

debug ip dvmrp

To display information on Distance Vector Multiprotocol Routing Protocol (DVMRP) packets received and sent, use the **debug ip dvmrp** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip dvmrp [**detail** [*access-list*] [**in** | **out**]]

no debug ip dvmrp [**detail** [*access-list*] [**in** | **out**]]

Syntax Description	detail	(Optional) Enables a more detailed level of output and displays packet contents.
	<i>access-list</i>	(Optional) Causes the debug ip dvmrp command to restrict output to one access list.
	in	(Optional) Causes the debug ip dvmrp command to output packets received in DVMRP reports.
	out	(Optional) Causes the debug ip dvmrp command to output packets sent in DVMRP reports.

Command Modes Privileged EXEC

Usage Guidelines Use the **debug ip dvmrp detail** command with care. This command generates a substantial amount of output and can interrupt other activity on the router when it is invoked.

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

```
Router# debug ip dvmrp

DVMRP: Received Report on Ethernet0 from 172.19.244.10
DVMRP: Received Report on Ethernet0 from 172.19.244.11
DVMRP: Building Report for Ethernet0 224.0.0.4
DVMRP: Send Report on Ethernet0 to 224.0.0.4
DVMRP: Sending IGMP Reports for known groups on Ethernet0
DVMRP: Received Report on Ethernet0 from 172.19.244.10
DVMRP: Received Report on Tunnel0 from 192.168.199.254
DVMRP: Received Report on Tunnel0 from 192.168.199.254
DVMRP: Received Report on Tunnel0 from 192.168.199.254
DVMRP: Received Report on Tunnel0 from 192.168.199.254
DVMRP: Received Report on Tunnel0 from 192.168.199.254
DVMRP: Received Report on Tunnel0 from 192.168.199.254
DVMRP: Received Report on Tunnel0 from 192.168.199.254
DVMRP: Building Report for Tunnel0 224.0.0.4
DVMRP: Send Report on Tunnel0 to 192.168.199.254
DVMRP: Send Report on Tunnel0 to 192.168.199.254
DVMRP: Send Report on Tunnel0 to 192.168.199.254
DVMRP: Send Report on Tunnel0 to 192.168.199.254
DVMRP: Radix tree walk suspension
DVMRP: Send Report on Tunnel0 to 192.168.199.254
```

The following lines show that the router received DVMRP routing information and placed it in the mroute table:

```
DVMRP: Received Report on Ethernet0 from 172.19.244.10
DVMRP: Received Report on Ethernet0 from 172.19.244.11
```

The following lines show that the router is creating a report to send to another DVMRP router:

```
DVMRP: Building Report for Ethernet0 224.0.0.4
DVMRP: Send Report on Ethernet0 to 224.0.0.4
```

[Table 112](#) provides a list of internet multicast addresses supported for host IP implementations.

Table 112 Internet Multicast Addresses

Address	Description	RFC
224.0.0.0	Base address (reserved)	RFC 1112
224.0.0.1	All systems on this subnet	RFC 1112
224.0.0.2	All routers on this subnet	
224.0.0.3	Unassigned	
224.0.0.4	DVMRP routers	RFC 1075
224.0.0.5	OSPF/IGP all routers	RFC 1583

The following lines show that a protocol update report has been sent to all known multicast groups. Hosts use Internet Group Management Protocol (IGMP) reports to communicate with routers and to request to join a multicast group. In this case, the router is sending an IGMP report for every known group to the host, which is running mroute. The host then responds as though the router were a host on the LAN segment that wants to receive multicast packets for the group.

```
DVMRP: Sending IGMP Reports for known groups on Ethernet0
```

