being sent.
max data len	Maximum length of data that can be received—the size of the receive buffer.

Related Commands

Command	Description
clear ip sctp statistics	Empties the buffer that holds SCTP statistics.
debug ip sctp congestion	Shows a list of all current SCTP associations.
show ip sctp association parameters	Shows the parameters configured for the association defined by the association identifier.
show ip sctp association statistics	Shows the current statistics for the association defined by the association identifier.
show ip sctp errors	Shows error counts logged by SCTP.
show ip sctp instances	Shows all currently defined SCTP instances.
show ip sctp statistics	Shows overall statistics counts for SCTP.
show iua as	Shows information about the current condition of an application server.
show iua asp	Shows information about the current condition of an application server process.

debug ip sctp congestion

To provide diagnostic information about Stream Control Transmission Protocol (SCTP) congestion parameters, use the **debug ip sctp congestion** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip sctp congestion

no debug ip sctp congestion

Syntax Description This command has no arguments or keywords.

Defaults No default behavior or values.

Command Modes Privileged EXEC

Command History	Release	Modification
	12.2(4)T	This command was introduced.

Usage Guidelines In a live system, the debug messages for performance, state, signal, and warnings are the most useful. These show any association or destination address failures and can be used to monitor the stability of any established associations.

Debug commands other than those for performance, state, signal, and warnings can generate a great deal of output and therefore can cause associations to fail. These commands should be used only in test environments or when there are very low amounts of traffic.

Examples The following example shows parameters used to calculate SCTP congestion:

```
Router# debug ip sctp congestion

SCTP: Assoc 0: Slow start 10.6.0.4, cwnd 3000
SCTP: Assoc 0: Data chunks rcvd, local rwnd 7800
SCTP: Assoc 0: Free chunks, local rwnd 9000
SCTP: Assoc 0: Data chunks rcvd, local rwnd 8200
SCTP: Assoc 0: Add Sack, local a_rwnd 8200
SCTP: Assoc 0: Free chunks, local rwnd 9000
SCTP: Assoc 0: Data chunks rcvd, local rwnd 7800
SCTP: Assoc 0: Data chunks rcvd, local rwnd 7000
SCTP: Assoc 0: Add Sack, local a_rwnd 7000
SCTP: Assoc 0: Free chunks, local rwnd 9000
SCTP: Assoc 0: Bundle for 10.5.0.4, rem rwnd 14000, cwnd 19500, outstand 0
SCTP: Assoc 0: Bundled 12 chunks, remote rwnd 12800, outstand 1200
SCTP: Assoc 0: Bundling data, next chunk dataLen (100) > remaining mtu size
SCTP: Assoc 0: Bundle for 10.5.0.4, rem rwnd 12800, cwnd 19500, outstand 1200
SCTP: Assoc 0: Bundled 12 chunks, remote rwnd 11600, outstand 2400
SCTP: Assoc 0: Bundling data, next chunk dataLen (100) > remaining mtu size
SCTP: Assoc 0: Bundle for 10.5.0.4, rem rwnd 11600, cwnd 19500, outstand 2400
```

```

SCTP: Assoc 0: Bundled 12 chunks, remote rwnd 10400, outstand 3600
SCTP: Assoc 0: Bundling data, next chunk dataLen (100) > remaining mtu size
SCTP: Assoc 0: Bundle for 10.5.0.4, rem rwnd 10400, cwnd 19500, outstand 3600
SCTP: Assoc 0: Bundled 4 chunks, remote rwnd 10000, outstand 4000
SCTP: Assoc 0: No additional chunks waiting.
SCTP: Assoc 0: Data chunks rcvd, local rwnd 7800
SCTP: Assoc 0: Data chunks rcvd, local rwnd 7000
SCTP: Assoc 0: Add Sack, local a_rwnd 7000
SCTP: Assoc 0: Chunk A22F3B45 ack'd, dest 10.5.0.4, outstanding 3900
SCTP: Assoc 0: Chunk A22F3B46 ack'd, dest 10.5.0.4, outstanding 3800
SCTP: Assoc 0: Chunk A22F3B47 ack'd, dest 10.5.0.4, outstanding 3700
SCTP: Assoc 0: Chunk A22F3B48 ack'd, dest 10.5.0.4, outstanding 3600
SCTP: Assoc 0: Chunk A22F3B49 ack'd, dest 10.5.0.4, outstanding 3500
SCTP: Assoc 0: Chunk A22F3B4A ack'd, dest 10.5.0.4, outstanding 3400
SCTP: Assoc 0: Chunk A22F3B4B ack'd, dest 10.5.0.4, outstanding 3300
SCTP: Assoc 0: Chunk A22F3B4C ack'd, dest 10.5.0.4, outstanding 3200
SCTP: Assoc 0: Chunk A22F3B4D ack'd, dest 10.5.0.4, outstanding 3100
SCTP: Assoc 0: Chunk A22F3B4E ack'd, dest 10.5.0.4, outstanding 3000
SCTP: Assoc 0: Chunk A22F3B4F ack'd, dest 10.5.0.4, outstanding 2900
SCTP: Assoc 0: Chunk A22F3B50 ack'd, dest 10.5.0.4, outstanding 2800
SCTP: Assoc 0: Chunk A22F3B51 ack'd, dest 10.5.0.4, outstanding 2700
SCTP: Assoc 0: Chunk A22F3B52 ack'd, dest 10.5.0.4, outstanding 2600
SCTP: Assoc 0: Chunk A22F3B53 ack'd, dest 10.5.0.4, outstanding 2500
SCTP: Assoc 0: Chunk A22F3B54 ack'd, dest 10.5.0.4, outstanding 2400
SCTP: Assoc 0: Chunk A22F3B55 ack'd, dest 10.5.0.4, outstanding 2300
SCTP: Assoc 0: Chunk A22F3B56 ack'd, dest 10.5.0.4, outstanding 2200

```

Table 114 describes the significant fields shown in the display.

Table 114 *debug ip sctp congestion Command Field Descriptions*

Field	Description
cwnd	Congestion window values for destination address.
rwnd, a_rwnd	Receiver window values as defined in RFC 2960.
outstanding	Number of bytes outstanding.

Related Commands

Command	Description
clear ip sctp statistics	Empties the buffer that holds SCTP statistics.
debug ip sctp congestion	Shows a list of all current SCTP associations.
show ip sctp association parameters	Shows the parameters configured for the association defined by the association identifier.
show ip sctp association statistics	Shows the current statistics for the association defined by the association identifier.
show ip sctp errors	Shows error counts logged by SCTP.
show ip sctp instances	Shows all currently defined SCTP instances.
show ip sctp statistics	Shows overall statistics counts for SCTP.
show iua as	Shows information about the current condition of an application server.
show iua asp	Shows information about the current condition of an application server process.

debug ip sctp init

To show datagrams and other information related to the initializing of new Stream Control Transmission Protocol (SCTP) associations, use the **debug ip sctp init** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip sctp init

no debug ip sctp init

Syntax Description This command has no arguments or keywords.

Defaults No default behavior or values.

Command Modes Privileged EXEC

Command History	Release	Modification
	12.2(4)T	This command was introduced.

Usage Guidelines All initialization chunks are shown, including the INIT, INIT_ACK, COOKIE_ECHO, and COOKIE_ACK chunks. This debug command can be used to see the chunks associated with any initialization sequence but does not display data chunks sent once the association is established. Therefore, it is safe to use in a live system that has traffic flowing when you have trouble with associations failing and being reestablished.

Examples The following example shows initialization chunks for SCTP associations:

```
Router# debug ip sctp init

*Mar 1 00:53:07.279: Sctp Test: Attempting to open assoc to remote port 8787...assoc ID is 0
*Mar 1 00:53:07.279: Sctp: Process Assoc Request
*Mar 1 00:53:07.279: Sctp: Assoc 0: dest addr list:
*Mar 1 00:53:07.279: Sctp:                addr 10.5.0.4
*Mar 1 00:53:07.279: Sctp:                addr 10.6.0.4
*Mar 1 00:53:07.279:
...
*Mar 1 00:53:13.279: Sctp: Assoc 0: Send Init
*Mar 1 00:53:13.279: Sctp:                INIT_CHUNK, len 42
*Mar 1 00:53:13.279: Sctp:                Initiate Tag: B4A10C4D, Initial TSN: B4A10C4D, rwnd 9000
*Mar 1 00:53:13.279: Sctp:                Streams Inbound: 13, Outbound: 13
*Mar 1 00:53:13.279: Sctp:                IP Addr: 10.1.0.2
*Mar 1 00:53:13.279: Sctp:                IP Addr: 10.2.0.2
*Mar 1 00:53:13.279: Sctp:                Supported addr types: 5
*Mar 1 00:53:13.307: Sctp: Process Init
*Mar 1 00:53:13.307: Sctp:                INIT_CHUNK, len 42
*Mar 1 00:53:13.307: Sctp:                Initiate Tag: 3C2D8327, Initial TSN: 3C2D8327, rwnd 18000
*Mar 1 00:53:13.307: Sctp:                Streams Inbound: 13, Outbound: 13
```

```

*Mar 1 00:53:13.307: Sctp:      IP Addr: 10.5.0.4
*Mar 1 00:53:13.307: Sctp:      IP Addr: 10.6.0.4
*Mar 1 00:53:13.307: Sctp:      Supported addr types: 5
*Mar 1 00:53:13.307: Sctp: Assoc 0: Send InitAck
*Mar 1 00:53:13.307: Sctp:      INIT_ACK_CHUNK, len 124
*Mar 1 00:53:13.307: Sctp:      Initiate Tag: B4A10C4D, Initial TSN: B4A10C4D, rwnd 9000
*Mar 1 00:53:13.307: Sctp:      Streams Inbound: 13, Outbound: 13
*Mar 1 00:53:13.307: Sctp:      Responder cookie len 88
*Mar 1 00:53:13.307: Sctp:      IP Addr: 10.1.0.2
*Mar 1 00:53:13.307: Sctp:      IP Addr: 10.2.0.2
*Mar 1 00:53:13.311: Sctp: Assoc 0: Process Cookie
*Mar 1 00:53:13.311: Sctp:      COOKIE_ECHO_CHUNK, len 88
*Mar 1 00:53:13.311: Sctp: Assoc 0: dest addr list:
*Mar 1 00:53:13.311: Sctp:          addr 10.5.0.4
*Mar 1 00:53:13.311: Sctp:          addr 10.6.0.4
*Mar 1 00:53:13.311:
*Mar 1 00:53:13.311: Sctp: Instance 0 dest addr list:
*Mar 1 00:53:13.311: Sctp:          addr 10.5.0.4
*Mar 1 00:53:13.311: Sctp:          addr 10.6.0.4
*Mar 1 00:53:13.311:
*Mar 1 00:53:13.311: Sctp: Assoc 0: Send CookieAck
*Mar 1 00:53:13.311: Sctp:      COOKIE_ACK_CHUNK

