

debug h225

To display additional information about the actual contents of H.225 Registration, Admission, and Status Protocol (RAS) messages, use the **debug h225** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

```
debug h225 {asn1 | events}
```

```
no debug h225
```

Syntax Description

asn1	Indicates that only the Abstract Syntax Notation One (ASN.1) contents of any H.225 message sent or received will be displayed.
events	Indicates that key Q.931 events that occur when placing an H.323 call from one gateway to another will be displayed.

Defaults

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
11.3(6)NA2	This command was introduced.
12.2(2)XB1	This command was implemented on the Cisco AS5850.
12.2(11)T	This command was integrated into Cisco IOS Release 12.2(11)T.

Usage Guidelines

Both versions of the **debug h225** command display information about H.225 messages. H.225 messages are used to exchange RAS information between gateways and gatekeepers as well as to exchange Q.931 information between gateways.

The **debug h225 events** command displays key Q.931 events that occur when placing an H.323 call from one gateway to another. Q.931 events are carried in H.225 messages. This command enables you to monitor Q.931 state changes such as setup, alert, connected, and released.



Note

Although the debug information includes the hexadecimal output of the entire H.225 message, only the key state changes are decoded.

The **debug h225 asn1** command displays the ASN.1 contents of any H.225 message sent or received that contains ASN.1 content. Not all H.225 messages contain ASN.1 content. Some messages contain both Q.931 information and ASN.1 information; if you enter this command, only ASN.1 information will be displayed.

Examples

The following sample output for the **debug h225 events** command shows a call being placed from gateway GW13 to gateway GW14. Before the call was placed, the gateway exchanged RAS messages with the gatekeeper. Because RAS messages do not contain Q.931 information, these messages do not appear in this output.

```
Router# debug h225 events
```

```
H.225 Event Messages debugging is on
```

```
Router#
```

```
*Mar 2 02:47:14.689:      H225Lib::h225TConn:connect in progress on socket [2]
*Mar 2 02:47:14.689:      H225Lib::h225TConn:Q.931 Call State is initialized to be
[Null].
*Mar 2 02:47:14.697:Hex representation of the SETUP TPKT to
send.0300004D080200DC05040380C0A36C0991313323313333303070099131342331343330307E00260500800
60008914A000102004B1F5E5D8990006C0000000005BF7454000C0700000000000000
*Mar 2 02:47:14.701:
*Mar 2 02:47:14.701:      H225Lib::h225SetupRequest:Q.931 SETUP sent from socket [2]
*Mar 2 02:47:14.701:      H225Lib::h225SetupRequest:Q.931 Call State changed to
[Call Initiated].
*Mar 2 02:47:14.729:Hex representation of the received
TPKT03000021080280DC013401017E0012050340060008914A000100000109350E2B28
*Mar 2 02:47:14.729:
*Mar 2 02:47:14.729:      H225Lib::h225RecvData:Q.931 ALERTING received from socket [2]
*Mar 2 02:47:14.729:      H225Lib::h225RecvData:Q.931 Call State changed to
[Call Delivered].
*Mar 2 02:47:17.565:Hex representation of the received
TPKT03000034080280DC07040380C0A37E0023050240060008914A0001000109350E2B2802004B1F5E5D899000
6C0000000005BF7454
*Mar 2 02:47:17.569:
*Mar 2 02:47:17.569:      H225Lib::h225RecvData:Q.931 CONNECT received from socket [2]
*Mar 2 02:47:17.569:      H225Lib::h225RecvData:Q.931 Call State changed to [Active].
*Mar 2 02:47:23.273:Hex representation of the received
TPKT0300001A080280DC5A080280107E000A050500060008914A0001
*Mar 2 02:47:23.273:
*Mar 2 02:47:23.273:      H225Lib::h225RecvData:Q.931 RELEASE COMPLETE received from
socket [2]
*Mar 2 02:47:23.273:      H225Lib::h225RecvData:Q.931 Call State changed to [Null].
*Mar 2 02:47:23.293:Hex representation of the RELEASE COMPLETE TPKT
to send.0300001A080200DC5A080280107E000A050500060008914A0001
*Mar 2 02:47:23.293:
*Mar 2 02:47:23.293:      H225Lib::h225TerminateRequest:Q.931 RELEASE COMPLETE sent from
socket [2]. Call state changed to [Null].
*Mar 2 02:47:23.293:      H225Lib::h225TClose:TCP connection from socket [2] closed
```

The following output shows the same call being placed from gateway GW13 to gateway GW14 using the **debug h225 asn1** command. The output is very long, but you can track the following information:

- The admission request to the gatekeeper.
- The admission confirmation from the gatekeeper.
- The ASN.1 portion of the H.225/Q.931 setup message from the calling gateway to the called gateway.
- The ASN.1 portion of the H.225/Q.931 setup response from the called gateway, indicating that the call has proceeded to alerting state.
- The ASN.1 portion of the H.225/Q.931 message from the called gateway, indicating that the call has been connected.
- The ASN.1 portion of the H.225/Q.931 message from the called gateway, indicating that the call has been released.

- The ANS.1 portion of the H.225 RAS message from the calling gateway to the gatekeeper, informing it that the call has been disengaged.
- The ASN.1 portion of the H.225 RAS message from the gatekeeper to the calling gateway, confirming the disengage request.
- The ASN.1 portion of the H.225/Q.931 release complete message sent from the called gateway to the calling gateway.

```
Router# debug h225 asn1
```

```
H.225 ASN1 Messages debugging is on
```

```
Router#
```

```
value RasMessage ::= admissionRequest :
*Mar 2 02:48:18.445: {
*Mar 2 02:48:18.445:   requestSeqNum 03320,
*Mar 2 02:48:18.445:   callType pointToPoint :NULL,
*Mar 2 02:48:18.445:   callModel direct :NULL,
*Mar 2 02:48:18.445:   endpointIdentifier "60D6BA4C00000001",
*Mar 2 02:48:18.445:   destinationInfo
*Mar 2 02:48:18.445:   {
*Mar 2 02:48:18.445:     e164 :"14#14300"
*Mar 2 02:48:18.445:   },
*Mar 2 02:48:18.449:   srcInfo
*Mar 2 02:48:18.449:   {
*Mar 2 02:48:18.449:     e164 :"13#13300"
*Mar 2 02:48:18.449:   },
*Mar 2 02:48:18.449:   bandwidth 0640,
*Mar 2 02:48:18.449:   callReferenceValue 0224,
*Mar 2 02:48:18.449:   conferenceID '4B1F5E5D899000720000000005C067A4'H,
*Mar 2 02:48:18.449:   activeMC FALSE,
*Mar 2 02:48:18.449:   answerCall FALSE
*Mar 2 02:48:18.449: }
*Mar 2 02:48:18.449:25800CF7 00F00036 00300044 00360042 00410034 00430030 00300030
00300030
00300030 00310103 80470476 33010380 46046633 40028000 E04B1F5E 5D899000
72000000 0005C067 A400
29000CF7 40028000 0109350E 06B80077
value RasMessage ::= admissionConfirm :
*Mar 2 02:48:18.469: {
*Mar 2 02:48:18.469:   requestSeqNum 03320,
*Mar 2 02:48:18.469:   bandwidth 0640,
*Mar 2 02:48:18.469:   callModel direct :NULL,
*Mar 2 02:48:18.469:   destCallSignalAddress ipAddress :
*Mar 2 02:48:18.469:   {
*Mar 2 02:48:18.469:     ip '0109350E'H,
*Mar 2 02:48:18.469:     port 01720
*Mar 2 02:48:18.469:   },
*Mar 2 02:48:18.469:   irrFrequency 0120
*Mar 2 02:48:18.473: }
*Mar 2 02:48:18.473:value H323-UserInformation ::=
*Mar 2 02:48:18.481:{
*Mar 2 02:48:18.481:  h323-uu-pdu
*Mar 2 02:48:18.481:  {
*Mar 2 02:48:18.481:    h323-message-body setup :
*Mar 2 02:48:18.481:    {
*Mar 2 02:48:18.481:      protocolIdentifier { 0 0 8 2250 0 1 },
*Mar 2 02:48:18.481:      sourceInfo
*Mar 2 02:48:18.481:      {
*Mar 2 02:48:18.481:        terminal
*Mar 2 02:48:18.481:        {
*Mar 2 02:48:18.481:          },

```

```

*Mar 2 02:48:18.481:          mc FALSE,
*Mar 2 02:48:18.481:          undefinedNode FALSE
*Mar 2 02:48:18.481:        },
*Mar 2 02:48:18.481:        activeMC FALSE,
*Mar 2 02:48:18.481:        conferenceID '4B1F5E5D899000720000000005C067A4'H,
*Mar 2 02:48:18.481:        conferenceGoal create :NULL,
*Mar 2 02:48:18.485:        callType pointToPoint :NULL,
*Mar 2 02:48:18.485:        sourceCallSignalAddress ipAddress :
*Mar 2 02:48:18.485:          {
*Mar 2 02:48:18.485:            ip '00000000'H,
*Mar 2 02:48:18.485:            port 00
*Mar 2 02:48:18.485:          }
*Mar 2 02:48:18.485:        }
*Mar 2 02:48:18.485:      }
*Mar 2 02:48:18.485:}
*Mar 2 02:48:18.485:00800600 08914A00 0102004B 1F5E5D89 90007200 00000005 C067A400
0C070000
00000000 00
value H323-UserInformation ::=
*Mar 2 02:48:18.525:{
*Mar 2 02:48:18.525:  h323-uu-pdu
*Mar 2 02:48:18.525:  {
*Mar 2 02:48:18.525:    h323-message-body alerting :
*Mar 2 02:48:18.525:    {
*Mar 2 02:48:18.525:      protocolIdentifier { 0 0 8 2250 0 1 },
*Mar 2 02:48:18.525:      destinationInfo
*Mar 2 02:48:18.525:      {
*Mar 2 02:48:18.525:        mc FALSE,
*Mar 2 02:48:18.525:        undefinedNode FALSE
*Mar 2 02:48:18.525:      },
*Mar 2 02:48:18.525:      h245Address ipAddress :
*Mar 2 02:48:18.525:      {
*Mar 2 02:48:18.525:        ip '0109350E'H,
*Mar 2 02:48:18.525:        port 011050
*Mar 2 02:48:18.525:      }
*Mar 2 02:48:18.525:    }
*Mar 2 02:48:18.525:  }
*Mar 2 02:48:18.525:}
*Mar 2 02:48:18.525:value H323-UserInformation ::=
*Mar 2 02:48:22.753:{
*Mar 2 02:48:22.753:  h323-uu-pdu
*Mar 2 02:48:22.753:  {
*Mar 2 02:48:22.753:    h323-message-body connect :
*Mar 2 02:48:22.753:    {
*Mar 2 02:48:22.753:      protocolIdentifier { 0 0 8 2250 0 1 },
*Mar 2 02:48:22.753:      h245Address ipAddress :
*Mar 2 02:48:22.753:      {
*Mar 2 02:48:22.753:        ip '0109350E'H,
*Mar 2 02:48:22.753:        port 011050
*Mar 2 02:48:22.753:      },
*Mar 2 02:48:22.753:      destinationInfo
*Mar 2 02:48:22.753:      {
*Mar 2 02:48:22.753:        terminal
*Mar 2 02:48:22.753:        {
*Mar 2 02:48:22.753:          },
*Mar 2 02:48:22.757:        mc FALSE,
*Mar 2 02:48:22.757:        undefinedNode FALSE
*Mar 2 02:48:22.757:      },
*Mar 2 02:48:22.757:      conferenceID '4B1F5E5D899000720000000005C067A4'H
*Mar 2 02:48:22.757:    }
*Mar 2 02:48:22.757:  }
*Mar 2 02:48:22.757:}
*Mar 2 02:48:22.757:value H323-UserInformation ::=
*Mar 2 02:48:27.109:{