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

```
Router# debug ip dvmrp detail

DVMRP: Sending IGMP Reports for known groups on Ethernet0
DVMRP: Advertise group 224.2.224.2 on Ethernet0
DVMRP: Advertise group 224.2.193.34 on Ethernet0
DVMRP: Advertise group 224.2.231.6 on Ethernet0
DVMRP: Received Report on Tunnel0 from 192.168.199.254
DVMRP: Origin 150.166.53.0/24, metric 13, distance 0
DVMRP: Origin 150.166.54.0/24, metric 13, distance 0
DVMRP: Origin 150.166.55.0/24, metric 13, distance 0
DVMRP: Origin 150.166.56.0/24, metric 13, distance 0
DVMRP: Origin 150.166.92.0/24, metric 12, distance 0
DVMRP: Origin 150.166.100.0/24, metric 12, distance 0
DVMRP: Origin 150.166.101.0/24, metric 12, distance 0
DVMRP: Origin 150.166.142.0/24, metric 8, distance 0
DVMRP: Origin 150.166.200.0/24, metric 12, distance 0
DVMRP: Origin 150.166.237.0/24, metric 12, distance 0
DVMRP: Origin 150.203.5.0/24, metric 8, distance 0
```

The following lines show that this group is available to the DVMRP router. The mroute process on the host will forward the source and multicast information for this group through the DVMRP cloud to other members.

```
DVMRP: Advertise group 224.2.224.2 on Ethernet0
```

The following lines show the DVMRP route information:

```
DVMRP: Origin 150.166.53.0/24, metric 13, distance 0  
DVMRP: Origin 150.166.54.0/24, metric 13, distance 0
```

The metric is the number of hops the route has covered, and the distance is the administrative distance.

debug ip eigrp

To display information on Enhanced IGRP (EIGRP) protocol packets, use the **debug ip eigrp** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip eigrp

no debug ip eigrp

Syntax Description This command has no arguments or keywords.

Command Modes Privileged EXEC

Usage Guidelines This command helps you analyze the packets that are sent and received on an interface. Because the **debug ip eigrp** command generates a substantial amount of output, only use it when traffic on the network is light.

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

```
Router# debug ip eigrp

IP-EIGRP: Processing incoming UPDATE packet
IP-EIGRP: Ext 192.168.3.0 255.255.255.0 M 386560 - 256000 130560 SM 360960 - 256000 104960
IP-EIGRP: Ext 192.168.0.0 255.255.255.0 M 386560 - 256000 130560 SM 360960 - 256000 104960
IP-EIGRP: Ext 192.168.3.0 255.255.255.0 M 386560 - 256000 130560 SM 360960 - 256000 104960
IP-EIGRP: 172.69.43.0 255.255.255.0, - do advertise out Ethernet0/1
IP-EIGRP: Ext 172.69.43.0 255.255.255.0 metric 371200 - 256000 115200
IP-EIGRP: 192.135.246.0 255.255.255.0, - do advertise out Ethernet0/1
IP-EIGRP: Ext 192.135.246.0 255.255.255.0 metric 46310656 - 45714176 596480
IP-EIGRP: 172.69.40.0 255.255.255.0, - do advertise out Ethernet0/1
IP-EIGRP: Ext 172.69.40.0 255.255.255.0 metric 2272256 - 1657856 614400
IP-EIGRP: 192.135.245.0 255.255.255.0, - do advertise out Ethernet0/1
IP-EIGRP: Ext 192.135.245.0 255.255.255.0 metric 40622080 - 40000000 622080
IP-EIGRP: 192.135.244.0 255.255.255.0, - do advertise out Ethernet0/1
```

[Table 113](#) describes the significant fields shown in the display.

Table 113 *debug ip eigrp* Field Descriptions

Field	Description
IP-EIGRP:	Indicates that this is an IP EIGRP.
Ext	Indicates that the following address is an external destination rather than an internal destination, which would be labeled as Int.
M	Displays the computed metric, which includes the value in the SM field and the cost between this router and the neighbor. The first number is the composite metric. The next two numbers are the inverse bandwidth and the delay, respectively.
SM	Displays the metric as reported by the neighbor.

debug ip eigrp notifications

To display Enhanced Interior Gateway Routing Protocol (EIGRP) events and notifications in the console of the router, use the **debug ip eigrp notifications** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip eigrp notifications

no debug ip eigrp notifications

Syntax Description This command has no arguments or keywords.

Defaults No default behavior or values

Command Modes Privileged EXEC

Command History	Release	Modification
	12.2(15)T	This command was introduced.

Usage Guidelines The output from the **debug ip eigrp notifications** command displays EIGRP events and notifications.