```

Table 115 describes the significant fields shown in the display.

Table 115 debug ip sctp init Command Field Descriptions

Field	Description
Initiate Tag	Initiation chunk identifier.
Initial TSN	Initial transmission sequence number.
rwnd	Receiver window values.

Related Commands

Command	Description
clear ip sctp statistics	Empties the buffer that holds Sctp statistics.
debug ip sctp congestion	Shows a list of all current Sctp associations.
show ip sctp association parameters	Shows the parameters configured for the association defined by the association identifier.
show ip sctp association statistics	Shows the current statistics for the association defined by the association identifier.
show ip sctp errors	Shows error counts logged by Sctp.
show ip sctp instances	Shows all currently defined Sctp instances.
show ip sctp statistics	Shows overall statistics counts for Sctp.
show iua as	Shows information about the current condition of an application server.
show iua asp	Shows information about the current condition of an application server process.

debug ip sctp multihome

To show the source and destination of datagrams in order to monitor the use of the multihome addresses for Stream Control Transmission Protocol (SCTP), use the **debug ip sctp multihome** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip sctp multihome

no debug ip sctp multihome

Syntax Description This command has no arguments or keywords.

Defaults No default behavior or values.

Command Modes Privileged EXEC

Command History	Release	Modification
	12.2(4)T	This command was introduced.

Usage Guidelines More than one IP address parameter can be included in an initialization (INIT) chunk when the INIT sender is multihomed. Datagrams should be sent to the primary destination addresses unless the network is experiencing problems, in which case the datagrams should be sent to secondary addresses.



Caution

The **debug ip sctp multihome** command generates one debug line for each datagram sent or received. It should be used with extreme caution in a live network.

Examples The following example shows source and destination for multihomed addresses:

```
Router# debug ip sctp multihome

SCTP: Rcvd s=10.5.0.4 8787, d=10.1.0.2 8787, len 1404
SCTP: Rcvd s=10.5.0.4 8787, d=10.1.0.2 8787, len 476
SCTP: Sent: Assoc 0: s=10.1.0.2 8787, d=10.5.0.4 8787, len 28
SCTP: Assoc 0: Send Data to dest 10.5.0.4
SCTP: Sent: Assoc 0: s=10.1.0.2 8787, d=10.5.0.4 8787, len 1404
SCTP: Sent: Assoc 0: s=10.1.0.2 8787, d=10.5.0.4 8787, len 1404
SCTP: Sent: Assoc 0: s=10.1.0.2 8787, d=10.5.0.4 8787, len 1404
SCTP: Sent: Assoc 0: s=10.1.0.2 8787, d=10.5.0.4 8787, len 476
SCTP: Rcvd s=10.5.0.4 8787, d=10.1.0.2 8787, len 28
SCTP: Rcvd s=10.5.0.4 8787, d=10.1.0.2 8787, len 28
SCTP: Rcvd s=10.5.0.4 8787, d=10.1.0.2 8787, len 1404
SCTP: Rcvd s=10.5.0.4 8787, d=10.1.0.2 8787, len 1404
SCTP: Sent: Assoc 0: s=10.1.0.2 8787, d=10.5.0.4 8787, len 28
SCTP: Rcvd s=10.5.0.4 8787, d=10.1.0.2 8787, len 1404
SCTP: Rcvd s=10.5.0.4 8787, d=10.1.0.2 8787, len 476
SCTP: Sent: Assoc 0: s=10.1.0.2 8787, d=10.5.0.4 8787, len 28
```

```

SCTP: Assoc 0: Send Data to dest 10.5.0.4
SCTP: Sent: Assoc 0: s=10.1.0.2 8787, d=10.5.0.4 8787, len 1404
SCTP: Sent: Assoc 0: s=10.1.0.2 8787, d=10.5.0.4 8787, len 1404
SCTP: Sent: Assoc 0: s=10.1.0.2 8787, d=10.5.0.4 8787, len 1404
SCTP: Sent: Assoc 0: s=10.1.0.2 8787, d=10.5.0.4 8787, len 476
SCTP: Rcvd s=10.6.0.4 8787, d=10.2.0.2 8787, len 44
SCTP: Sent: Assoc 0: s=10.2.0.2 8787, d=10.6.0.4 8787, len 44
SCTP: Rcvd s=10.5.0.4 8787, d=10.1.0.2 8787, len 28
SCTP: Rcvd s=10.5.0.4 8787, d=10.1.0.2 8787, len 28
SCTP: Rcvd s=10.5.0.4 8787, d=10.1.0.2 8787, len 1404
SCTP: Rcvd s=10.5.0.4 8787, d=10.1.0.2 8787, len 1404
SCTP: Sent: Assoc 0: s=10.1.0.2 8787, d=10.5.0.4 8787, len 28
SCTP: Rcvd s=10.5.0.4 8787, d=10.1.0.2 8787, len 1404
SCTP: Rcvd s=10.5.0.4 8787, d=10.1.0.2 8787, len 476

```

Table 116 describes the significant fields shown in the display.

Table 116 *debug ip sctp multihome Command Field Descriptions*

Field	Description
s	Source address and port.
d	Destination address and port.

Related Commands

Command	Description
clear ip sctp statistics	Empties the buffer that holds SCTP statistics.
debug ip sctp congestion	Shows a list of all current SCTP associations.
show ip sctp association parameters	Shows the parameters configured for the association defined by the association identifier.
show ip sctp association statistics	Shows the current statistics for the association defined by the association identifier.
show ip sctp errors	Shows error counts logged by SCTP.
show ip sctp instances	Shows all currently defined SCTP instances.
show ip sctp statistics	Shows overall statistics counts for SCTP.
show iua as	Shows information about the current condition of an application server.
show iua asp	Shows information about the current condition of an application server process.

debug ip sctp performance

To display the average number of Stream Control Transmission Protocol (SCTP) chunks and datagrams being sent and received per second, use the **debug ip sctp performance** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip sctp performance

no debug ip sctp performance

Syntax Description This command has no arguments or keywords.

Defaults No default behavior or values.

Command Modes Privileged EXEC

Command History	Release	Modification
	12.2(4)T	This command was introduced.

Usage Guidelines In a live system, the debug messages for performance, state, signal, and warnings are the most useful. These show any association or destination address failures and can be used to monitor the stability of any established associations.

Once enabled, the **debug ip sctp performance** command displays the average number of chunks and datagrams being sent and received per second once every 10 seconds. Note that the averages are cumulative since the last time the statistics were cleared using the **clear ip sctp statistics** command and may not accurately reflect the number of datagrams and chunks currently being sent and received at that particular moment.

Examples The following example shows a low rate of traffic:

```
Router# debug ip sctp performance

SCTP Sent: SCTP Dgrams 5, Chunks 28, Data Chunks 29, ULP Dgrams 29
SCTP Rcvd: SCTP Dgrams 7, Chunks 28, Data Chunks 29, ULP Dgrams 29
Chunks Discarded: 0, Retransmitted 0

SCTP Sent: SCTP Dgrams 6, Chunks 29, Data Chunks 30, ULP Dgrams 30
SCTP Rcvd: SCTP Dgrams 7, Chunks 29, Data Chunks 30, ULP Dgrams 30
Chunks Discarded: 0, Retransmitted 0
```

Table 117 describes the significant fields shown in the display.

Table 117 *debug ip sctp performance Command Field Descriptions*

Field	Description
SCTP Dgrams	Datagram sent to or received from the network.
Chunks	Includes data chunks and control chunks sent or received.
Data Chunks	Data chunks sent or received.
ULP Dgrams	Upper-layer protocol (ULP) datagrams, which are datagrams sent to or received from the ULP or application.