```

```

*Mar 2 02:48:27.109: h323-uu-pdu
*Mar 2 02:48:27.109: {
*Mar 2 02:48:27.109:   h323-message-body releaseComplete :
*Mar 2 02:48:27.109:     {
*Mar 2 02:48:27.109:       protocolIdentifier { 0 0 8 2250 0 1 }
*Mar 2 02:48:27.109:     }
*Mar 2 02:48:27.109:   }
*Mar 2 02:48:27.109:}
*Mar 2 02:48:27.109:value RasMessage ::= disengageRequest :
*Mar 2 02:48:27.117: {
*Mar 2 02:48:27.117:   requestSeqNum 03321,
*Mar 2 02:48:27.117:   endpointIdentifier "60D6BA4C00000001",
*Mar 2 02:48:27.117:   conferenceID '4B1F5E5D899000720000000005C067A4'H,
*Mar 2 02:48:27.121:   callReferenceValue 0224,
*Mar 2 02:48:27.121:   disengageReason normalDrop :NULL
*Mar 2 02:48:27.121: }
*Mar 2 02:48:27.121:3C0CF81E 00360030 00440036 00420041 00340043 00300030 00300030
00300030
00300031 4B1F5E5D 89900072 00000000 05C067A4 00E020
400CF8
value RasMessage ::= disengageConfirm :
*Mar 2 02:48:27.133: {
*Mar 2 02:48:27.133:   requestSeqNum 03321
*Mar 2 02:48:27.133: }
*Mar 2 02:48:27.133:value H323-UserInformation ::=
*Mar 2 02:48:27.133:{
*Mar 2 02:48:27.133: h323-uu-pdu
*Mar 2 02:48:27.133: {
*Mar 2 02:48:27.133:   h323-message-body releaseComplete :
*Mar 2 02:48:27.133:     {
*Mar 2 02:48:27.133:       protocolIdentifier { 0 0 8 2250 0 1 }
*Mar 2 02:48:27.133:     }
*Mar 2 02:48:27.133:   }
*Mar 2 02:48:27.133: }
*Mar 2 02:48:27.133:}
*Mar 2 02:48:27.133:05000600 08914A00 01

```

debug h245 asn1

To display Abstract Syntax Notation One (ASN.1) contents of H.245 messages, use the **debug h245 asn1** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug h245 asn1

no debug h245 asn1

Syntax Description This command has no arguments or keywords.

Command Modes Privileged EXEC

Command History	Release	Modification
	11.3(2)NA	This command was introduced.
	12.0(3)T	This command was integrated into Cisco IOS Release 12.0(3)T.

Usage Guidelines



Caution

This command slows the system down considerably. Connections may time out.

debug h245 events

To display H.245 events, use the **debug h245 events** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug h245 events

no debug h245 events

Syntax Description This command has no arguments or keywords.

Command Modes Privileged EXEC

Command History	Release	Modification
	11.3(2)NA	This command was introduced.
	12.0(3)T	This command was integrated into Cisco IOS Release 12.0(3)T.

debug h323-annexg

To display all pertinent Annex G messages that have been transmitted and received, use the **debug h323-annexg** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

```
debug h323-annexg {asn1 | errors | events | inout}
```

```
no debug h323-annexg
```

Syntax Description.

asn1	Displays the Annex G ASN.1 messages.
errors	Displays the Annex G error messages encountered during processing.
events	Displays the Annex G events received from the state machine.
inout	(Optional) This functionality is not yet implemented.

Defaults

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(2)XA	This command was introduced.
12.2(4)T	This command was integrated into Cisco IOS Release 12.2(4)T.
12.2(2)XB1	This command was implemented on the Cisco AS5850.
12.2(11)T	This command was implemented on the Cisco AS5850.

Examples

The following is sample output from the **debug h323-annexg events** command:

```
Router# debug h323-annexg events

Aug 16 14:03:40.983:be_process:BE QUEUE_EVENT (minor 73) wakeup
Aug 16 14:03:40.983:be_sm:Received event BE_EV_DO_QUERY
Aug 16 14:03:40.983:-<- query_neighbor:Sent descriptorIDRequest to
172.18.195.46:2099 [320]
Aug 16 14:03:40.983:be_sm:Started query-timer of 1 minutes for
neighbor at 172.18.195.46
Aug 16 14:03:40.991:-> nxd_recv_msg:Rcvd dscrptrIDCnfrmtn from
172.18.195.46:2099 [320]
Aug 16 14:03:41.531:-<- send_descriptor_request:Sent descriptorRequest
to 172.18.195.46:2099 [321]
Aug 16 14:03:41.539:-> nxd_recv_msg:Rcvd descriptorConfirmation from
172.18.195.46:2099 [321]
Aug 16 14:03:41.539:handle_descriptor_cfm:Descriptor from neighbor
172.18.195.46 unchanged, TTL is 60 Seconds
```

Related Commands

Command	Description
emulate	Displays all pertinent Annex E messages that have been transmitted and received.

debug hpi



Note

Effective with release 12.3(8)T, the **debug hpi** command is replaced by the **debug voip hpi** command. See the **debug voip hpi** command for more information.

To enable debugging for Host Port Interface (HPI) message events, use the **debug hpi** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

```
debug hpi { all | buffer size | capture | command | destination url | detail | error | notification | response | stats }
```

```
no debug hpi { all | buffer size | capture | command | destination url | detail | error | notification | response | stats }
```

Syntax Description

all	Enables all HPI debug options (command, detail, error, notification, and response).
buffer <i>size</i>	Sets the maximum amount of memory (in bytes) that the capture system allocates for its buffers when it is active. Valid size range is from 0 to 9000000. Default is 0.
capture	Displays HPI capture.
command	Displays commands that are being sent to the 54x DSP.
destination <i>url</i>	Turns capture on if it was off and sends the output to the specified URL. If capture was previously enabled for a different URL, the existing URL is closed, the new URL is opened, and output is sent to the new URL.
detail	Displays additional detail for the HPI debugs that are enabled.
error	Displays any HPI errors.
notification	Displays notification messages sent that are from the 54x DSP (for example, tone detection notification).
response	Displays responses (to commands) that are sent by the 54x DSP (for example, responses to statistic requests).
stats	Displays HPI statistics.

Defaults

This command is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.1(5)XM	This command was introduced on the Cisco AS5300 and Cisco AS5800.
12.2(2)T	This command was implemented on the Cisco 1700, Cisco 2600 series, Cisco 3600 series, and the Cisco MC3810. The stats keyword was added.

Release	Modification
12.2(10), 12.2(11)T	This command was implemented on the Cisco 827, Cisco 2400, Cisco 7200 series, and Cisco CVA 120. The following keywords were added: buffer , capture , and destination .
12.3(8)T	This command was replaced by the debug voip hpi command.

Usage Guidelines

This command enables debugging for HPI message events, which are used to communicate with digital signal processors (DSPs).

When used with the Voice DSP Control Message Logger feature, the **debug hpi buffer** command sets the maximum amount of memory (in bytes) that the capture system can allocate for its buffers when it is active. The **debug hpi capture destination url** command turns capture on if it was off and sends the output to the given URL. If capture was previously enabled for a different URL, the existing URL is closed, the new URL is opened, and output is sent to the new URL.

When you use the **no debug hpi capture** command, the capture option is turned off if it was on, any open files are closed, and any allocated memory is released.

Use the **debug hpi all** command to view gateway DSP modem relay termination codes. The DSP-to-host messages for the modem relay termination indicate to the host the modem relay session termination time, physical or link layer, and other probable causes for disconnection. On receiving this indication from the DSP, the host can disconnect the call or place the channel in the modem passthrough state.

When this command is used on a Cisco AS5300 during a calling session, the Cisco AS5300 displays the following information (of severity 6 whereas ordinary debug information is severity 7) on the screen by default:

```
2w6d:%ISDN-6-DISCONNECT:Interface Serial0:18 disconnected from 22022 , call lasted 12
seconds
2w6d:%ISDN-6-DISCONNECT:Interface Serial1:9 disconnected from 32010 , call lasted 14
seconds
2w6d:%ISDN-6-CONNECT:Interface Serial3:2 is now connected to 52003
2w6d:%ISDN-6-CONNECT:Interface Serial2:11 is now connected to 42002
```

To disable this default information on the Cisco AS5300 and to block the display of the **debug hpi capture** and **show voice hpi capture** commands, set the login console to a severity lower than 6.

Examples

The following example turns on the debug output from capture routines:

```
Router# debug hpi capture

HPI Capture/Logger debugging is on
```

Related Commands

Command	Description
show voice hpi capture	Verifies capture status and statistics.

debug http client

To display debugging messages for the HTTP client, use the **debug http client** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

```
debug http client {all | api | background | cache | error | main | msg | socket}
```

```
no debug http client {all | api | background | cache | error | main | msg | socket}
```

Syntax Description

all	Displays all debugging messages for the HTTP client.
api	Displays debugging information for the HTTP client application programming interface (API) process.
background	Displays background messages.
cache	Displays debugging information for the HTTP client cache module.
error	Displays the HTTP client error messages.
main	Displays debugging information for the HTTP client main process.
msg	Displays the HTTP client messages.
socket	Displays the HTTP client socket messages.

Defaults

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(2)XB	This command was introduced on the Cisco AS5300, Cisco AS5350, and Cisco AS5400.
12.2(11)T	This command was implemented on the Cisco 3640 and Cisco 3660, and the background keyword was added.

Usage Guidelines

The output of this command is effected by the **debug condition application voice** command. If the **debug condition application voice** command is configured and the <cisco-debug> element is enabled in the VoiceXML document, debugging output is limited to the VoiceXML application named in the **debug condition application voice** command.