Examples The following example output shows that the NSF-aware router has received the restart notification. The NSF-aware router will now wait for end of transmission (EOT) to be sent from the restarting neighbor (NSF-capable).

```
Router# debug ip eigrp notifications

*Oct  4 11:39:18.092:EIGRP:NSF:AS2. Rec RS update from 135.100.10.1,
00:00:00. Wait for EOT.
*Oct  4 11:39:18.092:%DUAL-5-NBRCHANGE:IP-EIGRP(0) 2:Neighbor
135.100.10.1 (POS3/0) is up:peer NSF restarted
```

Related Commands	Command	Description
	timers nsf-route-hold	Sets the route-hold timer for NSF-aware routers that run EIGRP.

debug ip error

To display IP errors, use the **debug ip error** command in privileged EXEC mode. To disable debugging errors, use the **no** form of this command.

debug ip error *access-list-number* [**detail**] [**dump**]

no debug ip error

Syntax Description	<i>access-list-number</i>	(Optional) The IP access list number that you can specify. If the datagram is not permitted by that access list, the related debugging output (or IP error) is suppressed. Standard, extended, and expanded access lists are supported. The range of standard and extended access lists is from 1 to 199. The range of expanded access lists is from 1300 to 2699.
	detail	(Optional) Displays detailed IP error debugging information.
	dump	(Hidden) Displays IP error debugging information along with raw packet data in hexadecimal and ASCII forms. This keyword can be enabled with individual access lists and also with the detail keyword.
	Note	The dump keyword is not fully supported and should be used only in collaboration with Cisco Technical Support. See the caution notes below, in the usage guidelines, for more specific information.

Defaults No default behavior or values

Command Modes Privileged EXEC

Usage Guidelines This command is used for IP error debugging. The output displays IP errors which are locally detected by this router.



Caution

Enabling this command will generate output only if IP errors occur. However, if the router starts to receive many packets that contain errors, substantial output may be generated and severely affect system performance. This command should be used with caution in production networks. It should only be enabled when traffic on the IP network is low, so other activity on the system is not adversely affected. Enabling the **detail** and **dump** keywords use the highest level of system resources of the available configuration options for this command, so a high level of caution should be applied when enabling either of these keywords.

**Caution**

The **dump** keyword is not fully supported and should be used only in collaboration with Cisco Technical Support. Because of the risk of using significant CPU utilization, the dump keyword is hidden from the user and cannot be seen using the “?” prompt. The length of the displayed packet information may exceed the actual packet length and include additional padding bytes that do not belong to the IP packet. Also note that the beginning of a packet may start at different locations in the dump output depending on the specific router, interface type, and packet header processing that may have occurred before the output is displayed.

Examples

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

```
Router# debug ip error

IP packet errors debugging is on

04:04:45:IP:s=10.8.8.1 (Ethernet0/1), d=10.1.1.1, len 28, dispose ip.hopcount
```

The IP error in the above output was caused when the router attempted to forward a packet with a time-to-live (TTL) value of 0. The “ip.hopcount” traffic counter is incremented when a packet is dropped because of an error. This error is also displayed in the output of the **show ip traffic** command by the “bad hop count” traffic counter.

[Table 114](#) describes the significant fields shown in the display.

Table 114 *debug ip error Field Descriptions*

Field	Description
IP:s=10.8.8.1 (Ethernet0/1)	The packet source IP address and interface.
d=10.1.1.1, len 28	The packet destination IP address and prefix length.
dispose ip.hopcount	This traffic counter increments when an IP packet is dropped because of an error.

The following is sample output from the **debug ip error** command enabled with the **detail** keyword:

```
Router# debug ip error detail

IP packet errors debugging is on (detailed)

1d08h:IP:s=10.0.19.100 (Ethernet0/1), d=10.1.1.1, len 28, dispose udp.noport
1d08h:   UDP src=41921, dst=33434

1d08h:IP:s=10.0.19.100 (Ethernet0/1), d=10.2.2.2, len 28, dispose ip.hopcount
1d08h:   UDP src=33691, dst=33434
```

The detailed output includes layer 4 information in addition to the standard output. The IP error in the above output was caused when the router received a UDP packet when no application was listening to the UDP port. The “udp.noport” traffic counter is incremented when the router drops a UDP packet because of this error. This error is also displayed in the output of the **show ip traffic** command by the “no port” traffic counter under “UDP statistics.”