Related Commands

Command	Description
clear ip sctp statistics	Empties the buffer that holds SCTP statistics.
debug ip sctp congestion	Shows a list of all current SCTP associations.
show ip sctp association parameters	Shows the parameters configured for the association defined by the association identifier.
show ip sctp association statistics	Shows the current statistics for the association defined by the association identifier.
show ip sctp errors	Shows error counts logged by SCTP.
show ip sctp instances	Shows all currently defined SCTP instances.
show ip sctp statistics	Shows overall statistics counts for SCTP.
show iua as	Shows information about the current condition of an application server.
show iua asp	Shows information about the current condition of an application server process.

debug ip sctp rcvchunks

To provide diagnostic information about chunks received with Stream Control Transmission Protocol (SCTP), use the **debug ip sctp rcvchunks** command in privileged EXEC mode. To disable diagnostic reporting, use the **no** form of this command.

debug ip sctp rcvchunks

no debug ip sctp rcvchunks

Syntax Description This command has no arguments or keywords.

Defaults No default behavior or values.

Command Modes Privileged EXEC

Command History	Release	Modification
	12.2(4)T	This command was introduced.

Usage Guidelines The **debug ip sctp rcvchunks** command shows the following information about received chunks:

- Whether the chunk is for a new datagram or is part of a datagram that is being reassembled.
- Whether the datagram is complete after receiving this chunk.
- If the datagram is complete, whether the datagram is in sequence within the specified stream and can be delivered to the upper-layer protocol (ULP).
- The selective acknowledgements (SACKs) that are returned to the remote SCTP peer.
- The cumulative transmission sequence number (Cum TSN) that was acknowledged and the number of fragments included.
- Whether the datagram is received by the ULP.



Caution

The **debug ip sctp rcvchunks** command generates multiple debug lines for each chunk received. It should be used with extreme caution in a live network.

Examples

In the following example, a segmented datagram is received in two chunks for stream 0 and sequence number 0. The length of the first chunk is 1452 bytes, and the second is 1 byte. The first chunk indicates that it is for a new datagram, but the second chunk indicates that it is part of an existing datagram that is already being reassembled. When the first chunk is processed, it is noted to be in sequence, but is not complete and so cannot be delivered yet. When the second chunk is received, the datagram is both in sequence and complete. The application receives the datagram, and a SACK is shown to acknowledge that both chunks were received with no missing chunks indicated (that is, with no fragments).

```
Router# debug ip sctp rcvchunks
```

```
SCTP: Assoc 0: New chunk (0/0/1452/2C33D822) for new dgram (0)
SCTP: Assoc 0: dgram (0) is in seq
SCTP: Assoc 0: Add Sack Chunk, CumTSN=2C33D822, numFrag=0
SCTP: Assoc 0: New chunk (0/0/1/2C33D823) for existing dgram (0)
SCTP: Assoc 0: dgram (0) is complete
SCTP: Assoc 0: ApplRecv chunk 0/0/1452/2C33D822
SCTP: Assoc 0: ApplRecv chunk 0/0/1/2C33D823
SCTP: Assoc 0: Add Sack Chunk, CumTSN=2C33D823, numFrag=0
```

Table 118 describes the significant fields shown in the display.

Table 118 *debug ip sctp rcvchunks Command Field Descriptions*

Field	Description
0 / 0 / 1452 / 2C33D822	Stream number / datagram sequence number / chunk length, in bytes / chunk transmission sequence number.
Sack Chunk	Selective acknowledgement chunk.
ApplRecv	Application has received the chunk.
CumTSN	Cumulative transmission sequence number that is being acknowledged.
numFrag	Number of fragments, or missing chunks.

Related Commands

Command	Description
clear ip sctp statistics	Empties the buffer that holds SCTP statistics.
debug ip sctp congestion	Shows a list of all current SCTP associations.
show ip sctp association parameters	Shows the parameters configured for the association defined by the association identifier.
show ip sctp association statistics	Shows the current statistics for the association defined by the association identifier.
show ip sctp errors	Shows error counts logged by SCTP.
show ip sctp instances	Shows all currently defined SCTP instances.
show ip sctp statistics	Shows overall statistics counts for SCTP.
show iua as	Shows information about the current condition of an application server.
show iua asp	Shows information about the current condition of an application server process.

debug ip sctp rto

To show adjustments that are made to the retransmission timeout (RTO) value when using Stream Control Transmission Protocol (SCTP), use the **debug ip sctp rto** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip sctp rto

no debug ip sctp rto

Syntax Description This command has no arguments or keywords.

Defaults No default behavior or values.

Command Modes Privileged EXEC

Command History	Release	Modification
	12.2(4)T	This command was introduced.

Usage Guidelines The **debug ip sctp rto** command shows adjustments that are made to the retransmission (retrans) timeout value because of either retransmission of data chunks or unacknowledged heartbeats.



Caution

The **debug ip sctp rto** command can generate a great deal of output. It should be used with extreme caution in a live network.

Examples In the following example, there is only one destination address available. Each time the chunk needs to be retransmitted, the RTO value is doubled.

```
Router# debug ip sctp rto

SCTP: Assoc 0: destaddr 10.5.0.4, retrans timeout on chunk 942BAC55
SCTP: Assoc 0: destaddr 10.5.0.4, rto backoff 2000 ms
SCTP: Assoc 0: destaddr 10.5.0.4, retrans timeout on chunk 942BAC55
SCTP: Assoc 0: destaddr 10.5.0.4, rto backoff 4000 ms
SCTP: Assoc 0: destaddr 10.5.0.4, retrans timeout on chunk 942BAC55
SCTP: Assoc 0: destaddr 10.5.0.4, rto backoff 8000 ms
SCTP: Assoc 0: destaddr 10.5.0.4, retrans timeout on chunk 942BAC55
SCTP: Assoc 0: destaddr 10.5.0.4, rto backoff 16000 ms
SCTP: Assoc 0: destaddr 10.5.0.4, retrans timeout on chunk 942BAC55
SCTP: Assoc 0: destaddr 10.5.0.4, rto backoff 32000 ms
```

Related Commands

Command	Description
clear ip sctp statistics	Empties the buffer that holds Sctp statistics.
debug ip sctp congestion	Shows a list of all current Sctp associations.
show ip sctp association parameters	Shows the parameters configured for the association defined by the association identifier.
show ip sctp association statistics	Shows the current statistics for the association defined by the association identifier.
show ip sctp errors	Shows error counts logged by Sctp.
show ip sctp instances	Shows all currently defined Sctp instances.
show ip sctp statistics	Shows overall statistics counts for Sctp.
show iua as	Shows information about the current condition of an application server.
show iua asp	Shows information about the current condition of an application server process.

debug ip sctp segments

To show short diagnostics for every datagram that is sent or received with Stream Control Transmission Protocol (SCTP), use the **debug ip sctp segments** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip sctp segments

no debug ip sctp segments

Syntax Description This command has no arguments or keywords.

Defaults No default behavior or values.

Command Modes Privileged EXEC

Command History	Release	Modification
	12.2(4)T	This command was introduced.

Usage Guidelines The **debug ip sctp segments** command provides the short form of the output about datagrams. For the verbose form, use the **debug ip sctp segmentv** command.



Caution

The **debug ip sctp segments** command generates several lines of output for each datagram sent or received. It should be used with extreme caution in a live network.