Note

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

Examples

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

```
Router# debug http client api

HTTP Client API Process debugging is on

*Jan 3 10:58:48.609: httpc_send_ev: event sent to HTTP Client:
*Jan 3 10:58:48.609:     method (GET), url (http://serverX.com/vxml/test/prompts/9.au)
*Jan 3 10:58:48.609:     callback (61008E78), argp (63590DB4), sid (0), timeout (60),
retries (2)
*Jan 3 10:58:48.609: httpc_free: app freeing response data(626FA608)
*Jan 3 10:58:59.353: httpc_send_ev: event sent to HTTP Client:
*Jan 3 10:58:59.353:     method (GET), url (http://1.7.100.1/vxml/test/dropoffRecord)
*Jan 3 10:58:59.353:     callback (61008E78), argp (6393B684), sid (0), timeout (60),
retries (0)
*Jan 3 10:58:59.369: httpc_free: app freeing response data(626F9348)
*Jan 3 10:59:45.033: httpc_send_ev: event sent to HTTP Client:
*Jan 3 10:59:45.033:     method (POST), url
(http://rtsp-ws/dropoffAppend.php?append=&disconnect=1)
*Jan 3 10:59:45.033:     callback (60FE9064), argp (63448820), sid (7179), timeout (0),
retries (0)
*Jan 3 10:59:57.369: httpc_free: app freeing response data(626F9340)
```

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

```
Router# debug http client cache

HTTP Client Cache Module debugging is on

*Jan 3 11:53:52.817: httpc_cache_rsp_return:
cache(626F8E50)URL:http://serverX.com/vxml/test/root.vxml
*Jan 3 11:53:52.829: httpc_cache_entry_free:
cache(626F8B30)URL:http://serverX.com/vxml/test/getPhoneInfo.vxml?ani=1234567&dnis=7654321
*Jan 3 11:53:52.837: httpc_cache_entry_free:
cache(626F9710)URL:http://1.7.100.1/vxml/test/engine.vxml?flow=iso
*Jan 3 11:53:52.853: httpc_cache_rsp_return:
cache(626F8B30)URL:http://1.7.100.1/vxml/test/root.vxml
*Jan 3 11:53:52.873: httpc_cache_rsp_return:
cache(626F9030)URL:http://1.7.100.1/vxml/test/getExtension.vxml
*Jan 3 11:53:59.517: httpc_cache_entry_free:
cache(626F9170)URL:http://1.7.100.1/vxml/test/checkExtension.vxml?extension=1234&attempt=1
*Jan 3 11:53:59.545: httpc_cache_rsp_return:
cache(626F9A30)URL:http://1.7.100.1/vxml/test/dropoff.vxml
*Jan 3 11:54:10.361: httpc_cache_rsp_return:
cache(626F9DF0)URL:http://serverX.com/vxml/test/init.vxml
*Jan 3 11:54:10.361: httpc_cache_rsp_return:
cache(626FA430)URL:http://1.7.100.1/vxml/test/dropoffRecord
*Jan 4 00:20:23.474: httpc_cache_store: entry(http://ServerY.com/vxml/init.vxml)
size(10114 bytes) is too large to cache.
```

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

```
Router# debug http client main

HTTP Client Main Process debugging is on

*Jan 3 11:56:05.885: httpc_get, url: http://serverX.com/vxml/test/root.vxml
*Jan 3 11:56:05.889: httpc_msg_send, sid: 0, method: 83951618
*Jan 3 11:56:05.889: httpc_enqueue_wmsg, sid: 0, method: 83951618
*Jan 3 11:56:05.893: httpc_process_write_queue, socket writeable fd: 0, process enqueued
msg, sid: 0, method: 83951618
*Jan 3 11:56:05.893: httpc_msg_write, sid: 0, method: 83951618
*Jan 3 11:56:05.901: HTTPC_MSG_COMPLETE:
rsp_code(304),msg(62C9C25C)URL:http://serverX.com/vxml/test/root.vxml, fd(0)
```

```

*Jan 3 11:56:05.901: httpc_process_redirect_rsp:
msg(62C9C25C)URL:http://serverX.com/vxml/test/root.vxml, response code
HTTPC_NOT_MODIFIED_304
*Jan 3 11:56:05.913: httpc_get, url:
http://serverX.com/vxml/test/getPhoneInfo.vxml?ani=1234567&dnis=7654321
*Jan 3 11:56:05.917: httpc_msg_send, sid: 0, method: 65538
*Jan 3 11:56:05.917: httpc_enqueue_wmsg, sid: 0, method: 65538
*Jan 3 11:56:05.917: httpc_process_write_queue, socket writeble fd: 1, process enqueued
msg,
                                sid: 0, method: 65538
*Jan 3 11:56:05.917: httpc_msg_write, sid: 0, method: 65538
*Jan 3 11:56:05.925: HTTPC_MSG_COMPLETE:
rsp_code(200),msg(62CB5824)URL:http://serverX.com/vxml/test/getPhoneInfo.vxml?ani=1234567&
dnis=7654321, fd(1)
*Jan 3 11:56:05.929: httpc_get, url: http://1.7.100.1/vxml/test/engine.vxml?flow=iso
*Jan 3 11:56:05.929: httpc_msg_send, sid: 0, method: 65538
*Jan 3 11:56:05.929: httpc_enqueue_wmsg, sid: 0, method: 65538
*Jan 3 11:56:05.929: httpc_process_free_rsp: User returns noncache response (626F9670)
*Jan 3 11:56:05.929: httpc_process_write_queue, socket writeble fd: 1, process enqueued
msg,
                                sid: 0, method: 65538
*Jan 3 11:56:05.929: httpc_msg_write, sid: 0, method: 65538
*Jan 3 11:56:05.937: HTTPC_MSG_COMPLETE:
rsp_code(200),msg(62CB03AC)URL:http://1.7.100.1/vxml/test/engine.vxml?flow=iso, fd(1)

```

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

```
Router# debug http client msg
```

```
HTTP Client:
```

```
HTTP Client Messages debugging is on
```

```

*Jan 1 05:07:30.534: HTTP Client write buffer fd(0):
GET /vxml/abcdefg/test/init.vxml HTTP/1.1
Host: c5300-2
Content-Type: application/x-www-form-urlencoded
Connection: Keep-Alive
Accept: text/vxml; level = 1, text/plain, text/html, audio/basic
User-Agent: Cisco-IOS-C5300/12.2(20010829:180555) VoiceXML/1.0

```

```

*Jan 1 05:07:30.538: about to send data to socket 0 :
first 263 bytes of data:

```

```

62397130:                                47455420                GET
62397140: 2F76786D 6C2F6162 63646566 672F7465 /vxml/abcdefg/te
62397150: 73742F69 6E69742E 76786D6C 20485454 st/init.vxml HTT
62397160: 502F312E 310D0A48 6F73743A 20633533 P/1.1..Host: c53
62397170: 30302D32 0D0A436F 6E74656E 742D5479 00-2..Content-Ty
62397180: 70653A20 6170706C 69636174 696F6E2F pe: application/
62397190: 782D7777 772D666F 726D2D75 726C656E x-www-form-urle
623971A0: 636F6465 640D0A43 6F6E6E65 6374696F coded..Connectio
623971B0: 6E3A204B 6565702D 416C6976 650D0A41 n: Keep-Alive..A
623971C0: 63636570 743A2074 6578742F 76786D6C ccept: text/vxml
623971D0: 3B206C65 76656C20 3D20312C 20746578 ; level = 1, tex
623971E0: 742F706C 61696E2C 20746578 742F6874 t/plain, text/ht
623971F0: 6D6C2C20 61756469 6F2F6261 7369630D ml, audio/basic.
62397200: 0A557365 722D4167 656E743A 20436973 .User-Agent: Cis
62397210: 636F2D49 4F532D43 35333030 2F31322E co-IOS-C5300/12.
62397220: 32283230 30313038 32393A31 38303535 2(20010829:18055
62397230: 35292056 6F696365 584D4C2F 312E300D 5) VoiceXML/1.0.
62397240: 0A0D0A00                ....

```

```

*Jan 1 05:07:30.546: read data from socket 0 :
first 400 bytes of data:

```

```

628DE8F0:          48545450 2F312E31 20323030      HTTP/1.1 200
628DE900: 204F4B0D 0A446174 653A2046 72692C20      OK..Date: Fri,
628DE910: 33312041 75672032 30303120 30373A30      31 Aug 2001 07:0
628DE920: 363A3335 20474D54 0D0A5365 72766572      6:35 GMT..Server
628DE930: 3A204170 61636865 2F312E33 2E313120      : Apache/1.3.11
628DE940: 28556E69 78292041 70616368 654A5365      (Unix) ApacheJSe
628DE950: 72762F31 2E300D0A 4C617374 2D4D6F64      rv/1.0..Last-Mod
628DE960: 69666965 643A2057 65642C20 3233204D      ified: Wed, 23 M
628DE970: 61792032 30303120 31353A35 333A3233      ay 2001 15:53:23
628DE980: 20474D54 0D0A4554 61673A20 22323430      GMT..ETag: "240
628DE990: 37642D31 39322D33 62306264 63663322      7d-192-3b0bdcf3"
628DE9A0: 0D0A4163 63657074 2D52616E 6765733A      ..Accept-Ranges:
628DE9B0: 20627974 65730D0A 436F6E74 656E742D      bytes..Content-
628DE9C0: 4C656E67 74683A20 3430320D 0A4B6565      Length: 402..Kee
628DE9D0: 702D416C 6976653A 2074696D 656F7574      p-Alive: timeout
628DE9E0: 3D352C20 6D61783D 31300D0A 436F6E6E      =5, max=10..Conn
628DE9F0: 65637469 6F6E3A20 4B656570 2D416C69      ection: Keep-Ali
628DEA00: 76650D0A 436F6E74 656E742D 54797065      ve..Content-Type
628DEA10: 3A207465 78742F76 786D6C0D 0A0D0A3C      : text/vxml....<
628DEA20: 3F786D6C 20766572 73696F6E 3D22312E      ?xml version="1.
628DEA30: 30223F3E 0A3C7678 6D6C2076 65727369      0"?>.<vxml versi
628DEA40: 6F6E3D22 312E3022 20617070 6C696361      on="1.0" applica
628DEA50: 74696F6E 3D22726F 6F742E76 786D6C22      tion="root.vxml"
628DEA60: 3E0A2020 3C666F72 6D3E0A20 2020203C      >.<form>.<
628DEA70: 626C6F63 6B3E0A20 20202020 203C212D      block>.<!--
628DEA80: 2D0A2020 20      -.
*Jan 1 05:07:30.550: httpc_decode_msgheader: Client ignores this header: Server:
Apache/1.3.11 (Unix) ApacheJServ/1.0
*Jan 1 05:07:30.554: httpc_decode_msgheader: Client ignores this header: Accept-Ranges:
bytes
*Jan 1 05:07:30.554: processing server rsp msg:
msg(62C711C4)URL:http://vww.com/vxml/abcdefg/test/init.vxml, fd(0):
*Jan 1 05:07:30.554: Request msg: GET /vxml/abcdefg/test/init.vxml HTTP/1.1
*Jan 1 05:07:30.554: Message Response Code: 200
*Jan 1 05:07:30.554: Message Rsp Decoded Headers:
*Jan 1 05:07:30.554: Date:Fri, 31 Aug 2001 07:06:35 GMT
*Jan 1 05:07:30.554: Content-Length:402
*Jan 1 05:07:30.554: Content-Type:text/vxml
*Jan 1 05:07:30.554: ETag:"2407d-192-3b0bdcf3"
*Jan 1 05:07:30.554: Last-Modified:Wed, 23 May 2001 15:53:23 GMT
*Jan 1 05:07:30.554: Connection:Keep-Alive
*Jan 1 05:07:30.554: Keep-Alive:timeout=5, max=10
*Jan 1 05:07:30.554: httpc_dump_msg: headers:
*Jan 1 05:07:30.554: HTTP/1.1 200 OK
Date: Fri, 31 Aug 2001 07:06:35 GMT
Server: Apache/1.3.11 (Unix) ApacheJServ/1.0
Last-Modified: Wed, 23 May 2001 15:53:23 GMT
ETag: "2407d-192-3b0bdcf3"
Accept-Ranges: bytes
Content-Length: 402
Keep-Alive: timeout=5, max=10
Connection: Keep-Alive
Content-Type: text/vxml

*Jan 1 05:07:30.558: httpc_dump_msg: body:
*Jan 1 05:07:30.558: <?xml version="1.0"?>
<vxml version="1.0" application="root.vxml">
  <form>
    <block>
      <!--
      <var name="ani" expr="session.telephone.ani"/>
      <var name="dnis" expr="session.telephone.dnis"/>
      -->
      <var name="ani" expr="1234567"/>