Table 115 describes the significant fields shown in the display.

Table 115 *debug ip error detail Field Descriptions*

Field	Description
IP:s=10.0.19.100 (Ethernet0/1)	The IP packet source IP address and interface.
d=10.1.1.1, len 28	The IP packet destination and prefix length.
dispose udp.noport	The traffic counter that is incremented when a UDP packet is dropped because of this error.

The following is sample output from the **debug ip error** command enabled with the **detail** and **dump** keywords:

```
Router# debug ip error detail dump

IP packet errors debugging is on (detailed) (dump)

1d08h:IP:s=10.0.19.100 (Ethernet0/1), d=10.1.1.1, len 28, dispose udp.noport
1d08h:   UDP src=37936, dst=33434
03D72360:                0001 42AD4242                ..B-BB
03D72370:0002FCA5 DC390800 4500001C 30130000 ..|\9..E...0...
03D72380:01116159 0A001364 0A010101 9430829A ..aY...d.....0..
03D72390:0008C0AD                ..@-

1d08h:IP:s=10.0.19.100 (Ethernet0/1), d=10.2.2.2, len 28, dispose ip.hopcount
1d08h:   UDP src=41352, dst=33434
03C01600:                0001 42AD4242                ..B-BB
03C01610:0002FCA5 DC390800 4500001C 302A0000 ..|\9..E...0*...
03C01620:01116040 0A001364 0A020202 A188829A ..`@...d....!...
03C01630:0008B253                ..2S
```



Note

The **dump** keyword is not fully supported and should be used only in collaboration with Cisco Technical Support. See the caution in the usage guidelines section of this command reference page for more specific information.

The output from the **debug ip error** command, when the **dump** keyword is enabled, provides raw packet data in hexadecimal and ASCII forms. This additional output is displayed in addition to the standard output. The **dump** keyword can be used with all of the available configuration options of this command.

Table 116 describes the significant fields shown in the display.

Table 116 *debug ip error detail dump Field Descriptions*

Field	Description
IP:s=10.0.19.100 (Ethernet0/1)	The IP packet source IP address and interface.
d=10.1.1.1, len 28	The IP packet destination and prefix length.
dispose udp.noport	The traffic counter that is incremented when a UDP packet is dropped because of this error.

Related Commands

Command	Description
show ip traffic	Displays statistics about IP traffic.

debug ip flow cache

To enable debugging output for NetFlow cache, use the **debug ip flow cache** command in user EXEC or privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip flow cache

no debug ip flow cache

Syntax Description This command has no keywords or arguments.

Defaults Debugging output for NetFlow data export is disabled.

Command Modes User EXEC
Privileged EXEC

Command History	Release	Modification
	12.0(1)	This command was introduced.
	12.3(1)	Debugging output for NetFlow v9 data export was added.
	12.3(7)T	Debugging output for NetFlow for IPv6 was added.

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

```
Router# debug ip flow cache
IP Flow cache allocation debugging is on

Router# show ipv6 flow

IP packet size distribution (0 total packets):
  1-32  64  96 128 160 192 224 256 288 320 352 384 416 448 480
  .000 .000 .000 .000 .000 .000 .000 .000 .000 .000 .000 .000 .000 .000 .000

    512 544 576 1024 1536 2048 2560 3072 3584 4096 4608
    .000 .000 .000 .000 .000 .000 .000 .000 .000 .000 .000

IP Flow Switching Cache, 0 bytes
  0 active, 0 inactive, 0 added
  0 ager polls, 0 flow alloc failures
  Active flows timeout in 30 minutes
  Inactive flows timeout in 15 seconds
SrcAddress                               InpIf  DstAddress
      OutIf  Prot SrcPrt DstPrt Packets
c7200-vxr-2#
000037: 01:56:26: IPFLOW: Allocating Sub-Flow cache, without hash flags.
000038: 01:56:26: IPFLOW: Sub-Flow table enabled.
```

```

000039: 01:56:26: IPFLOW: Sub-Flow numbers are:
      24 sub-flows per chunk, 0 hashflag len,
      1 chunks allocated, 12 max chunks,
      24 allocated records, 24 free records, 960 bytes allocated
000040: 01:56:26: IPFLOW: Sub-Flow cache removed