Examples The following output shows an example in which an association is established, a few heartbeats are sent, the remote endpoint fails, and the association is restarted.

```
Router# debug ip sctp segments

SCTP: Sent:  Assoc 0: s=10.1.0.2  8787, d=10.5.0.4  8787, len 56
SCTP:      INIT_CHUNK, Tag: 3C72A02A, TSN: 3C72A02A
SCTP: Recv:  Assoc 0: s=10.5.0.4  8787, d=10.1.0.2  8787, len 56
SCTP:      INIT_CHUNK, Tag: 13E5AD6C, TSN: 13E5AD6C
SCTP: Sent:  Assoc NULL: s=10.1.0.2  8787, d=10.5.0.4  8787, len 136
SCTP:      INIT_ACK_CHUNK, Tag: 3C72A02A, TSN: 3C72A02A
SCTP: Recv:  Assoc 0: s=10.5.0.4  8787, d=10.1.0.2  8787, len 100
SCTP:      COOKIE_ECHO_CHUNK, len 88
SCTP: Sent:  Assoc NULL: s=10.1.0.2  8787, d=10.5.0.4  8787, len 16
SCTP:      COOKIE_ACK_CHUNK
SCTP: Sent:  Assoc 0: s=10.1.0.2  8787, d=10.5.0.4  8787, len 52
SCTP:      HEARTBEAT_CHUNK
SCTP: Sent:  Assoc 0: s=10.1.0.2  8787, d=10.5.0.4  8787, len 52
SCTP:      HEARTBEAT_CHUNK
SCTP: Sent:  Assoc 0: s=10.1.0.2  8787, d=10.5.0.4  8787, len 52
SCTP:      HEARTBEAT_CHUNK
```

```

SCTP: Recv: Assoc 0: s=10.5.0.4 8787, d=10.1.0.2 8787, len 56
SCTP: INIT_CHUNK, Tag: 4F2D8235, TSN: 4F2D8235
SCTP: Sent: Assoc NULL: s=10.1.0.2 8787, d=10.5.0.4 8787, len 136
SCTP: INIT_ACK_CHUNK, Tag: 7DD7E424, TSN: 7DD7E424
SCTP: Recv: Assoc 0: s=10.5.0.4 8787, d=10.1.0.2 8787, len 100
SCTP: COOKIE_ECHO_CHUNK, len 88
SCTP: Sent: Assoc NULL: s=10.1.0.2 8787, d=10.5.0.4 8787, len 16
SCTP: COOKIE_ACK_CHUNK
SCTP: Recv: Assoc 0: s=10.5.0.4 8787, d=10.1.0.2 8787, len 144
SCTP: SACK_CHUNK, TSN ack: 7DD7E423, rwnd 18000, num frags 0
SCTP: DATA_CHUNK, 4/0/100/4F2D8235
SCTP: Sent: Assoc 0: s=10.1.0.2 8787, d=10.5.0.4 8787, len 28
SCTP: SACK_CHUNK, TSN ack: 4F2D8235, rwnd 8900, num frags 0
SCTP: Sent: Assoc 0: s=10.1.0.2 8787, d=10.5.0.4 8787, len 128
SCTP: DATA_CHUNK, 4/0/100/7DD7E424
SCTP: Recv: Assoc 0: s=10.5.0.4 8787, d=10.1.0.2 8787, len 28
SCTP: SACK_CHUNK, TSN ack: 7DD7E424, rwnd 17900, num frags 0
SCTP: Recv: Assoc 0: s=10.6.0.4 8787, d=10.2.0.2 8787, len 44
SCTP: HEARTBEAT_CHUNK
SCTP: Sent: Assoc 0: s=10.2.0.2 8787, d=10.6.0.4 8787, len 44
SCTP: HEARTBEAT_ACK_CHUNK
SCTP: Recv: Assoc 0: s=10.5.0.4 8787, d=10.1.0.2 8787, len 128
SCTP: DATA_CHUNK, 7/0/100/4F2D8236
SCTP: Sent: Assoc 0: s=10.1.0.2 8787, d=10.5.0.4 8787, len 144
SCTP: SACK_CHUNK, TSN ack: 4F2D8236, rwnd 9000, num frags 0
SCTP: DATA_CHUNK, 7/0/100/7DD7E425
SCTP: Recv: Assoc 0: s=10.5.0.4 8787, d=10.1.0.2 8787, len 28
SCTP: SACK_CHUNK, TSN ack: 7DD7E424, rwnd 18000, num frags 0
SCTP: Recv: Assoc 0: s=10.5.0.4 8787, d=10.1.0.2 8787, len 28
SCTP: SACK_CHUNK, TSN ack: 7DD7E425, rwnd 17900, num frags 0
SCTP: Recv: Assoc 0: s=10.5.0.4 8787, d=10.1.0.2 8787, len 128
SCTP: DATA_CHUNK, 4/1/100/4F2D8237

```

Table 119 describes the significant fields shown in the display.

Table 119 debug ip sctp segments Command Field Descriptions

Field	Description
s	Source address and port.
d	Destination address and port.
len	Length of chunk, in bytes.
Tag	The identifier for an initialization chunk.
TSN	Transmission sequence number.
rwnd	Receiver window value.
num frags	Number of fragments received.
7 / 0 / 100 / 4F2D8236	(Data chunks) Stream number / datagram sequence number / chunk length, in bytes / chunk transmission sequence number.

Related Commands

Command	Description
clear ip sctp statistics	Empties the buffer that holds SCTP statistics.
debug ip sctp congestion	Shows a list of all current SCTP associations.
debug ip sctp segmentv	Shows every datagram that is sent or received and the chunks that are contained in each. This is the verbose form of the output, and it shows detailed information for each chunk type.
show ip sctp association parameters	Shows the parameters configured for the association defined by the association identifier.
show ip sctp association statistics	Shows the current statistics for the association defined by the association identifier.
show ip sctp errors	Shows error counts logged by SCTP.
show ip sctp instances	Shows all currently defined SCTP instances.
show ip sctp statistics	Shows overall statistics counts for SCTP.
show iua as	Shows information about the current condition of an application server.
show iua asp	Shows information about the current condition of an application server process.

debug ip sctp segmentv

To show verbose diagnostics for every datagram that is sent or received with Stream Control Transmission Protocol (SCTP), use the **debug ip sctp segmentv** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip sctp segmentv

no debug ip sctp segmentv

Syntax Description This command has no arguments or keywords.

Defaults No default behavior or values.

Command Modes Privileged EXEC

Command History	Release	Modification
	12.2(4)T	This command was introduced.

Usage Guidelines The **debug ip sctp segmentv** command provides the verbose form of the output for datagrams. For the simple form, use the **debug ip sctp segments** command.



Caution

The **debug ip sctp segmentv** command generates multiple lines of output for each datagram sent and received. It should be used with extreme caution in a live network.

Examples The following output shows an example in which an association is established, a few heartbeats are sent, the remote endpoint fails, and the association is restarted:

```
Router# debug ip sctp segmentv

SCTP: Sent:  Assoc 0: s=10.1.0.2 8787, d=10.5.0.4 8787, len 56, ver tag 0
SCTP:      INIT_CHUNK, len 42
SCTP:      Initiate Tag: B131ED6A, Initial TSN: B131ED6A, rwnd 9000
SCTP:      Streams Inbound: 13, Outbound: 13
SCTP:      IP Addr: 10.1.0.2
SCTP:      IP Addr: 10.2.0.2
SCTP:      Supported addr types: 5
SCTP: Recv:  Assoc 0: s=10.5.0.4 8787, d=10.1.0.2 8787, len 56, ver tag 0
SCTP:      INIT_CHUNK, len 42
SCTP:      Initiate Tag: 5516B2F3, Initial TSN: 5516B2F3, rwnd 18000
SCTP:      Streams Inbound: 13, Outbound: 13
SCTP:      IP Addr: 10.5.0.4
SCTP:      IP Addr: 10.6.0.4
SCTP:      Supported addr types: 5
SCTP: Sent:  Assoc NULL: s=10.1.0.2 8787, d=10.5.0.4 8787, len 136, ver tag 5516B2F3
SCTP:      INIT_ACK_CHUNK, len 124
```

```

SCTP:      Initiate Tag: B131ED6A, Initial TSN: B131ED6A, rwnd 9000
SCTP:      Streams Inbound: 13, Outbound: 13
SCTP:      Responder cookie len 88
SCTP:      IP Addr: 10.1.0.2
SCTP:      IP Addr: 10.2.0.2
SCTP: Recv: Assoc 0: s=10.5.0.4 8787, d=10.1.0.2 8787, len 100, ver tag B131ED6A
SCTP:      COOKIE_ECHO_CHUNK, len 88
SCTP: Sent: Assoc NULL: s=10.1.0.2 8787, d=10.5.0.4 8787, len 16, ver tag 5516B2F3
SCTP:      COOKIE_ACK_CHUNK
SCTP: Recv: Assoc 0: s=10.5.0.4 8787, d=10.1.0.2 8787, len 144, ver tag B131ED6A
SCTP:      SACK_CHUNK, len 16
SCTP:      TSN ack: (0xB131ED69)
SCTP:      Rcv win credit: 18000
SCTP:      Num frags: 0
SCTP:      DATA_CHUNK, flags 3, chunkLen 116
SCTP:      DATA_CHUNK, 0/0/100/5516B2F3
SCTP: Sent: Assoc 0: s=10.1.0.2 8787, d=10.5.0.4 8787, len 28, ver tag 5516B2F3
SCTP:      SACK_CHUNK, len 16
SCTP:      TSN ack: (0x5516B2F3)
SCTP:      Rcv win credit: 8900
SCTP:      Num frags: 0
SCTP: Sent: Assoc 0: s=10.1.0.2 8787, d=10.5.0.4 8787, len 128, ver tag 5516B2F3
SCTP:      DATA_CHUNK, flags 3, chunkLen 116
SCTP:      DATA_CHUNK, 0/0/100/B131ED6A
SCTP: Recv: Assoc 0: s=10.6.0.4 8787, d=10.2.0.2 8787, len 44, ver tag B131ED6A
SCTP:      HEARTBEAT_CHUNK
SCTP: Sent: Assoc 0: s=10.2.0.2 8787, d=10.6.0.4 8787, len 44, ver tag 5516B2F3
SCTP:      HEARTBEAT_ACK_CHUNK
SCTP: Recv: Assoc 0: s=10.5.0.4 8787, d=10.1.0.2 8787, len 28, ver tag B131ED6A
SCTP:      SACK_CHUNK, len 16

```

Table 120 describes the significant fields shown in the display.

Table 120 *debug ip sctp segmentv Command Field Descriptions*

Field	Description
s	Source address and port.
d	Destination address and port.
len	Length of chunk, in bytes.
ver tag	Verification identifier.
Tag	The identifier for an initialization chunk.
TSN	Transmission sequence number.
rwnd	Receive window value.
Rcv win credit	Receive window value. Same as rwnd.
Num frags	Number of fragments received.
0/0/100/5516B2F3	(Data chunks) Stream number / datagram sequence number / chunk length, in bytes / chunk transmission sequence number.

Related Commands

Command	Description
clear ip sctp statistics	Empties the buffer that holds Sctp statistics.
debug ip sctp congestion	Shows a list of all current Sctp associations.