```

```

        <var name="dnis" expr="7654321"/>
        <submit next="getPhoneInfo.vxml" method="get" namelist="ani dnis"/>
    </block>
</form>
</vxml>

```

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

```
Router# debug http client socket
```

```
HTTP Client Sockets debugging is on
```

```

*Jan  3 11:32:38.353: httpc_process_read_ev: HTTPC SOCK_PENDING --> SOCK_CONNECTED fd(0)
port(80)
*Jan  3 11:32:38.377: httpc_process_read_ev: HTTPC SOCK_PENDING --> SOCK_CONNECTED fd(1)
port(80)
*Jan  3 11:32:38.381: httpc_socket_cleanup: fd(1)
*Jan  3 11:32:38.389: httpc_process_read_ev: HTTPC SOCK_PENDING --> SOCK_CONNECTED fd(1)
port(80)
*Jan  3 11:32:38.393: httpc_socket_cleanup: fd(1)
*Jan  3 11:32:38.397: httpc_process_read_ev: HTTPC SOCK_PENDING --> SOCK_CONNECTED fd(1)
port(80)
*Jan  3 11:32:40.361: httpc_socket_cleanup: fd(0)
*Jan  3 11:32:40.413: httpc_socket_cleanup: fd(1)
*Jan  3 11:40:43.557: httpc_socket_connect failed fd(2)

```

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

```
Router# debug http client error
```

```
HTTP Client Error debugging is on
```

```

*Jan  3 12:07:40.209: HTTPC URL:http://serverX.com/vxml/test.vxml, Server rsp_code(404),
fd(0)
*Jan  3 12:08:01.677: HTTPC SOCK_FAIL() - msg(62CA5FB4)URL:http://serverX/vxml/test.vxml
*Jan  3 12:08:01.677: httpc_free: NULL pointer argument
*Jan  3 12:08:01.677: HTTPC URL:http://serverX/vxml/test.vxml, MSG_XMIT_ERROR, fd(-1)
Jan   3 23:44:06 PDT: HTTPC
URL:http://serverY.com:9000/ivr/sid-351/dropoffDeposit?pri=0&disconnect=1, TIMEOUT(60000
msec), fd(-1)
Jan   3 23:44:07 PDT: HTTPC msg timeout waiting for rsp - fd(21)
Jan   3 23:44:07 PDT: httpc_free: NULL pointer argument

Jan   4 02:45:07 PDT: HTTPC msg timeout waiting for rsp - fd(0)
Jan   4 02:45:07 PDT: HTTPC URL:http://rtsp-ws/dropoffAppend.php?append=&disconnect=1,
TIMEOUT(10000 msec), fd(-1)

Jan   4 02:46:07 PDT: httpc_msg_read: URL(http://1.7.100.1/vxml/root.vxml) - msg length
not available.Failed to parse message body.
Jan   4 02:46:07 PDT: httpc_msg_read: ERROR - DECODE
Jan   4 02:46:08 PDT: HTTPC bad message read - fd(6), comp(632A93B4),
msg(63280794)URL:http://1.7.100.1/vxml/test/root.vxml, len(1611)
Jan   4 02:46:08 PDT: First 400 bytes read from socket:

6241D9C0:                               3C3F78                                <?x
6241D9D0: 6D6C2076 65727369 6F6E3D22 312E3022  m1 version="1.0"
6241D9E0: 3F3E0A3C 76786D6C 20766572 73696F6E  ?>.<vxml version
6241D9F0: 3D22312E 30223E0A 0A20203C 70726F70  ="1.0">.. <prop
6241DA00: 65727479 206E616D 653D2263 61636869  erty name="cachi
6241DA10: 6E672220 76616C75 653D2273 61666522  ng" value="safe"
6241DA20: 2F3E0A20 203C7072 6F706572 7479206E  />. <property n
6241DA30: 616D653D 2274696D 656F7574 22207661  ame="timeout" va
6241DA40: 6C75653D 22333073 222F3E0A 20203C70  lue="30s"/>. <p
6241DA50: 726F7065 72747920 6E616D65 3D226665  roperty name="fe

```

```

6241DA60: 74636874 696D656F 75742220 76616C75 tchtimeout" valu
6241DA70: 653D2236 3073222F 3E0A2020 3C70726F e="60s"/>. <pro
6241DA80: 70657274 79206E61 6D653D22 696E7075 perty name="inpu
6241DA90: 746D6F64 65732220 76616C75 653D2264 tmodes" value="d
6241DAA0: 746D6622 2F3E0A0A 20203C76 6172206E tmf"/>.. <var n
6241DAB0: 616D653D 22736964 22206578 70723D22 ame="sid" expr="
6241DAC0: 27313131 31312722 2F3E0A20 203C7661 '11111'"/>. <va
6241DAD0: 72206E61 6D653D22 6C632220 65787072 r name="lc" expr
6241DAE0: 3D222765 6E2D7573 27222F3E 0A20203C ="'en-us'"/>. <
6241DAF0: 76617220 6E616D65 3D226861 6E646C65 var name="handle
6241DB00: 22206578 70723D22 74727565 222F3E0A " expr="true"/>.
6241DB10: 0A20203C 63617463 68206576 656E743D . <catch event=
6241DB20: 2274656C 6570686F 6E652E64 6973636F "telephone.disco
6241DB30: 6E6E6563 74222063 6F6E643D 2268616E nnect" cond="han
6241DB40: 646C6522 3E0A2020 20203C61 73736967 dle">. <assign
6241DB50: 6E206E61 6D653D22 68616E64 6C65 n name="handle
Jan 4 02:47:08 PDT: httpc_free: NULL pointer argument
Jan 4 02:47:09 PDT: HTTPC URL:http://1.7.100.1/vxml/test/root.vxml, MSG_DECODE_ERROR,
fd(6)
Jan 4 03:47:09 PDT: WARNING:httpc_msg_retry:
msg(6325CDD4):http://vvv.com/vxml/prompts/5.au

```

Related Commands

Command	Description
debug condition application voice	Displays debugging messages for only the specified VoiceXML application.
debug voip ivr	Displays debugging messages for VoIP IVR interactions.
debug vxml	Displays debugging messages for VoIP VoiceXML interactions.

debug http client cookie

To display debugging traces for cookie-related processes, including sending, receiving, validating, storing, and expiring a cookie, use the **debug http client cookie** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug http client cookie

no debug http client cookie

Syntax Description This command has no arguments or keywords.

Command Modes Privileged EXEC

Command History	Release	Modification
	12.3(8)T	This command was introduced.

Examples The following sample output from the **debug http client cookie** command shows that a cookie is being received and stored:

```
Router# debug http client cookie

*May 6 23:23:41.995: //38//HTTPPC:/httpc_decode_msgheader: received
cookie:TestCookieX=username; path=/; domain=.cisco.com
URL:http://rtsp-ws.cisco.com/cookie.php
*May 6 23:23:41.995: //38//HTTPPC:/httpc_decode_msgheader: received
cookie:TestCookieY=password; expires=Thu, 06-May-04 22:30:47 GMT; path=/;
domain=.cisco.com URL:http://rtsp-ws.cisco.com/cookie.php
*May 6 23:23:41.995: //38//HTTPPC:/httpc_cookie_store: validating
cookie:TestCookieX=username; path=/; domain=.cisco.com
*May 6 23:23:41.995: //38//HTTPPC:/httpc_cookie_store: store cookie:TestCookieX=username
path=/ domain=.cisco.com
*May 6 23:23:41.995: //38//HTTPPC:/httpc_cookie_store:
rtsp-7#validating cookie:TestCookieY=password; expires=Thu, 06-May-04 22:30:47 GMT;
path=/; domain=.cisco.com
*May 6 23:23:41.995: //38//HTTPPC:/httpc_cookie_store: store cookie:TestCookieY=password
path=/ domain=.cisco.com
*May 6 23:23:41.995: //38//HTTPPC:/httpc_process_response: TestCookieY=password path=/
domain=.cisco.com
TestCookieX=username path=/ domain=.cisco.com
```

Related Commands	Command	Description
	http client cache memory	Configures the memory limits for the HTTP client cache.
	http client cache refresh	Configures the refresh time for the HTTP client cache.
	http client cookie	Enables the HTTP client to send and receive cookies.
	show http client cookie	Displays cookies that are being stored by the HTTP client.

debug ima

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

debug ima

no debug ima

Syntax Description This command has no arguments or keywords.

Defaults Debugging for IMA groups is not enabled.