```

Related Commands	Command	Description
	export destination	Enables the exporting of information from NetFlow aggregation caches.
	ip flow-aggregation cache	Enables NetFlow aggregation cache schemes.
	ip flow-export	Enables the exporting of information in NetFlow cache entries.
	ipv6 flow-aggregation cache	Enables NetFlow aggregation cache schemes for IPv6 configurations.
	ipv6 flow export	Enables the exporting of information in NetFlow cache entries for IPv6 NetFlow configurations.
	show ip cache flow aggregation	Displays the NetFlow aggregation cache configuration.
	show ip flow export	Display the statistics for NetFlow data export.

debug ip flow export

To enable debugging output for NetFlow data export, use the **debug ip flow export** command in user EXEC or privileged EXEC mode. To disable debugging output for NetFlow data export, use the **no** form of this command.

debug ip flow export

no debug ip flow export

Syntax Description This command has no arguments or keywords.

Defaults Debugging output for NetFlow data export is disabled.

Command Modes User EXEC
Privileged EXEC

Command History	Release	Modification
	12.0(1)	This command was introduced.
	12.3(1)	Debugging output for NetFlow v9 data export was added.

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

```
Router# debug ip flow export

IP Flow export mechanism debugging is on
*Mar 6 22:56:21.627:IPFLOW:Sending export pak to 1.1.1.1 port 9999
*Mar 6 22:56:21.627:IPFLOW:Error sending export packet:Adjacency failure
```

Related Commands	Command	Description
	export destination	Enables the exporting of information from NetFlow aggregation caches.
	ip flow-aggregation cache	Enables NetFlow aggregation cache schemes.
	ip flow-export	Enables the exporting of information in NetFlow cache entries.
	show ip cache flow aggregation	Displays the NetFlow aggregation cache configuration.
	show ip flow export	Display the statistics for NetFlow data export.

debug ip http all

To enable debugging output for all HTTP processes on the system, use the **debug ip http all** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip http all

no debug ip http all

Syntax Description This command has no arguments or keywords.

Defaults Disabled

Command Modes Privileged EXEC

Command History	Release	Modification
	12.2(15)T	This command was introduced.

Usage Guidelines This command enables debugging messages for all HTTP processes and activity. Issuing this command is equivalent to issuing the following commands:

- **debug ip http authentication**
- **debug ip http ezsetup**
- **debug ip http ssi**
- **debug ip http token**
- **debug ip http transaction**
- **debug ip http url**

Examples For sample output and field descriptions of this command, see the documentation of the commands listed in the “Usage Guidelines” section.

Related Commands	Command	Description
	debug ip http authentication	Displays authentication processes for HTTP server and client access.
	debug ip http ezsetup	Displays the configuration changes that occur during the EZ Setup process.
	debug ip http server	Enables the HTTP 1.1 server, including the Cisco web browser user interface.

Command	Description
debug ip http ssi	Displays SSI translations and SSI ECHO command execution.
debug ip http ssl error	Displays individual tokens parsed by the HTTP server.
debug ip http transaction	Displays HTTP server transaction processing.
debug ip http url	Displays the URLs accessed from the router.

debug ip http authentication

To troubleshoot HTTP authentication problems, use the **debug ip http authentication** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip http authentication

no debug ip http authentication

Syntax Description This command has no arguments or keywords.

Command Modes Privileged EXEC

Usage Guidelines The **debug ip http authentication** command displays the authentication method the router attempted and authentication-specific status messages.

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

```
Router# debug ip http authentication

Authentication for url '/' '/' level 15 privless '/'
Authentication username = 'local15' priv-level = 15 auth-type = local
```

[Table 117](#) describes the significant fields shown in the display.

Table 117 *debug ip http authentication Field Descriptions*

Field	Description
Authentication for url	Provides information about the URL in different forms.
Authentication username	Identifies the user.
priv-level	Indicates the user privilege level.
auth-type	Indicates the authentication method.

debug ip http client

To enable debugging output for the HTTP client, use the **debug ip http client** command in privileged EXEC mode. To disable debugging client, use the **no** or **undebug** form of this command.