Command	Description
debug ip sctp segments	Shows short diagnostics for every datagram that is sent or received with Sctp.
show ip sctp association parameters	Shows the parameters configured for the association defined by the association identifier.
show ip sctp association statistics	Shows the current statistics for the association defined by the association identifier.
show ip sctp errors	Shows error counts logged by Sctp.
show ip sctp instances	Shows all currently defined Sctp instances.
show ip sctp statistics	Shows overall statistics counts for Sctp.
show iua as	Shows information about the current condition of an application server.
show iua asp	Shows information about the current condition of an application server process.

debug ip sctp signal

To show signals that are sent from Stream Control Transmission Protocol (SCTP) to the application or upper-layer protocol (ULP), use the **debug ip sctp signal** command in privileged EXEC mode. To disable diagnostic reporting, use the **no** form of this command.

debug ip sctp signal

no debug ip sctp signal

Syntax Description This command has no arguments or keywords.

Defaults No default behavior or values.

Command Modes Privileged EXEC

Command History	Release	Modification
	12.2(4)T	This command was introduced.

Usage Guidelines The **debug ip sctp signal** command can be used to see if the current associations are stable or not. Because it generates output only on state transitions, it is safe to use in a live environment. It still should be used with caution, however, depending on the number of associations being handled by the system and the stability of the network.

The **debug ip sctp state** command is often used at the same time as the **debug ip sctp signal** command. Using the two commands together gives good insight into the stability of associations.

Examples In the following example, a new association is requested and established. The peer then restarts the association and notes that the association failed and is being reestablished. The local peer then indicates that the association has failed because it has tried to retransmit the specified chunk more than the maximum number of times without success. As a result, the association fails (because of communication loss) and is terminated. The ULP requests that the association be attempted again, and this attempt succeeds. A shutdown is then received from the remote peer, and the local peer enters the shutdown acknowledge sent state, which is followed by the association being terminated. Again, another association attempt is made and succeeds.

```
Router# debug ip sctp signal
Router# debug ip sctp state
```

```
<new assoc attempt>
00:20:08: SCTP: Assoc 0: state CLOSED -> COOKIE_WAIT
00:20:15: SCTP: Assoc 0: state COOKIE_WAIT -> ESTABLISHED
00:20:15: SCTP: Assoc 0: Sent ASSOC_UP signal for CONFIGD_ASSOC
00:21:03: SCTP: Assoc 0: Restart rcvd from peer
00:21:03: SCTP: Assoc 0: Sent ASSOC_RESTART signal
00:21:04: SCTP: Assoc 0: chunk 62EA7F40 retransmitted more than max times, failing assoc
```

```

00:21:04: Sctp: Assoc 0: Sent ASSOC_FAILED signal, reason: Sctp_COMM_LOST
00:21:04: Sctp: Assoc 0: Sent ASSOC_TERMINATE signal
00:21:04: Sctp: Assoc 0: state ESTABLISHED -> CLOSED
<new assoc attempt>
00:21:04: Sctp: Assoc 0: state CLOSED -> COOKIE_WAIT
00:21:04: Sctp: Assoc 0: state COOKIE_WAIT -> COOKIE_ECHOED
00:21:04: Sctp: Assoc 0: state COOKIE_ECHOED -> ESTABLISHED
00:21:04: Sctp: Assoc 0: Sent ASSOC_UP signal for CONFIGD_ASSOC
00:21:04: Sctp: Assoc 0: Sent TERMINATE_PENDING signal
00:21:04: Sctp: Assoc 0: state ESTABLISHED -> SHUTDOWN_ACKSENT
00:21:04: Sctp: Assoc 0: Sent ASSOC_TERMINATE signal
00:21:04: Sctp: Assoc 0: state SHUTDOWN_ACKSENT -> CLOSED
<new assoc attempt>
00:21:04: Sctp: Assoc 0: state CLOSED -> COOKIE_WAIT
00:21:04: Sctp: Assoc 0: state COOKIE_WAIT -> COOKIE_ECHOED
00:21:04: Sctp: Assoc 0: state COOKIE_ECHOED -> ESTABLISHED
00:21:04: Sctp: Assoc 0: Sent ASSOC_UP signal for CONFIGD_ASSOC

```

Related Commands

Command	Description
clear ip sctp statistics	Empties the buffer that holds Sctp statistics.
debug ip sctp congestion	Shows a list of all current Sctp associations.
debug ip sctp state	Shows Sctp state transitions.
show ip sctp association parameters	Shows the parameters configured for the association defined by the association identifier.
show ip sctp association statistics	Shows the current statistics for the association defined by the association identifier.
show ip sctp errors	Shows error counts logged by Sctp.
show ip sctp instances	Shows all currently defined Sctp instances.
show ip sctp statistics	Shows overall statistics counts for Sctp.
show iua as	Shows information about the current condition of an application server.
show iua asp	Shows information about the current condition of an application server process.

debug ip sctp sndchunks

To show information about chunks that are being sent to remote Stream Control Transmission Protocol (SCTP) peers, use the **debug ip sctp sndchunks** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip sctp sndchunks

no debug ip sctp sndchunks

Syntax Description This command has no arguments or keywords.

Defaults No default behavior or values.

Command Modes Privileged EXEC

Command History	Release	Modification
	12.2(4)T	This command was introduced.

Usage Guidelines The **debug ip sctp sndchunks** command provides the following information:

- Application send requests from the local SCTP peer
- Chunks being bundled and sent to the remote peer
- Processing of the Selective acknowledgements (SACKs) from the remote peer, indicating which chunks were successfully received
- Chunks that are marked for retransmission



Caution

The **debug ip sctp sndchunks** command generates large amounts of data if there is any significant amount of traffic flowing. It should be used with extreme caution in live networks.

Examples The following example shows output for the **debug ip sctp sndchunks** command for a case in which data chunks are being sent, with some of them marked for retransmission:

```
Router# debug ip sctp sndchunks

SCTP: Assoc 0: ApplSend, chunk: 0/10412/100/A23134F8 to 10.5.0.4
SCTP: Assoc 0: ApplSend, chunk: 5/10443/100/A23134F9 to 10.5.0.4
SCTP: Assoc 0: ApplSend, chunk: 5/10448/100/A231355C to 10.5.0.4
SCTP: Assoc 0: Set oldest chunk for dest 10.5.0.4 to TSN A23134F8
SCTP: Assoc 0: Bundling data, added 0/10412/100/A23134F8, outstanding 100
SCTP: Assoc 0: Bundling data, added 5/10443/100/A23134F9, outstanding 200
SCTP: Assoc 0: Bundling data, added 4/10545/100/A23134FA, outstanding 300
SCTP: Assoc 0: Bundling data, added 10/10371/100/A23134FB, outstanding 400
SCTP: Assoc 0: Bundling data, added 11/10382/100/A23134FC, outstanding 500
```

```

SCTP: Assoc 0: Process Sack Chunk, CumTSN=A231350F, numFrag=0
SCTP: Assoc 0: Reset oldest chunk on addr 10.5.0.4 to A2313510
SCTP: Assoc 0: Process Sack Chunk, CumTSN=A2313527, numFrag=0
SCTP: Assoc 0: Reset oldest chunk on addr 10.5.0.4 to A2313528
SCTP: Assoc 0: Process Sack Chunk, CumTSN=A231353F, numFrag=0
SCTP: Assoc 0: Reset oldest chunk on addr 10.5.0.4 to A2313540
SCTP: Assoc 0: Process Sack Chunk, CumTSN=A2313557, numFrag=0
SCTP: Assoc 0: Reset oldest chunk on addr 10.5.0.4 to A2313558
SCTP: Assoc 0: ApplSend, chunk: 10/10385/100/A23135BE to 10.5.0.4
SCTP: Assoc 0: ApplSend, chunk: 8/10230/100/A23135BF to 10.5.0.4
SCTP: Assoc 0: ApplSend, chunk: 5/10459/100/A23135C0 to 10.5.0.4
SCTP: Assoc 0: ApplSend, chunk: 4/10558/100/A23135C1 to 10.5.0.4
SCTP: Assoc 0: Set oldest chunk for dest 10.5.0.4 to TSN A231355D
SCTP: Assoc 0: Bundling data, added 5/10449/100/A231355D, outstanding 100
SCTP: Assoc 0: Bundling data, added 3/10490/100/A231355E, outstanding 200
SCTP: Assoc 0: Process Sack Chunk, CumTSN=A23135A4, numFrag=0
SCTP: Assoc 0: Reset oldest chunk on addr 10.5.0.4 to A23135A5
SCTP: Assoc 0: Process Sack Chunk, CumTSN=A23135BC, numFrag=0
SCTP: Assoc 0: Reset oldest chunk on addr 10.5.0.4 to A23135BD
SCTP: Assoc 0: Process Sack Chunk, CumTSN=A23135C1, numFrag=0
SCTP: Assoc 0: ApplSend, chunk: 5/10460/100/A23135C2 to 10.5.0.4
SCTP: Assoc 0: ApplSend, chunk: 5/10461/100/A23135C3 to 10.5.0.4
SCTP: Assoc 0: ApplSend, chunk: 11/10403/100/A2313626 to 10.5.0.4
SCTP: Assoc 0: Set oldest chunk for dest 10.5.0.4 to TSN A23135C2
SCTP: Assoc 0: Bundling data, added 5/10460/100/A23135C2, outstanding 100
SCTP: Assoc 0: Bundling data, added 5/10461/100/A23135C3, outstanding 200
SCTP: Assoc 0: Bundling data, added 5/10462/100/A23135C4, outstanding 300
SCTP: Assoc 0: Bundling data, added 4/10559/100/A23135C5, outstanding 400
SCTP: Assoc 0: Bundling data, added 4/10560/100/A23135C6, outstanding 500
SCTP: Assoc 0: Bundled 12 chunk(s) in next dgram to 10.5.0.4
SCTP: Assoc 0: Bundling data, added 1/10418/100/A2313622, outstanding 9700
SCTP: Assoc 0: Bundling data, added 3/10502/100/A2313623, outstanding 9800
SCTP: Assoc 0: Bundling data, added 7/10482/100/A2313624, outstanding 9900
SCTP: Assoc 0: Bundling data, added 3/10503/100/A2313625, outstanding 10000
SCTP: Assoc 0: Bundling data, added 11/10403/100/A2313626, outstanding 10100
SCTP: Assoc 0: Bundled 5 chunk(s) in next dgram to 10.5.0.4
SCTP: Assoc 0: Mark chunk A23135C2 for retrans
SCTP: Assoc 0: Mark chunk A23135C3 for retrans
SCTP: Assoc 0: Mark chunk A23135C4 for retrans
SCTP: Assoc 0: Mark chunk A23135C5 for retrans
SCTP: Assoc 0: Mark chunk A23135C6 for retrans
SCTP: Assoc 0: Mark chunk A23135C7 for retrans
SCTP: Assoc 0: Mark chunk A23135C8 for retrans
SCTP: Assoc 0: Mark chunk A23135C9 for retrans
SCTP: Assoc 0: Mark chunk A23135CA for retrans
SCTP: Assoc 0: Bundled 6 chunk(s) in next dgram to 10.6.0.4
SCTP: Assoc 0: Mark chunk A23135C2 for retrans
SCTP: Assoc 0: Mark chunk A23135C3 for retrans
SCTP: Assoc 0: Mark chunk A23135C4 for retrans