Command Modes Privileged EXEC

Command History

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

Examples

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

```
Router# debug ima

IMA network interface debugging is on
Router# config terminal
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)# interface atm1/0
Router(config-if)# ima-group 1
Router(config-if)#
01:35:08:IMA shutdown atm layer of link ATM1/0
01:35:08:ima_clear_atm_layer_if ATM1/0
01:35:08:IMA link ATM1/0 removed in firmware
01:35:08:ima_release_channel:ATM1/0 released channel 0.
01:35:08:Bring up ATM1/4 that had been waiting for a free channel.
01:35:08:IMA:no shut the ATM interface.
01:35:08:IMA allocate_channel:ATM1/4 using channel 0.
01:35:08:IMA config_restart ATM1/4
01:35:08:IMA adding link 0 to Group ATM1/IMA1ATM1/0 is down waiting for IMA group 1 to be
activated
01:35:08:Link 0 was added to Group ATM1/IMA1
01:35:08:ATM1/0 is down waiting for IMA group 1 to be created.
01:35:08:IMA send AIS on link ATM1/0
01:35:08:IMA Link up/down Alarm:port 0, new status 0x10, old_status 0x1.
01:35:10:%LINK-3-UPDOWN:Interface ATM1/4, changed state to up
01:35:10:%LINK-3-UPDOWN:Interface ATM1/0, changed state to down
```

```

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

```

Related Commands

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

debug ip access-list turboacl

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

debug ip access-list turboacl

no debug ip access-list turboacl

Syntax Description This command has no arguments or keywords.

Defaults No default behaviors or values

Command Modes Privileged EXEC

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

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

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

```
Router# debug ip access-list turboacl

*Aug 20 00:41:17.843 UTC:Miss at index 73, 19
*Aug 20 00:41:17.843 UTC:Adding dynamic entry, update = 1
*Aug 20 00:41:17.843 UTC:Miss at index 21, 39
*Aug 20 00:41:17.847 UTC:Adding dynamic entry, update = 1
*Aug 20 00:41:17.847 UTC:Miss at index 116, 42
*Aug 20 00:41:17.851 UTC:Adding dynamic entry, update = 1
*Aug 20 00:41:17.851 UTC:Miss at index 119, 28
*Aug 20 00:41:17.851 UTC:Adding dynamic entry, update = 1
*Aug 20 00:41:17.855 UTC:Miss at index 116, 42
*Aug 20 00:41:17.855 UTC:Adding dynamic entry, update = 1
*Aug 20 00:41:17.855 UTC:Miss at index 92, 20
*Aug 20 00:41:17.855 UTC:Adding dynamic entry, update = 1
*Aug 20 00:41:17.855 UTC:Miss at index 119, 28
*Aug 20 00:41:17.855 UTC:Adding dynamic entry, update = 1
*Aug 20 00:41:17.855 UTC:Miss at index 56, 29
*Aug 20 00:41:17.859 UTC:Adding dynamic entry, update = 1
```

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

Table 97 describes the significant fields shown in the display.

Table 97 *debug ip access-list turboacl Field Descriptions*

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

debug ip admission eapoudp

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

debug ip admission eapoudp

no debug ip admission eapoudp

Syntax Description This command has no arguments or keywords.

Defaults Debugging is not enabled.

Command Modes Privileged EXEC

Command History	Release	Modification
	12.3(8)T	This command was introduced.

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

```
Router# debug ip admission eapoudp

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

The fields in the display are self-explanatory.

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

debug ip auth-proxy

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

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

```
no debug ip auth-proxy
```

Syntax Description

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

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(5)T	This command was introduced.
12.3(1)	The detailed keyword was added.

Usage Guidelines

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



Note

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

Examples

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

```
00:37:50:AUTH-PROXY:proto_flag=5, dstport_index=1
00:37:50: SYN SEQ 245972 LEN 0
00:37:50:dst_addr 192.168.127.2 src_addr 192.168.27.1 dst_port 21 src_port 4347
00:37:50:AUTH-PROXY:auth_proxy_half_open_count++ 1
00:37:50:AUTH-PROXY:proto_flag=5, dstport_index=1
00:37:50: ACK 1820245643 SEQ 245973 LEN 0
00:37:50:dst_addr 192.168.127.2 src_addr 192.168.27.1 dst_port 21 src_port 4347
00:37:50:clientport 4347 state 0
00:37:50:AUTH-PROXY:incremented proxy_proc_count=1
00:37:50:AUTH-PROXY:proto_flag=5, dstport_index=1
00:37:50: ACK 1820245674 SEQ 245973 LEN 0
00:37:50:dst_addr 192.168.127.2 src_addr 192.168.27.1 dst_port 21 src_port 4347
00:37:50:clientport 4347 state 0
```

```

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

```

Related Commands

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

debug ip auth-proxy ezvpn

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

debug ip auth-proxy ezvpn

no debug ip auth-proxy ezvpn

Syntax Description This command has no arguments or keywords.

Defaults Debugging is not turned on.

Command Modes Privileged EXEC

Command History	Release	Modification
	12.3(14)T	This command was introduced.

Usage Guidelines



Caution

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

Examples

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

```
Router# debug ip auth-proxy ezvpn

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

The output in the display is self-explanatory.

Related Commands

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

debug ip bgp

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

debug ip bgp [*A.B.C.D.* | **dampening** | **events** | **in** | **keepalives** | **out** | **updates** | **vpn4** | **mpls**]

no debug ip bgp [*A.B.C.D.* | **dampening** | **events** | **in** | **keepalives** | **out** | **updates** | **vpn4** | **mpls**]

Syntax Description

<i>A.B.C.D.</i>	(Optional) Displays the BGP neighbor IP address.
dampening	(Optional) Displays BGP dampening.
events	(Optional) Displays BGP events.
in	(Optional) Displays BGP inbound information.
keepalives	(Optional) Displays BGP keepalives.
out	(Optional) Displays BGP outbound information.
updates	(Optional) Displays BGP updates.
vpn4	(Optional) Displays Virtual Private Network version 4 (VPNv4) network layer reachability information (NLRI).
mpls	(Optional) Displays the Multiprotocol Label Switching (MPLS) information.

Defaults

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(5)T	This command was introduced.
12.0(21)ST	This command was integrated into Cisco IOS Release 12.0(21)ST. The mpls keyword was added.
12.0(22)S	This command was integrated into Cisco IOS Release 12.0(22)S.
12.2(13)T	The mpls keyword was added.

Examples

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

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

```
03:47:14:vpn:bgp_vpnv4_bnetinit:100:2:58.0.0.0/8
03:47:14:vpn:bnetable add:100:2:58.0.0.0 / 8
03:47:14:vpn:bestpath_hook route_tag_change for vpn2:58.0.0.0/255.0.0.0(ok)
03:47:14:vpn:bgp_vpnv4_bnetinit:100:2:57.0.0.0/8
03:47:14:vpn:bnetable add:100:2:57.0.0.0 / 8
03:47:14:vpn:bestpath_hook route_tag_change for vpn2:57.0.0.0/255.0.0.0(ok)
03:47:14:vpn:bgp_vpnv4_bnetinit:100:2:14.0.0.0/8
03:47:14:vpn:bnetable add:100:2:14.0.0.0 / 8
03:47:14:vpn:bestpath_hook route_tag_chacle ip bgp *nge for vpn2:14.0.0.0/255.0.0.0(ok)
```

debug ip bgp groups

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

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

no debug ip bgp groups

Syntax Description

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

Defaults

No default behavior or values

Command Modes

Privileged EXEC

Command History

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

Usage Guidelines

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



Note

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

When a change to outbound policy occurs, the router automatically recalculates update-group memberships and applies the changes by triggering an outbound soft reset after a 3-minute timer expires. This behavior is designed to provide the network operator with time to change the configuration if a mistake is made. You can manually enable an outbound soft reset before the timer expires by entering the **clear ip bgp ip-address soft out** command.

Examples

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

```
Router# debug ip bgp groups

5w4d: BGP-DYN(0): Comparing neighbor 10.4.9.8 flags 0x0 cap 0x0 and updgrp 1 f10
5w4d: BGP-DYN(0): Created update-group(0) flags 0x0 cap 0x0 from neighbor 10.4.0
5w4d: BGP-DYN(0): Adding neighbor 10.4.9.8 flags 0x0 cap 0x0, to update-group 0
5w4d: %BGP-5-ADJCHANGE: neighbor 10.4.9.8 Up
```

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

```
Router# debug ip bgp groups

5w4d: %BGP-5-ADJCHANGE: neighbor 10.4.9.5 Down User reset
5w4d: BGP-DYN(0): Comparing neighbor 10.4.9.5 flags 0x0 cap 0x0 and updgrp 2 f10
5w4d: BGP-DYN(0): Update-group 2 flags 0x0 cap 0x0 policies same as 10.4.9.5 f10
5w4d: %BGP-5-ADJCHANGE: neighbor 10.4.9.8 Down User reset
5w4d: BGP-DYN(0): Comparing neighbor 10.4.9.8 flags 0x0 cap 0x0 and updgrp 2 f10
5w4d: BGP-DYN(0): Update-group 2 flags 0x0 cap 0x0 policies same as 10.4.9.8 f10
5w4d: %BGP-5-ADJCHANGE: neighbor 10.4.9.21 Down User reset
5w4d: BGP-DYN(0): Comparing neighbor 10.4.9.21 flags 0x0 cap 0x0 and updgrp 1 f0
5w4d: BGP-DYN(0): Update-group 1 flags 0x0 cap 0x0 policies same as 10.4.9.21 f0
5w4d: %BGP-5-ADJCHANGE: neighbor 10.4.9.5 Up
5w4d: %BGP-5-ADJCHANGE: neighbor 10.4.9.21 Up
5w4d: %BGP-5-ADJCHANGE: neighbor 10.4.9.8 Up
```

Table 98 describes the significant fields shown in the display.

Table 98 *debug ip bgp groups Field Descriptions*

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

Related Commands

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

debug ip casa affinities

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

debug ip casa affinities

no debug ip casa affinities

Syntax Description This command has no arguments or keywords.

Defaults Debugging for affinities is not enabled.

Command Modes Privileged EXEC

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

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

```
Router# debug ip casa affinities

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

[Table 99](#) describes the significant fields shown in the display.

Table 99 *debug ip casa affinities Field Descriptions*

Field	Description
Adding fixed affinity	Adding a fixed affinity to affinity table.
Updating fixed affinity	Modifying a fixed affinity table with information from the services manager.
flags	Bit field indicating actions to be taken on this affinity.
fwd addr	Address to which packets will be directed.
interest	Services manager that is interested in packets for this affinity.

Table 99 *debug ip casa affinities Field Descriptions (continued)*

Field	Description
int ip:port	Services manager port to which interest packets are sent.
sequence delta	Used to adjust TCP sequence numbers for this affinity.

debug ip casa packets

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

debug ip casa packets

no debug ip casa packets

Syntax Description This command has no arguments or keywords.

Defaults Debugging for packets is not enabled.

Command Modes Privileged EXEC

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

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

```
Router# debug ip casa packets

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

Table 100 describes the significant fields shown in the display.