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

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

undebug ip http client {all | api | cache | error | main | msg | socket}

Syntax Description	all	Enables debugging for all HTTP client elements.
	api	Enables debugging output for the HTTP client application interface (API).
	cache	Enables debugging output for the HTTP client cache.
	error	Enables debugging output for HTTP communication errors.
	main	Enables debugging output specific to the Voice XML (VXML) applications interacting with the HTTP client.
	msg	Enables debugging output of HTTP client messages.
	socket	Enables debugging output specific to the HTTP client socket.

Command Modes Privileged EXEC

Command History	Release	Modification
	12.3(2)T	This command was introduced.

Usage Guidelines This command shows transactional information for the HTTP client for debugging purposes.

Examples The following example show sample debugging output for a failed **copy** transfer operation when the host name resolution fails:

```
Router# debug ip http client all

2w4d: Cache ager called
Router# copy http://www.example.com/index.html flash:index.html
Destination filename [index.html]?
Erase flash: before copying? [confirm] no
Translating "www.example.com"

% Bad IP address for host www.example.com
%Error opening http://www.example.com/index.html (I/O error)
Router#

2w4d: http_client_request:

2w4d: httpc_setup_request:
```

```
2w4d: http_client_process_request:
2w4d: HTTPC: Host name resolution failed for www.example.com
2w4d: http_transaction_free:
2w4d: http_transaction_free: freed httpc_transaction_t
```

The following example show sample debugging output for a failed **copy** transfer operation when the source file is not available:

```
Router# copy http://example.com/hi/file.html flash:/file.html
Destination filename [file.html]?
%Error opening http://example.com/hi/file.html (No such file or directory)
Router#
2w4d: http_client_request:
2w4d: httpc_setup_request:
2w4d: http_client_process_request:
2w4d: httpc_request:Dont have the credentials
Thu, 17 Jul 2003 07:05:25 GMT http://209.168.200.225/hi/file.html ok
      Protocol = HTTP/1.1
      Content-Type = text/html; charset=iso-8859-1
      Date = Thu, 17 Jul 2003 14:24:29 GMT
2w4d: http_transaction_free:
2w4d: http_transaction_free:freed httpc_transaction_t
2w4d: http_client_abort_request:
2w4d: http_client_abort_request:Bad Transaction Id
Router#
```

Table 118 describes the significant fields shown in the display.

Table 118 debug ip http client Field Descriptions

Field	Description
2w4d:	In the examples shown, the string “2w4d” is the time-stamp configured on the system. Indicates two weeks and four days since the last system reboot. <ul style="list-style-type: none"> The time stamp format is configured using the service timestamps debug global configuration mode command.
HTTPC: or httpc	Indicates the HTTP client in Cisco IOS software.

Table 118 *debug ip http client Field Descriptions (continued)*

Field	Description
htpc_request:Dont have the credentials	<p>Indicates that this HTTP client request did not supply any authentication information to the server.</p> <p>The authentication information consists of a username and password combination.</p> <p>The message is applicable to both HTTP and Secure HTTP (HTTPS).</p>
Thu, 17 Jul 2003 07:05:25 GMT http://209.168.200.225/hi/file.html ok	<p>The “ok” in this line indicates that there were no internal errors relating to processing this HTTP client transaction by the HTTP client. In other words, the HTTP client was able to send the request and receive some response.</p> <p>Note The “ok” value in this line does not indicate file availability (“200: OK” message or “404: File Not Found” message).</p>

Related Commands

Command	Description
copy	Copies a file from any supported remote location to a local file system, or from a local file system to a remote location, or from a local file system to a local file system.
ip http client connection	Configures the HTTP client connection.
ip http client password	Configures a password for all HTTP client connections.
ip http client proxy-server	Configures an HTTP proxy server.
ip http client source-interface	Configures a source interface for the HTTP client.
ip http client username	Configures a login name for all HTTP client connections.
service timestamps	Configures the time-stamping format for debugging or system logging messages.
show ip http client connection	Displays a report about HTTP client active connections.
show ip http client history	Displays the URLs accessed by the HTTP client.
show ip http client session-module	Displays a report about sessions that have registered with the HTTP client.

debug ip http ezsetup

To display the configuration changes that occur during the EZ Setup process, use the **debug ip http ezsetup** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip http ezsetup

no debug ip http ezsetup

Syntax Description This command has no arguments or keywords.