```

Table 121 describes the significant fields shown in the display.

Table 121 *debug ip sctp sndchunks Command Field Descriptions*

Field	Description
0 / 10412 / 100 / A23134F8	Stream number / datagram sequence number / chunk length, in bytes / chunk transmission sequence number.
outstanding	Number of bytes outstanding to the specified destination address.
CumTSN	Cumulative transmission sequence number (TSN).
numFrag	Number of fragments sent.

Related Commands	Command	Description
	clear ip sctp statistics	Empties the buffer that holds SCTP statistics.
	debug ip sctp congestion	Shows a list of all current SCTP associations.
	show ip sctp association parameters	Shows the parameters configured for the association defined by the association identifier.
	show ip sctp association statistics	Shows the current statistics for the association defined by the association identifier.
	show ip sctp errors	Shows error counts logged by SCTP.
	show ip sctp instances	Shows all currently defined SCTP instances.
	show ip sctp statistics	Shows overall statistics counts for SCTP.
	show iua as	Shows information about the current condition of an application server.
	show iua asp	Shows information about the current condition of an application server process.

debug ip sctp state

To show state transitions in the Stream Control Transmission Protocol (SCTP), use the **debug ip sctp state** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip sctp state

no debug ip sctp state

Syntax Description This command has no arguments or keywords.

Defaults No default behavior or values.

Command Modes Privileged EXEC

Command History	Release	Modification
	12.2(4)T	This command was introduced.

Usage Guidelines The **debug ip sctp state** command can be used to see if the current associations are stable or not. Because it generates output only on state transitions, it is safe to use in a live environment. It still should be used with caution, however, depending on the number of associations being handled by the system and the stability of the network.

The **debug ip sctp state** command is often used at the same time as the **debug ip sctp signal** command. Using the two commands together gives good insight into the stability of associations.

Examples In the following example, a new association is requested and established. The peer then restarts the association and notes that the association failed and is being reestablished. The local peer then indicates that the association has failed because it has tried to retransmit the specified chunk more than the maximum number of times without success. As a result, the association fails (because of communication loss) and is terminated. The upper-layer protocol (ULP) requests that the association be attempted again, and this attempt succeeds. A shutdown is then received from the remote peer, and the local peer enters the shutdown acknowledge sent state, which is followed by the association being terminated. Again, another association attempt is made and succeeds.

```
Router# debug ip sctp signal
Router# debug ip sctp state
```

```
<new assoc attempt>
00:20:08: SCTP: Assoc 0: state CLOSED -> COOKIE_WAIT
00:20:15: SCTP: Assoc 0: state COOKIE_WAIT -> ESTABLISHED
00:20:15: SCTP: Assoc 0: Sent ASSOC_UP signal for CONFIGD_ASSOC
00:21:03: SCTP: Assoc 0: Restart rcvd from peer
00:21:03: SCTP: Assoc 0: Sent ASSOC_RESTART signal
00:21:04: SCTP: Assoc 0: chunk 62EA7F40 retransmitted more than max times, failing assoc
```

```

00:21:04: Sctp: Assoc 0: Sent ASSOC_FAILED signal, reason: Sctp_COMM_LOST
00:21:04: Sctp: Assoc 0: Sent ASSOC_TERMINATE signal
00:21:04: Sctp: Assoc 0: state ESTABLISHED -> CLOSED
<new assoc attempt>
00:21:04: Sctp: Assoc 0: state CLOSED -> COOKIE_WAIT
00:21:04: Sctp: Assoc 0: state COOKIE_WAIT -> COOKIE_ECHOED
00:21:04: Sctp: Assoc 0: state COOKIE_ECHOED -> ESTABLISHED
00:21:04: Sctp: Assoc 0: Sent ASSOC_UP signal for CONFIGD_ASSOC
00:21:04: Sctp: Assoc 0: Sent TERMINATE_PENDING signal
00:21:04: Sctp: Assoc 0: state ESTABLISHED -> SHUTDOWN_ACKSENT
00:21:04: Sctp: Assoc 0: Sent ASSOC_TERMINATE signal
00:21:04: Sctp: Assoc 0: state SHUTDOWN_ACKSENT -> CLOSED
<new assoc attempt>
00:21:04: Sctp: Assoc 0: state CLOSED -> COOKIE_WAIT
00:21:04: Sctp: Assoc 0: state COOKIE_WAIT -> COOKIE_ECHOED
00:21:04: Sctp: Assoc 0: state COOKIE_ECHOED -> ESTABLISHED
00:21:04: Sctp: Assoc 0: Sent ASSOC_UP signal for CONFIGD_ASSOC

```

Table 122 describes the significant fields shown in the display.

Table 122 *debug ip sctp state Command Field Descriptions*

Field	Description
CLOSED -> COOKIE_WAIT	SCTP endpoint sends initialization chunk and moves to the COOKIE-WAIT state to wait for acknowledgement and a state cookie from the remote endpoint.
COOKIE_WAIT -> COOKIE_ECHOED	SCTP endpoint returns the state cookie to the remote endpoint and enters COOKIE-ECHOED state.
COOKIE_ECHOED -> ESTABLISHED	SCTP endpoint enters ESTABLISHED state after receiving acknowledgement that the state cookie has been received by the remote endpoint.
ESTABLISHED -> SHUTDOWN_ACKSENT	SCTP endpoint enters SHUTDOWN-ACKSENT state after receiving a shutdown message and sending a shutdown acknowledgement to the remote endpoint.
SHUTDOWN_ACKSENT -> CLOSED	SCTP endpoint enters CLOSED state.

Related Commands

Command	Description
clear ip sctp statistics	Empties the buffer that holds SCTP statistics.
debug ip sctp congestion	Shows a list of all current SCTP associations.
debug ip sctp signal	Shows signals that are sent from SCTP to the application or ULP.
show ip sctp association parameters	Shows the parameters configured for the association defined by the association identifier.
show ip sctp association statistics	Shows the current statistics for the association defined by the association identifier.
show ip sctp errors	Shows error counts logged by SCTP.
show ip sctp instances	Shows all currently defined SCTP instances.
show ip sctp statistics	Shows overall statistics counts for SCTP.

Command	Description
show iua as	Shows information about the current condition of an application server.
show iua asp	Shows information about the current condition of an application server process.

debug ip sctp timer

To provide information about Stream Control Transmission Protocol (SCTP) timers that are started, stopped, and triggering, use the **debug ip sctp timer** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip sctp timer

no debug ip sctp timer

Syntax Description This command has no arguments or keywords.

Defaults No default behavior or values.

Command Modes Privileged EXEC

Command History	Release	Modification
	12.2(4)T	This command was introduced.

Usage Guidelines Many SCTP timers should not be restarted after they have been started once. For these timers, the first call succeeds in starting the timer, and subsequent calls do nothing until the timer either expires or is stopped. For example, the retransmission timer is started when the first chunk is sent, but then is not started again for subsequent chunks when there is outstanding data.



Caution

The **debug ip sctp timer** command generates a significant amount of output. It should be used with extreme caution in a live network.