Table 100 *debug ip casa packets Field Descriptions*

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

debug ip casa wildcards

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

debug ip casa wildcards

no debug ip casa wildcards

Syntax Description This command has no arguments or keywords.

Defaults Debugging for wildcards is not enabled.

Command Modes Privileged EXEC

Command History

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

Examples

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

```
Router# debug ip casa wildcards

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

[Table 101](#) describes the significant fields shown in the display.

Table 101 *debug ip casa wildcards Field Descriptions*

Field	Description
src mask	Source of connection.
dest mask	Destination of connection.
no frag, not advertising	Not accepting IP fragments.
flags	Bit field indicating actions to be taken on this affinity.
fwd addr	Address to which packets matching the affinity will be directed.

Table 101 *debug ip casa wildcards Field Descriptions (continued)*

Field	Description
interest	Services manager that is interested in packets for this affinity.
int ip: port	Services manager port to which interest packets are sent.
sequence delta	Used to adjust sequence numbers for this affinity.

debug ip cef

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

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

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

Specific to IPC Records

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

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

Syntax Description	
drops	Records dropped packets.
rpf	(Optional) Records the result of the Reverse Path Forwarding (RPF) check for packets.
<i>access-list</i>	(Optional) Limits debugging collection to packets that match the list.
receive	Records packets that are ultimately destined to the router, as well as packets destined to a tunnel endpoint on the router. If the decapsulated tunnel is IP, it is CEF switched; otherwise packets are process switched.
events	Records general CEF events.
interface	Records IP CEF interface events.
dialer	Records IP CEF interface events for dialer interfaces.
ipc	Records information related to Interprocess communications (IPC) in CEF. Possible types of events include the following: <ul style="list-style-type: none"> • Transmission status of IPC messages • Status of buffer space for IPC messages • IPC messages received out of sequence • Status of resequenced messages • Throttle requests sent from a line card to the Route Processor
interface-ipc	Records IPC updates related to interfaces. Possible reporting includes an interface coming up or going down, and updates to fibhwidb, fibidb, and so on.
prefix-ipc	Records updates related to IP prefix information. Possible updates include the following: <ul style="list-style-type: none"> • Debugging of IP routing updates in a line card • Reloading of a line card with a new table • Updates related to exceeding the maximum number of routes • Control messages related to Forwarding Information Base (FIB) table prefixes

Defaults Disabled

Command Modes Privileged EXEC

Command History	Release	Modification
	11.2 GS	This command was introduced.
	11.1 CC	Multiple platform support was added.
	12.0(5)T	The rpf keyword was added.
	12.2(4)T	The dialer keyword was added.

Usage Guidelines This command gathers additional information for the handling of CEF interface, IPC, or packet events.



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

Examples

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

```
Router# debug ip cef drops rpf
```

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

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

```
Router# debug ip cef receive
```

```
IP CEF received packets debugging is on
00:47:52:CEF-receive:Receive packet for 10.1.104.13
```

[Table 102](#) describes the significant fields shown in the display.

Table 102 *debug ip cef receive Field Descriptions*

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

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

```
Router# debug ip cef dialer

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

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

```
Router# debug ip cef dialer

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

Table 103 describes the significant fields shown in the display.

Table 103 *debug ip cef dialer Field Descriptions*

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

debug ip cef accounting non-recursive

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

```
debug ip cef accounting non-recursive
```

```
no debug ip cef accounting non-recursive
```

Syntax Description This command has no arguments or keywords.

Defaults Disabled

Command Modes Privileged EXEC

Command History	Release	Modification
	11.1 CC	This command was introduced.

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

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

```
Router# debug ip cef accounting non-recursive

03:50:19:CEF-Acct:tmstats_binary:Beginning generation of tmstats
ephemeral file (mode binary)
03:50:19:CEF-Acct:snapshotting loadinfo 0x63FF2000
03:50:19:CEF-Acct:snapshotting loadinfo 0x63FF1EA0
03:50:19:CEF-Acct:snapshotting loadinfo 0x63FF17C0
03:50:19:CEF-Acct:snapshotting loadinfo 0x63FF1D40
03:50:19:CEF-Acct:snapshotting loadinfo 0x63FF1A80
03:50:19:CEF-Acct:snapshotting loadinfo 0x63FF0740
03:50:19:CEF-Acct:snapshotting loadinfo 0x63FF08A0
03:50:19:CEF-Acct:snapshotting loadinfo 0x63FF0B60
03:50:19:CEF-Acct:snapshotting loadinfo 0x63FF0CC0
03:50:19:CEF-Acct:snapshotting loadinfo 0x63FF0F80
03:50:19:CEF-Acct:snapshotting loadinfo 0x63FF10E0
03:50:19:CEF-Acct:snapshotting loadinfo 0x63FF1240
03:50:19:CEF-Acct:snapshotting loadinfo 0x63FF13A0
03:50:19:CEF-Acct:snapshotting loadinfo 0x63FF1500
03:50:19:CEF-Acct:snapshotting loadinfo 0x63FF1920
03:50:19:CEF-Acct:snapshotting loadinfo 0x63FF0E20
03:50:19:CEF-Acct:snapshotting loadinfo 0x63FF1660
03:50:19:CEF-Acct:snapshotting loadinfo 0x63FF05E0
03:50:19:CEF-Acct:snapshotting loadinfo 0x63FF0A00
03:50:19:CEF-Acct:snapshotting loadinfo 0x63FF1BE0
```

```

03:50:19:CEF-Acct:snapshoting loadinfo 0x63FF0480
03:50:19:CEF-Acct:tmstats_binary:aggregation complete, duration 0 seconds
03:50:21:CEF-Acct:tmstats_binary:writing 45 bytes
03:50:24:CEF-Acct:tmstats_binary:writing 45 bytes
03:50:24:CEF-Acct:tmstats_binary:writing 45 bytes
03:50:27:CEF-Acct:tmstats_binary:writing 45 bytes
03:50:29:CEF-Acct:tmstats_binary:writing 45 bytes
03:50:32:CEF-Acct:tmstats_binary:writing 45 bytes
03:50:35:CEF-Acct:tmstats_binary:writing 45 bytes
03:50:38:CEF-Acct:tmstats_binary:writing 45 bytes
03:50:41:CEF-Acct:tmstats_binary:writing 45 bytes
03:50:45:CEF-Acct:tmstats_binary:writing 45 bytes
03:50:48:CEF-Acct:tmstats_binary:writing 45 bytes
03:50:49:CEF-Acct:tmstats_binary:writing 45 bytes
03:50:52:CEF-Acct:tmstats_binary:writing 45 bytes
03:50:55:CEF-Acct:tmstats_binary:writing 45 bytes
03:50:57:CEF-Acct:tmstats_binary:writing 45 bytes
03:50:57:CEF-Acct:tmstats_binary:writing 45 bytes
03:50:57:CEF-Acct:tmstats_binary:writing 45 bytes
03:50:57:CEF-Acct:tmstats_binary:writing 45 bytes
03:50:57:CEF-Acct:tmstats_binary:writing 45 bytes
03:50:57:CEF-Acct:tmstats_binary:tmstats file written, status 0

```

Table 104 describes the significant fields shown in the display.

Table 104 debug ip cef accounting non-recursive Field Descriptions

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

debug ip cef fragmentation

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

```
debug ip cef fragmentation
```

```
no debug ip cef fragmentation
```

Syntax Description This command has no arguments or keywords.

Defaults Disabled

Command Modes Privileged EXEC

Command History	Release	Modification
	12.0(14)S	This command was introduced.
	12.2(2)T	This command was integrated into Cisco IOS Release 12.2(2)T.

Usage Guidelines This command is used to troubleshoot fragmentation problems when CEF switching is enabled.

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

```
Router# debug ip cef fragmentation

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

Table 105 describes the significant fields shown in the display.

Table 105 *debug ip cef fragmentation Field Descriptions*

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

debug ip cef hash

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

debug ip cef hash

no debug ip cef hash

Syntax Description This command has no arguments or keywords.

Defaults Disabled

Command Modes Privileged EXEC

Command History	Release	Modification
	12.0(12)S	This command was introduced.
	12.1(5)T	This command was integrated into Cisco IOS Release 12.1(5)T.

Usage Guidelines Use this command when changing the load sharing algorithm to view the hash table details.

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

```
Router# debug ip cef hash

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

The following lines show IP CEF load algorithm universal information:

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

Table 106 describes the significant fields shown in the display.

Table 106 *debug ip cef hash Field Descriptions*

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

debug ip cef rhash

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

debug ip cef rhash

no debug ip cef rhash

Syntax Description This command has no arguments or keywords.

Defaults Disabled

Command Modes Privileged EXEC

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

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

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

```
Router# debug ip cef rhash

00:27:15:CEF:rrhash/check:found 9.1.104.7 on down idb [ok to delete]
00:27:15:CEF:rrhash/check:found 9.1.104.0 on down idb [ok to delete]
00:27:15:CEF:rrhash/check:found 9.1.104.255 on down idb [ok to delete]
00:27:15:CEF:rrhash/check:found 9.1.104.7 on down idb [ok to delete]
00:27:15:CEF:rrhash/check:found 9.1.104.7 on down idb [ok to delete]
00:27:15:CEF:rrhash/check:found 9.1.104.0 on down idb [ok to delete]
00:27:15:CEF:rrhash/check:found 9.1.104.255 on down idb [ok to delete]
00:27:15:CEF:rrhash/check:found 9.1.104.7 on down idb [ok to delete]
```

[Table 107](#) describes the significant fields shown in the display.

Table 107 debug ip cef rhash Field Descriptions

Field	Description
rrhash/check	Verify address is on the receive list.
found 9.1.104.7 on down idb [ok to delete]	Found a valid address on the receive list for a shutdown interface which is okay to delete.

debug ip cef subblock

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

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

```
no debug ip cef subblock
```

Syntax Description

id	(Optional) Subblock types.
all	(Optional) All subblock types.
hw <i>hw-id</i>	(Optional) Hardware subblock and identifier.
sw <i>sw-id</i>	(Optional) Software subblock and identifier.
xdr	(Optional) XDR message types.
control	(Optional) All XDR message types.
event	(Optional) Event XDR messages only.
none	(Optional) No XDR messages.
statistic	(Optional) Statistic XDR messages.

Defaults

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0 S	This command was introduced.
12.2(2)T	This command was integrated into Cisco IOS Release 12.2(2)T.

Usage Guidelines

This command is used to record CEF subblock messages and events.

Examples

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

```
Router# debug ip cef subblock

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

Table 108 describes the significant fields shown in the display.

Table 108 *debug ip cef subblock Field Descriptions*

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

debug ip cef table

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

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

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

Syntax Description

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

Defaults

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
11.2 GS	This command was introduced.
11.1 CC	Multiple platform support was added.
12.0(15)S	The consistency-checkers keyword was added.
12.2(2)T	This command was integrated into Cisco IOS Release 12.2(2)T.