Command Modes Privileged EXEC

Usage Guidelines Use the **debug ip http ezsetup** command to verify the EZ Setup actions without changing the configuration of the router.

EZ Setup is a form you fill out to perform basic router configuration from most HTML browsers.

Examples The following sample output from the **debug ip http ezsetup** command shows the configuration changes for the router when the EZ Setup form has been submitted:

```
Router# debug ip http ezsetup

service timestamps debug
service timestamps log
service password-encryption
!
hostname router-name
!
enable secret router-pw
line vty 0 4
password router-pw
!
interface ethernet 0
 ip address 172.69.52.9 255.255.255.0
 no shutdown
 ip helper-address 172.31.2.132
 ip name-server 172.31.2.132
 isdn switch-type basic-5ess
 username Remote-name password Remote-chap
interface bri 0
 ip unnumbered ethernet 0
 encapsulation ppp
 no shutdown
 dialer map ip 192.168.254.254 speed 56 name Remote-name Remote-number
 isdn spid1 spid1
 isdn spid2 spid2
 ppp authentication chap callin
 dialer-group 1
!
ip classless
access-list 101 deny udp any any eq snmp
```

```

access-list 101 deny udp any any eq ntp
access-list 101 permit ip any any
dialer-list 1 list 101
ip route 0.0.0.0 0.0.0.0 192.168.254.254
ip route 192.168.254.254 255.255.255.255 bri 0
logging buffered
snmp-server community public RO
ip http server
ip classless
ip subnet-zero
!
end

```

Related Commands

Command	Description
debug ip http ssl error	Displays individual tokens parsed by the HTTP server.
debug ip http transaction	Displays HTTP server transaction processing.
debug ip http url	Displays the URLs accessed from the router.

debug ip http ssi

To display information about the HTML SSI EXEC command or HTML SSI ECHO command, use the **debug ip http ssi** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip http ssi

no debug ip http ssi

Syntax Description This command has no arguments or keywords.

Command Modes Privileged EXEC

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

```
Router# debug ip http ssi

HTML: filtered command `exec cmd="show users"`
HTML: SSI command `exec`
HTML: SSI tag `cmd` = "show users"
HTML: Executing CLI `show users` in mode `exec` done
```

The following line shows the contents of the SSI EXEC command:

```
HTML: filtered command `exec cmd="show users"``
```

The following line indicates the type of SSI command that was requested:

```
HTML: SSI command `exec`
```

The following line shows the argument *show users* assigned to the tag *cmd*:

```
HTML: SSI tag `cmd` = "show users"
```

The following line indicates that the **show users** command is being executed in EXEC mode:

```
HTML: Executing CLI `show users` in mode `exec` done
```

debug ip http ssl error

To enable debugging messages for the HTTP secure (HTTPS) web server and client, use the **debug ip http ssl error** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip http ssl error

no debug ip http ssl error

Syntax Description This command has no arguments or keywords.

Defaults Disabled

Command Modes Privileged EXEC

Command History	Release	Modification
	12.2(15)T	This command was introduced.

Examples None.

Related Commands	Command	Description
	ip http secure-server	Enables the HTTPS server.

debug ip http token

To display individual tokens parsed by the HTTP server, use the **debug ip http token** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip http token

no debug ip http token

Syntax Description This command has no arguments or keywords.

Command Modes Privileged EXEC

Usage Guidelines Use the **debug ip http token** command to display low-level HTTP server parsings. To display high-level HTTP server parsings, use the **debug ip http transaction** command.