Examples The following example shows the starting and stopping of various SCTP timers:

```
Router# debug ip sctp timer

SCTP: Assoc 0: Starting CUMSACK timer
SCTP: Timer already started, not restarting
SCTP: Assoc 0: Starting CUMSACK timer
SCTP: Timer already started, not restarting
SCTP: Assoc 0: Timer BUNDLE triggered
SCTP: Assoc 0: Starting RETRANS timer for destaddr 10.5.0.4
SCTP: Assoc 0: Starting RETRANS timer for destaddr 10.5.0.4
SCTP: Timer already started, not restarting
SCTP: Assoc 0: Starting RETRANS timer for destaddr 10.5.0.4
SCTP: Timer already started, not restarting
SCTP: Assoc 0: Starting RETRANS timer for destaddr 10.5.0.4
SCTP: Timer already started, not restarting
SCTP: Assoc 0: Stopping RETRANS timer for destaddr 10.5.0.4
SCTP: Assoc 0: Starting RETRANS timer for destaddr 10.5.0.4
```

```

SCTP: Assoc 0: Stopping RETRANS timer for destaddr 10.5.0.4
SCTP: Assoc 0: Starting CUMSACK timer
SCTP: Timer already started, not restarting
SCTP: Assoc 0: Starting CUMSACK timer
SCTP: Timer already started, not restarting
SCTP: Assoc 0: Starting CUMSACK timer
SCTP: Timer already started, not restarting
SCTP: Assoc 0: Starting CUMSACK timer
SCTP: Timer already started, not restarting
SCTP: Assoc 0: Starting CUMSACK timer
SCTP: Timer already started, not restarting
SCTP: Assoc 0: Stopping CUMSACK timer
SCTP: Assoc 0: Starting CUMSACK timer
SCTP: Assoc 0: Starting CUMSACK timer
SCTP: Timer already started, not restarting

```

Table 123 describes the significant fields shown in the display.

Table 123 *debug ip sctp timer Command Field Descriptions*

Field	Description
CUMSACK	Cumulative selective acknowledgement.
RETRANS	Retransmission.

Related Commands

Command	Description
clear ip sctp statistics	Empties the buffer that holds SCTP statistics.
debug ip sctp congestion	Shows a list of all current SCTP associations.
show ip sctp association parameters	Shows the parameters configured for the association defined by the association identifier.
show ip sctp association statistics	Shows the current statistics for the association defined by the association identifier.
show ip sctp errors	Shows error counts logged by SCTP.
show ip sctp instances	Shows all currently defined SCTP instances.
show ip sctp statistics	Shows overall statistics counts for SCTP.
show iua as	Shows information about the current condition of an application server.
show iua asp	Shows information about the current condition of an application server process.

debug ip sctp warnings

To display diagnostic information about unusual situations in Stream Control Transmission Protocol (SCTP), use the **debug ip sctp warnings** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip sctp warnings

no debug ip sctp warnings

Syntax Description This command has no arguments or keywords.

Defaults No default behavior or values.

Command Modes Privileged EXEC

Command History	Release	Modification
	12.2(4)T	This command was introduced.

Usage Guidelines In a live system, the debug messages for performance, state, signal, and warnings are the most useful. They show any association or destination address failures and can be used to monitor the stability of established associations.

The **debug ip sctp warnings** command displays information on any unusual situation that is encountered. These situations may or may not indicate problems, depending on the particulars of the situation.

Examples The following example shows some events and conditions that are flagged as warnings:

```
Router# debug ip sctp warnings

SCTP: Assoc 0: No cookie in InitAck, discarding
SCTP: Assoc 0: Incoming INIT_ACK: inbound streams req'd 15, allowed 13
SCTP: Assoc 0: Incoming INIT_ACK request: outbound streams req'd 13, allowed 1
SCTP: Assoc 0: Remote verification tag in init ack is zero, discarding
SCTP: Remote verification tag in init is zero, discarding
SCTP: Assoc 0: Rwnd less than min allowed (1500) in incoming INITACK, rcvd 0
SCTP: Assoc 0: Rwnd less than min allowed (1500) in incoming INITACK, rcvd 1499
SCTP: Rwnd in INIT too small (0), discarding
SCTP: Rwnd in INIT too small (1499), discarding
SCTP: Unknown INIT param 16537 (0x4099), length 8
SCTP: Assoc 0: Unknown INITACK param 153 (0x99), length 8
SCTP: Assoc 0: No cookie in InitAck, discarding
SCTP: Assoc 0: No cookie in InitAck, discarding
SCTP: Processing INIT, invalid param len 0, discarding...
SCTP: Assoc 0: Processing INITACK, invalid param len 0, discarding...
```

Related Commands

Command	Description
clear ip sctp statistics	Empties the buffer that holds Sctp statistics.
debug ip sctp congestion	Shows a list of all current Sctp associations.
show ip sctp association parameters	Shows the parameters configured for the association defined by the association identifier.
show ip sctp association statistics	Shows the current statistics for the association defined by the association identifier.
show ip sctp errors	Shows error counts logged by Sctp.
show ip sctp instances	Shows all currently defined Sctp instances.
show ip sctp statistics	Shows overall statistics counts for Sctp.
show iua as	Shows information about the current condition of an application server.
show iua asp	Shows information about the current condition of an application server process.

debug ip sd

To display all session directory (SD) announcements received, use the **debug ip sd** command in privileged EXEC mode. 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.

Command Modes Privileged EXEC

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

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

```
Router# debug ip sd

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

[Table 124](#) describes the significant fields in the output.

Table 124 *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.

Table 124 *debug ip sd Output Descriptions (continued)*

Field	Description
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** command in privileged EXEC mode. 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.

Command Modes Privileged EXEC

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



Caution

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

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

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

Table 125 describes significant fields shown in the output.

Table 125 *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 Modes EXEC

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

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 snat

To display information about IP packets translated by the IP stateful network address translation (SNAT) feature, use the **debug ip snat** command in privileged EXEC mode . To disable debugging output, use the **no** form of this command.

debug ip snat [detailed]

no debug ip snat [detailed]

Syntax Description

detailed (Optional) Displays debug information in a detailed format.

Defaults

This command is disabled by default.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(13)T	This command was introduced.

Usage Guidelines

The SNAT feature allows two or more network address translators to function as a translation group. One member of the translation group handles traffic requiring translation of IP address information. It informs the backup translator of active flows as they occur. The backup translator can then use information from the active translator to prepare duplicate translation table entries enabling the backup translator to become the active translator in the event of a critical failure. Traffic continues to flow without interruption since the same network address translations are used and the state of those translation has been previously defined.



Caution

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

Examples

The following is sample output from the **debug ip snat** command.

```
Router# debug ip snat detailed

2w6d:SNAT:Establish TCP peers for PRIMARY
2w6d:SNAT (Send):Enqueuing SYNC Message for Router-Id 100
2w6d:SNAT(write2net):192.168.123.2 <---> 192.168.123.3 send message
2w6d:SNAT(write2net):ver 2, id 100, opcode 1, len 68
2w6d:SNAT (Send):Enqueuing DUMP-REQUEST Message for Router-Id 100
2w6d:SNAT(write2net):192.168.123.2 <---> 192.168.123.3 send message
2w6d:SNAT(write2net):ver 2, id 100, opcode 6, len 68
2w6d:SNAT (readfromnet):Enqueuing SYNC Message msg to readQ
2w6d:SNAT (Receive):Processed SYNC Message from Router-Id:0 for Router-Id:200's
entry/entries
2w6d:SNAT (readfromnet):Enqueuing DUMP-REQUEST Message msg to readQ
```

```
try/entries
2w6d:SNAT(sense):Send SYNC message
2w6d:SNAT (Send):Enqueuing SYNC Message for Router-Id 100
2w6d:SNAT(write2net):192.168.123.2 <---> 192.168.123.3 send message
2w6d:SNAT(write2net):ver 2, id 100, opcode 1, len 68
2w6d:SNAT (readfromnet):Enqueuing SYNC Message msg to readQ
2w6d:SNAT (Receive):Processed SYNC Message from Router-Id:200 for Router-Id:200's
entry/entries
```

Table 126 describes the significant fields shown in the display.

Table 126 *debug ip snat Field Descriptions*

Field	Description
SNAT:	Indicates that the packet is being translated by the SNAT feature.
DUMP-REQUEST Message	Requests for entries after the SNAT router is active.

debug ip socket

To display all state change information for all sockets, use the **debug ip socket** command in privileged EXEC mode. 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.

Command Modes Privileged EXEC

Usage Guidelines Use this command to collect information on the socket interface. To get more complete information on a socket/TCP port pair, use this command in conjunction with the [debug ip tcp transactions](#) command. Because the socket debugging information is state change oriented, you will not see the 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 127 describes the significant fields shown in the display.

Table 127 *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 127 *debug ip socket Field Descriptions (continued)*

Field	Description
socket state: SS_ISCONNECTED SS_CANTSENDMORE SS_ISDISCONNECTING	<p>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.</p> <p>Possible SOCKET state flags follow:</p> <ul style="list-style-type: none"> • 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 Modes EXEC

Command History	Release	Modification
	12.0(5)S	This command was introduced.
	12.1(1)T	This command was integrated into Cisco IOS Release 12.1 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_MSG_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** command in privileged EXEC mode. 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.

Command Modes Privileged EXEC

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

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

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

```
Router# debug ip tcp driver

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

[Table 128](#) describes the significant fields shown in the display.