Usage Guidelines

This command is used to record CEF table events related to the forwarding information base (FIB) table. Possible types of events include the following:

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

Examples

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

```
Router# debug ip cef table

01:25:46:CEF-Table:Event up, 1.1.1.1/32 (rdbs:1, flags:1000000)
01:25:46:CEF-IP:Checking dependencies of 0.0.0.0/0
01:25:47:CEF-Table:attempting to resolve 1.1.1.1/32
01:25:47:CEF-IP:resolved 1.1.1.1/32 via 9.1.104.1 to 9.1.104.1 Ethernet2/0/0
01:26:02:CEF-Table:Event up, default, 0.0.0.0/0 (rdbs:1, flags:400001)
01:26:02:CEF-IP:Prefix exists - no-op change
```

Table 109 describes the significant fields shown in the display.

Table 109 *debug ip cef table Field Descriptions*

Field	Description
CEF-Table	Indicates a table event.
Event up, 1.1.1.1/32	IP prefix 1.1.1.1/32 is being added.
rdbs:1	Event is from routing descriptor block 1.
flags:1000000	Indicates the network descriptor block flags.
CEF-IP	Indicates a CEF IP event.
Checking dependencies of 0.0.0.0/0	Resolves the next hop dependencies for 0.0.0.0/0.
attempting to resolve 1.1.1.1/32	Resolves the next hop dependencies.
resolved 1.1.1.1/32 via 9.1.104.1 to 9.1.104.1 Ethernet2/0/0	Next hop to IP prefix 1.1.1.1/32 is set and is added to the table.
Event up, default, 0.0.0.0/0 Prefix exists - no-op change	Indicates no table change is necessary for 0.0.0.0/32.

debug ip ddns update

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

debug ip ddns update

no debug ip ddns update

Syntax Description This command has no arguments or keywords.

Command Modes Privileged EXEC

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

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

Sample Configuration for the Client to Update A RRs and the Server to Update PTR RRs

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

```
!DHCP Client Configuration
ip ddns update method testing
  ddns

interface Ethernet1
 ip dhcp client update dns
 ip ddns update testing
 ip address dhcp
end

!DHCP Server Configuration
ip dhcp pool test
 network 11.0.0.0 255.0.0.0
 update dns
```

```

!Debug Output Enabled
Router# debug ip ddns update

00:14:39: %DHCP-6-ADDRESS_ASSIGN: Interface Ethernet1 assigned DHCP address 10.0.0.4, mask
255.0.0.0, hostname canada_reserved
00:14:39: DYNDNSUPD: Adding DNS mapping for canada_reserved.hacks <=> 10.0.0.4
00:14:39: DYNDNSUPD: Sleeping for 3 seconds waiting for interface Ethernet1 configuration
to settle
00:14:42: DHCP: Server performed PTR update
00:14:42: DDNS: Enqueuing new DDNS update 'canada_reserved.hacks' <=> 10.0.0.4
00:14:42: DDNS: Zone name for 'canada_reserved.hacks' is 'hacks'
00:14:42: DDNS: Dynamic Update 1: (sending to server 10.19.192.32)
00:14:42: DDNS:   Zone = hacks
00:14:42: DDNS:   Prerequisite: canada_reserved.hacks not in use
00:14:42: DDNS:   Update: add canada_reserved.hacks IN A 10.0.0.4
00:14:42: DDNS: Dynamic DNS Update 1 (A) for host canada_reserved.hacks returned 0
(NOERROR)
00:14:42: DDNS: Update of 'canada_reserved.hacks' <=> 10.0.0.4 finished
00:14:42: DYNDNSUPD: Another update completed (total outstanding=0)

```

Sample Configuration for the Client to Update Both A and DNS RRs and the Server to Update Neither

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

```

!DHCP Client Configuration
ip dhcp-client update dns server none

ip ddns update method testing
  dns both

interface Ethernet1
  ip ddns update testing
  ip address dhcp
end

!DHCP Server Configuration
ip dhcp pool test
  network 10.0.0.0 255.0.0.0
  update dns

!Debug Output Enabled
Router# debug ip ddns update

00:15:33: %DHCP-6-ADDRESS_ASSIGN: Interface Ethernet1 assigned DHCP address 10.0.0.5, mask
255.0.0.0, hostname canada_reserved
00:15:33: DYNDNSUPD: Adding DNS mapping for canada_reserved.hacks <=> 10.0.0.5
00:15:33: DYNDNSUPD: Sleeping for 3 seconds waiting for interface Ethernet1 configuration
to settle
00:15:36: DDNS: Enqueuing new DDNS update 'canada_reserved.hacks' <=> 10.0.0.5
00:15:36: DDNS: Zone name for '11.0.0.11.in-addr.arpa.' is '11.in-addr.arpa'
00:15:36: DDNS: Dynamic Update 1: (sending to server 10.19.192.32)
00:15:36: DDNS:   Zone = 11.in-addr.arpa
00:15:36: DDNS:   Prerequisite: 5.0.0.11.in-addr.arpa. not in use
00:15:36: DDNS:   Update: add 5.0.0.11.in-addr.arpa. IN PTR canada_reserved.hacks
00:15:36: DDNS: Dynamic DNS Update 1 (PTR) for host canada_reserved.hacks returned 0
(NOERROR)
00:15:36: DDNS: Zone name for 'canada_reserved.hacks' is 'hacks'
00:15:36: DDNS: Dynamic Update 1: (sending to server 10.19.192.32)
00:15:36: DDNS:   Zone = hacks

```

```

00:15:36: DDNS: Prerequisite: canada_reserved.hacks not in use
00:15:36: DDNS: Update: add canada_reserved.hacks IN A 10.0.0.5
00:15:36: DDNS: Dynamic DNS Update 1 (A) for host canada_reserved.hacks returned 0
(NOERROR)
00:15:36: DDNS: Update of 'canada_reserved.hacks' <=> 10.0.0.5 finished
00:15:36: DYNDNSUPD: Another update completed (total outstanding=0)

```

Sample Configuration for the Client to Update A and DNS RRs and the Server to Update Neither

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

```

!DHCP Client Configuration
ip dhcp client update dns server none

ip ddns update method testing
  ddns both

interface Ethernet1
  ip dhcp client update dns server none
  ip ddns update testing
  ip address dhcp
end

!DHCP Server Configuration
ip dhcp pool test
  network 11.0.0.0 255.0.0.0
  update dns both override

!Debug Output Enabled on DHCP Client
Router# debug ip ddns update

00:16:30: %DHCP-6-ADDRESS_ASSIGN: Interface Ethernet1 assigned DHCP address 10.0.0.6, mask
255.0.0.0, hostname canada_reserved
00:16:30: DYNDNSUPD: Adding DNS mapping for canada_reserved.hacks <=> 10.0.0.6
00:16:30: DYNDNSUPD: Sleeping for 3 seconds waiting for interface Ethernet1 configuration
to settle
00:16:33: DHCP: Server performed both updates

```

Sample Configuration for the Client to Update A and DNS RRs and the Server to Update Neither

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

```

!DHCP Client Configuration
ip dhcp client update dns server non

ip ddns update method testing
  ddns both

interface Ethernet1
  ip dhcp client update dns server none
  ip ddns update testing host 172.19.192.32
  ip address dhcp
end

```

```

!DHCP Server Configuration
ip dhcp pool test
 network 11.0.0.0 255.0.0.0
 update dns

!Debug Output Enabled on DHCP Client
Router# debug ip ddns update

00:17:52: %DHCP-6-ADDRESS_ASSIGN: Interface Ethernet1 assigned DHCP address 10.0.0.7, mask
255.0.0.0, hostname canada_reserved
00:17:52: DYNDNSUPD: Adding DNS mapping for canada_reserved.hacks <=> 10.0.0.6
00:17:52: DYNDNSUPD: Sleeping for 3 seconds waiting for interface Ethernet1 configuration
to settle
00:17:55: DDNS: Enqueuing new DDNS update 'canada_reserved.hacks' <=> 10.0.0.7
00:17:55: DYNDNSUPD: Adding DNS mapping for canada_reserved.hacks <=> 10.0.0.7 server
10.19.192.32
00:17:55: DDNS: Enqueuing new DDNS update 'canada_reserved.hacks' <=> 10.0.0.7 server
10.19.192.32
00:17:55: DDNS: Zone name for '7.0.0.11.in-addr.arpa.' is '11.in-addr.arpa'
00:17:55: DDNS: Dynamic Update 1: (sending to server 10.19.192.32)
00:17:55: DDNS: Zone = 11.in-addr.arpa
00:17:55: DDNS: Prerequisite: 7.0.0.11.in-addr.arpa. not in use
00:17:55: DDNS: Update: add 7.0.0.11.in-addr.arpa. IN PTR canada_reserved.hacks
00:17:55: DDNS: Zone name for '7.0.0.11.in-addr.arpa.' is '11.in-addr.arpa'
00:17:55: DDNS: Using server 10.19.192.32
00:17:55: DDNS: Dynamic Update 1: (sending to server 10.19.192.32)
00:17:55: DDNS: Zone = 11.in-addr.arpa
00:17:55: DDNS: Prerequisite: 7.0.0.11.in-addr.arpa. not in use
00:17:55: DDNS: Update: add 7.0.0.11.in-addr.arpa. IN PTR canada_reserved.hacks
00:17:55: DDNS: Dynamic DNS Update 1 (PTR) for host canada_reserved.hacks returned 0
(NOERROR)
00:17:55: DDNS: Dynamic DNS Update 1 (PTR) for host canada_reserved.hacks returned 6
(YXDOMAIN)
00:17:55: DDNS: Dynamic Update 2: (sending to server 10.19.192.32)
00:17:55: DDNS: Zone = 11.in-addr.arpa
00:17:55: DDNS: Update: delete 7.0.0.11.in-addr.arpa. all PTR RRs
00:17:55: DDNS: Update: add 7.0.0.11.in-addr.arpa. IN PTR canada_reserved.hacks
00:17:55: DDNS: Dynamic DNS Update 2 (PTR) for host canada_reserved.hacks returned 0
(NOERROR)
00:17:55: DDNS: Zone name for 'canada_reserved.hacks' is 'hacks'
00:17:55: DDNS: Dynamic Update 1: (sending to server 10.19.192.32)
00:17:55: DDNS: Zone = hacks
00:17:55: DDNS: Prerequisite: canada_reserved.hacks not in use
00:17:55: DDNS: Update: add canada_reserved.hacks IN A 10.0.0.7
00:17:55: DDNS: Dynamic DNS Update 1 (A) for host canada_reserved.hacks returned 0
(NOERROR)
00:17:55: DDNS: Update of 'canada_reserved.hacks' <=> 10.0.0.7 finished
00:17:55: DYNDNSUPD: Another update completed (total outstanding=1)
00:17:55: DDNS: Zone name for 'canada_reserved.hacks' is 'hacks'
00:17:55: DDNS: Using server 10.19.192.32
00:17:55: DDNS: Dynamic Update 1: (sending to server 10.19.192.32)
00:17:55: DDNS: Zone = hacks
00:17:55: DDNS: Prerequisite: canada_reserved.hacks not in use
00:17:55: DDNS: Update: add canada_reserved.hacks IN A 10.0.0.7
00:17:55: DDNS: Dynamic DNS Update 1 (A) for host canada_reserved.hacks returned 6
(YXDOMAIN)
00:17:55: DDNS: Dynamic Update 2: (sending to server 10.19.192.32)
00:17:55: DDNS: Zone = hacks
00:17:55: DDNS: Update: delete canada_reserved.hacks all A RRs
00:17:55: DDNS: Update: add canada_reserved.hacks IN A 10.0.0.7
00:17:55: DDNS: Dynamic DNS Update 2 (A) for host canada_reserved.hacks returned 0
(NOERROR)
00:17:55: DDNS: Update of 'canada_reserved.hacks' <=> 10.0.0.7 finished
00:17:55: DYNDNSUPD: Another update completed (total outstanding=0)