Table 128 *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.)

Table 128 *debug ip tcp driver Field Descriptions (continued)*

Field	Description
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** command in privileged EXEC mode. 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.

Command Modes Privileged EXEC

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

The TCP driver is the process that the router software uses to send packet data over a TCP connection. 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 129](#) describes the significant fields shown in the display.

Table 129 *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.

Table 129 *debug ip tcp driver-pak Field Descriptions (continued)*

Field	Description
(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** command in privileged EXEC mode. The **no** form of this command disables debugging output.

debug ip tcp intercept

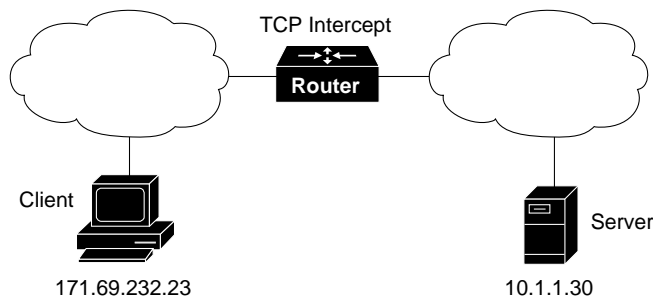
no debug ip tcp intercept

Syntax Description This command has no arguments or keywords.

Command Modes Privileged EXEC

Examples [Figure 3](#) illustrates a scenario in which a router configured with TCP intercept operates between a client and a server.

Figure 3 Example TCP Intercept Environment



171.69.232.23 sends all packets
172.19.160.17 does not exist

92895

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** command in privileged EXEC mode. 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.

Command Modes Privileged EXEC

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 130](#) describes the significant fields shown in the display.

Table 130 *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.

Table 130 *debug ip tcp transactions Field Descriptions (continued)*

Field	Description
TCP0:	Indicates the TTY number (0, in this case) with which this TCP connection is associated.
Connection to 10.9.0.13:22530	Indicates the remote address with which a connection has been established.
advertising MSS 966	Indicates the maximum segment size 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: <ul style="list-style-type: none"> • 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: <ul style="list-style-type: none"> • 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** command in privileged EXEC mode. 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.

Command Modes

Privileged EXEC

Usage Guidelines

Use this command when troubleshooting automated double authentication.

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

What is the remote host table? Whenever a remote user needs to be user-authenticated in the second stage of automated double authentication, the local device sends a 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 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

<i>hostname</i>	(Optional) The domain Name System (DNS) name.
<i>ip-address</i>	(Optional) The IP address.

Defaults

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

Command Modes

EXEC

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 urlfilter

To enable debug information of URL filter subsystems, use the **debug ip urlfilter** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip urlfilter { **function-trace** | **detailed** | **events** }

no debug ip urlfilter { **function-trace** | **detailed** | **events** }

Syntax Description	function-trace	The system prints a sequence of important functions that are called when configuring URL filtering.
	detailed	The system prints detailed information about various activities that occur during URL filtering.
	events	The system prints various events such as queue event, timer event, and socket event.

Defaults This command is not enabled.

Command Modes Privileged EXEC

Command History	Release	Modification
	12.2(11)YU	This command was introduced.
	12.2(15)T	This command was integrated into Cisco IOS Release 12.2(15)T.

Examples The following is sample output for the **debug ip urlfilter** command:

```
Router# debug ip urlfilter
urlfilter:
  Urlfilter Detailed Debugs debugging is on

Router# show ip urlfilter config

N2H2 URL Filtering is ENABLED

Primary N2H2 server configurations
=====
N2H2 server IP address:192.168.1.103
N2H2 server port:4005
N2H2 retransmission time out:6 (in seconds)
N2H2 number of retransmission:2

Secondary N2H2 servers configurations
=====
Other configurations
=====
Allow Mode:OFF
System Alert:ENABLED
Audit Trail:ENABLED
```

```

Log message on N2H2 server:DISABLED
Maximum number of cache entries:5
Maximum number of packet buffers:20
Maximum outstanding requests:1000
fwl_4#
1d15h:URLF:got a socket read event...
1d15h:URLF:socket recv failed.
1d15h:URLF:Closing the socket for server (192.168.1.103:4005)
1d15h:%URLF-3-SERVER_DOWN:Connection to the URL filter server 192.168.1.103 is down
1d15h:URLF:Opening a socket for server (192.168.1.103:4005)
1d15h:URLF:socket fd 0
1d15h:%URLF-5-SERVER_UP:Connection to an URL filter server(192.168.1.103) is made, the
router is returning from ALLOW MODE
1d15h:URLF:got cache idle timer event...
1d16h:URLF:got cache absolute timer event...
1d16h:URLF:got cache idle timer event...
1d16h:URLF:creating uis 0x63A95DB4, pending request 1
1d16h:URLF:domain name not found in the exclusive list
1d16h:URLF:got an cbac queue event...
1d16h:URLF:socket send successful...172.17.192.130:8080) -> 192.168.1.103:1052 seq
3344720064 wnd 24820
1d16h:URLF:holding pak 0x634A8A08 (172.17.192.130:8080) -> 192.168.1.103:1052 seq
3344721524 wnd 24820
1d16h:URLF:holding pak 0x634A98CC (172.17.192.130:8080) -> 192.168.1.103:1052 seq
3344722984 wnd 24820
1d16h:URLF:got a socket read event...
1d16h:URLF:socket recv (header) successful.
1d16h:URLF:socket recv (data) successful.
1d16h:URLF:n2h2 lookup code = 1
1d16h:URLF:Site/URL Blocked:sis 0x63675DC4, uis 0x63A95DB4
1d16h:%URLF-4-URL_BLOCKED:Access denied URL 'http://www.google.com/', client
192.168.1.103:1052 server 172.17.192.130:8080
1d16h:URLF:(192.168.1.103:1052) RST -> 172.17.192.130:8080 seq 3361738063 wnd 0
1d16h:URLF:(172.17.192.130:8080) FIN -> 192.168.1.103:1052 seq 3344720064 wnd 0
1d16h:URLF:deleting uis 0x63A95DB4, pending requests 0
1d16h:URLF:got cache idle timer event...
1d16h:URLF:creating uis 0x63A95DB4, pending request 1
1d16h:URLF:domain name not found in the exclusive list
1d16h:URLF:got an cbac queue event...
1d16h:URLF:socket send successfull...
1d16h:URLF:holding pak 0x634A812C (172.17.192.130:8080) -> 192.168.1.103:1101 seq
3589711120 wnd 24820
1d16h:URLF:holding pak 0x634A2E7C (172.17.192.130:8080) -> 192.168.1.103:1101 seq
3589712580 wnd 24820
1d16h:URLF:holding pak 0x634A3464 (172.17.192.130:8080) -> 192.168.1.103:1101 seq
3589714040 wnd 24820
1d16h:URLF:got a socket read event...
1d16h:URLF:socket recv (header) successful.
1d16h:URLF:socket recv (data) successful.
1d16h:URLF:n2h2 lookup code = 0
1d16h:%URLF-6-URL_ALLOWED:Access allowed for URL 'http://www.alcohol.com/', client
192.168.1.103:1101 server 172.17.192.130:8080
1d16h:URLF:Site/URL allowed:sis 0x6367D0C4, uis 0x63A95DB4
1d16h:URLF:releasing pak 0x634A812C:(172.17.192.130:8080) -> 192.168.1.103:1101 seq
3589711120 wnd 24820
1d16h:URLF:releasing pak 0x634A2E7C:(172.17.192.130:8080) -> 192.168.1.103:1101 seq
3589712580 wnd 24820
1d16h:URLF:releasing pak 0x634A3464:(172.17.192.130:8080) -> 192.168.1.103:1101 seq
3589714040 wnd 24820
1d16h:URLF:deleting uis 0x63A95DB4, pending requests 0
1d16h:URLF:got cache idle timer event...
1d16h:URLF:creating uis 0x63A9777C, pending request 1
1d16h:URLF:domain name not found in the exclusive list
1d16h:URLF:got an cbac queue event...

```

```
1d16h:URLF:socket send successful...
1d16h:URLF:got a socket read event...
1d16h:URLF:socket recv (header) successful.
1d16h:URLF:socket recv (data) successful.
1d16h:URLF:n2h2 lookup code = 1
1d16h:URLF:Site/URL Blocked:sis 0x63677ED4, uis 0x63A9777C
1d16h:%URLF-4-URL_BLOCKED:Access denied URL 'http://www.google.com/', client
192.168.1.103:1123 server 172.17.192.130:8080
1d16h:URLF:(192.168.1.103:1123) RST -> 172.17.192.130:8080 seq 3536466275 wnd 0
1d16h:URLF:(172.17.192.130:8080) FIN -> 192.168.1.103:1123 seq 3618929551 wnd 0
1d16h:URLF:deleting uis 0x63A9777C, pending requests 0
1d16h:URLF:got cache idle timer event...
```

debug ip wccp events

To display information about significant Web Cache Control Protocol (WCCP) events, use the **debug ip wccp events** command in privileged EXEC mode. 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.

Command Modes Privileged EXEC

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** command in privileged EXEC mode. 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.

Command Modes Privileged EXEC

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
```