```

Sample Configuration for Updating the Internal Host Table

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

```
!Cisco IOS Software Configuration
ip domain name hacks

ip ddns update method test
  internal

interface ethernet0/0
  ip ddns update test hostname test2
  ip addr dhcp

!Debug Output Enabled
Router# debug ip ddns update

*Jun 4 03:11:10.591: %DHCP-6-ADDRESS_ASSIGN: Interface Ethernet0/0 assigned DHCP address
10.0.0.5, mask 255.0.0.0, hostname test2
*Jun 4 03:11:10.591: DYNDNSUPD: Adding DNS mapping for test2.hacks <=> 10.0.0.5
*Jun 4 03:11:10.591: DYNDNSUPD: Adding internal mapping test2.hacks <=> 10.0.0.5
```

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

```
Router# show hosts

Default domain is hacks
Name/address lookup uses domain service
Name servers are 255.255.255.255

Codes: UN - unknown, EX - expired, OK - OK, ?? - revalidate
       temp - temporary, perm - permanent
       NA - Not Applicable None - Not defined

Host                Port Flags      Age Type   Address(es)
test2.hacks         None (perm, OK) 0   IP     10.0.0.5
```

Shutting down the interface removes the host table entry.

```
interface ethernet0/0
  shutdown

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

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

```
Router# show hosts

Default domain is hacks
Name/address lookup uses domain service
Name servers are 255.255.255.255

Codes: UN - unknown, EX - expired, OK - OK, ?? - revalidate
       temp - temporary, perm - permanent
       NA - Not Applicable None - Not defined

Host                Port Flags      Age Type   Address(es)
```

Sample Configuration of HTTP DDNS Updates

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

```
!DHCP Client Configuration
ip ddns update method dyndns
  http
    add http://test:test@<s>/nic/update?system=dyndns&hostname=<h>&myip=<a>
      interval max 28 0 0 0

interface ethernet1
  ip ddns update hostname test.dyndns.org
  ip ddns update dyndns host members.dyndns.org
  ip addr dhcp

!Debugging Enabled
Router# debug ip ddns update

00:04:35: %DHCP-6-ADDRESS_ASSIGN: Interface Ethernet1 assigned DHCP address 10.32.254.187,
mask 255.255.255.240, hostname test.dyndns.org
00:04:35: DYNDNSUPD: Adding DNS mapping for test.dyndns.org <=> 10.32.254.187 server
63.208.196.94
00:04:35: DYNDNSUPD: Sleeping for 3 seconds waiting for interface Ethernet1 configuration
to settle
00:04:38: HTTPDNS: Update add called for test.dyndns.org <=> 10.32.254.187
00:04:38: HTTPDNS: Update called for test.dyndns.org <=> 10.32.254.187
00:04:38: HTTPDNS: init
00:04:38: HTTPDNSUPD: Session ID = 0x7
00:04:38: HTTPDNSUPD: URL =
'http://test:test@63.208.196.94/nic/update?system=dyndns&hostname=test.dyndns.org&myip=10.
32.254.187'
00:04:38: HTTPDNSUPD: Sending request
00:04:40: HTTPDNSUPD: Response for update test.dyndns.org <=> 10.32.254.187
00:04:40: HTTPDNSUPD: DATA START
good 10.32.254.187
00:04:40: HTTPDNSUPD: DATA END, Status is Response data received, successfully
00:04:40: HTTPDNSUPD: Call returned SUCCESS for update test.dyndns.org <=> 10.32.254.187
00:04:40: HTTPDNSUPD: Freeing response
00:04:40: DYNDNSUPD: Another update completed (outstanding=0, total=0)
00:04:40: HTTPDNSUPD: Clearing all session 7 info

!28 days later, the automatic update happens.

00:05:39: DYNDNSUPD: Adding DNS mapping for test.dyndns.org <=> 10.32.254.187 server
63.208.196.94
00:05:39: HTTPDNS: Update add called for test.dyndns.org <=> 10.32.254.187
00:05:39: HTTPDNS: Update called for test.dyndns.org <=> 10.32.254.187
00:05:39: HTTPDNS: init
00:05:39: HTTPDNSUPD: Session ID = 0x8
00:05:39: HTTPDNSUPD: URL =
'http://test:test@63.208.196.94/nic/update?system=dyndns&hostname=test.dyndns.org&myip=10.
32.254.187'
00:05:39: HTTPDNSUPD: Sending request
00:05:39: HTTPDNSUPD: Response for update test.dyndns.org <=> 10.32.254.187
00:05:39: HTTPDNSUPD: DATA START
nochg 10.32.254.187
00:05:39: HTTPDNSUPD: DATA END, Status is Response data received, successfully
00:05:39: HTTPDNSUPD: Call returned SUCCESS for update test.dyndns.org <=> 10.32.254.187
```

```
00:05:39: HTTPDNSUPD: Freeing response
00:05:39: DYNDNSUPD: Another update completed (outstanding=0, total=0)
00:05:39: HTTPDNSUPD: Clearing all session 8 info
```

Table 110 describes the significant fields shown in the output.

Table 110 debug ip ddns update Field Descriptions

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

Related Commands

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

debug ip dfp agent

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

debug ip dfp agent

no debug ip dfp agent

Syntax Description

This command has no arguments or keywords.

Defaults

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.1(8a)E	This command was introduced.
12.2(14)S	This command was integrated into Cisco IOS Release 12.2(14)S.
12.3(14)T	This command was integrated into Cisco OS Release 12.3(14)T.

Usage Guidelines



Caution

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

Examples

The following example configures a DFP agent debug session:

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

debug ip dhcp server

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

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

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

Syntax Description

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

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(1)T	This command was introduced.
12.2(13)ZH	The class keyword was added.
12.3(4)T	This command was integrated into Cisco IOS Release 12.3(4)T.
12.3(8)YA	The output was enhanced to show DDNS updates on a DHCP server.
12.3(11)T	The output was enhanced to show static mappings.

Examples

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

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

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

The following example shows database linkage information:

```
Router# debug ip dhcp server linkage

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

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

The following example shows when a DHCP class is removed:

```
Router# debug ip dhcp server class
```

```
DHCPD:deleting class CLASS1
```

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

```
Router# debug ip dhcp server class
```

```
DHCPD:Searching for a match to 'relay-information
0106000 400020202020800060009e80b8800' in class CLASS1
DHCPD:Searching for a match to 'relay-information 0106000400020202020800060009e80b8800' in
class CLASS1
DHCPD:Searching for a match to 'relay-information 0106000
```

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

```
Router# debug ip dhcp server class
```

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

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

```
Router# debug ip dhcp server class
```

```
DHCPD:Searching for a match to 'relay-information
0106000 400020202020800060009e80b8800' in class CLASS1
DHCPD:input pattern 'relay-information 010600040002020202 0800060009e80b8800' matches
class CLASS1
DHCPD:input matches class CLASS1
```

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

```
Loading abc/static_pool from 10.19.192.33 (via Ethernet0): !
[OK - 333 bytes]

*May 26 23:14:21.259: DHCPD: contacting agent tftp://10.19.192.33/abc/static_pool (attempt
0)
*May 26 23:14:21.467: DHCPD: agent tftp://10.19.192.33/abc/static_pool is responding.
*May 26 23:14:21.467: DHCPD: IFS is ready.
*May 26 23:14:21.467: DHCPD: reading bindings from
tftp://10.19.192.33/abc/static_pool.
*May 26 23:14:21.707: DHCPD: read 333 / 1024 bytes.
*May 26 23:14:21.707: DHCPD: parsing text line "*time* Apr 22 2002 11:31 AM"
*May 26 23:14:21.707: DHCPD: parsing text line ""
*May 26 23:14:21.707: DHCPD: parsing text line
"!IP address Type Hardware address Lease expiration"
*May 26 23:14:21.707: DHCPD: parsing text line
"10.9.9.1/24 id 0063.6973.636f.2d30.3036.302e.3437"
*May 26 23:14:21.707: DHCPD: creating binding for 10.9.9.1
*May 26 23:14:21.707: DHCPD: Adding binding to radix tree (10.9.9.1)
*May 26 23:14:21.707: DHCPD: Adding binding to hash tree
*May 26 23:14:21.707: DHCPD: parsing text line
"10.9.9.4 id 0063.7363.2d30.3036.302e.3762.2e39.3634.632d"
*May 26 23:14:21.711: DHCPD: creating binding for 10.9.9.4
```

```
*May 26 23:14:21.711: DHCPD: Adding binding to radix tree (10.9.9.4)
*May 26 23:14:21.711: DHCPD: Adding binding to hash tree
*May 26 23:14:21.711: DHCPD: parsing text line "Infinite"
*May 26 23:14:21.711: DHCPD: parsing text line ""
*May 26 23:14:21.711: DHCPD: parsing text line
"!IP address Interface-index Lease expiration VRF"
*May 26 23:14:21.711: DHCPD: parsing text line "**end*"
*May 26 23:14:21.711: DHCPD: read static bindings from
tftp://10.19.192.33/athenmoz/static_pool.
```

Related Commands

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

debug ip drp

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

debug ip drp

no debug ip drp

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

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

Examples

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

```
Router# debug ip drp
```

```
DRP: received v1 packet from 172.69.232.8, via Ethernet0
DRP: RTQUERY for 172.69.58.94 returned internal=0, external=0
```

[Table 111](#) describes the significant fields shown in the display.

Table 111 debug ip drp Field Descriptions

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