Examples The following is part of sample output from the **debug ip http token** command. In this example, the browser accessed the router's home page `http://router-name/`. The output gives the token parsed by the HTTP server and its length.

```
Router# debug ip http token

HTTP: token len 3: 'GET'
HTTP: token len 1: ' '
HTTP: token len 1: '/'
HTTP: token len 1: ' '
HTTP: token len 4: 'HTTP'
HTTP: token len 1: '/'
HTTP: token len 1: '1'
HTTP: token len 1: '.'
HTTP: token len 1: '0'
HTTP: token len 2: '\15\12'
HTTP: token len 7: 'Referer'
HTTP: token len 1: ':'
HTTP: token len 1: ' '
HTTP: token len 4: 'http'
HTTP: token len 1: ':'
HTTP: token len 1: '/'
HTTP: token len 1: '/'
HTTP: token len 3: 'www'
HTTP: token len 1: '.'
HTTP: token len 3: 'thesite'
HTTP: token len 1: '.'
HTTP: token len 3: 'com'
HTTP: token len 1: '/'
HTTP: token len 2: '\15\12'
HTTP: token len 10: 'Connection'
HTTP: token len 1: ':'
HTTP: token len 1: ' '
HTTP: token len 4: 'Keep'
HTTP: token len 1: '-'
HTTP: token len 5: 'Alive'
HTTP: token len 2: '\15\12'
```

```

HTTP: token len 4: 'User'
HTTP: token len 1: '-'
HTTP: token len 5: 'Agent'
HTTP: token len 1: ':'
HTTP: token len 1: ' '
HTTP: token len 7: 'Mozilla'
HTTP: token len 1: '/'
HTTP: token len 1: '2'
HTTP: token len 1: '.'
.
.
.

```

Related Commands

Command	Description
debug ip http ezsetup	Displays the configuration changes that occur during the EZ Setup process.
debug ip http transaction	Displays HTTP server transaction processing.
debug ip http url	Displays the URLs accessed from the router.

debug ip http transaction

To display HTTP server transaction processing, use the **debug ip http transaction** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip http transaction

no debug ip http transaction

Syntax Description This command has no arguments or keywords.

Command Modes Privileged EXEC

Usage Guidelines Use the **debug ip http transaction** command to display what the HTTP server is parsing at a high level. To display what the HTTP server is parsing at a low level, use the **debug ip http token** command.

Examples The following is sample output from the **debug ip http transaction** command. In this example, the browser accessed the router's home page `http://router-name/`.

```
Router# debug ip http transaction

HTTP: parsed uri '/'
HTTP: client version 1.0
HTTP: parsed extension Referer
HTTP: parsed line http://www.company.com/
HTTP: parsed extension Connection
HTTP: parsed line Keep-Alive
HTTP: parsed extension User-Agent
HTTP: parsed line Mozilla/2.01 (X11; I; FreeBSD 2.1.0-RELEASE i386)
HTTP: parsed extension Host
HTTP: parsed line router-name
HTTP: parsed extension Accept
HTTP: parsed line image/gif, image/x-xbitmap, image/jpeg, image/
HTTP: parsed extension Authorization
HTTP: parsed authorization type Basic
HTTP: received GET ''
```

[Table 119](#) describes the significant fields shown in the display.

Table 119 *debug ip http transaction Field Descriptions*

Field	Description
HTTP: parsed uri '/'	Uniform resource identifier that is requested.
HTTP: client version 1.0	Client HTTP version.
HTTP: parsed extension Referer	HTTP extension.
HTTP: parsed line http://www.company.com/	Value of HTTP extension.
HTTP: received GET ''	HTTP request method.

Related Commands	Command	Description
	debug ip http ezsetup	Displays the configuration changes that occur during the EZ Setup process.
	debug ip http ssl error	Displays individual tokens parsed by the HTTP server.
	debug ip http url	Shows the URLs accessed from the router.

debug ip http url

To show the URLs accessed from the router, use the **debug ip http url** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip http url

no debug ip http url

Syntax Description This command has no arguments or keywords.

Command Modes Privileged EXEC

Usage Guidelines Use the **debug ip http url** command to keep track of the URLs that are accessed and to determine from which hosts the URLs are accessed.

Examples The following is sample output from the **debug ip http url** command. In this example, the HTTP server accessed the URLs and /exec. The output shows the URL being requested and the IP address of the host requesting the URL.

```
Router# debug ip http url

HTTP: processing URL '/' from host 172.31.2.141
HTTP: processing URL '/exec' from host 172.31.2.141
```

Related Commands	Command	Description
	debug ip http ezsetup	Displays the configuration changes that occur during the EZ Setup process.
	debug ip http ssl error	Displays individual tokens parsed by the HTTP server.
	debug ip http transaction	Displays HTTP server transaction processing.

debug ip icmp

To display information on Internal Control Message Protocol (ICMP) transactions, use the **debug ip icmp** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip icmp

no debug ip icmp

Syntax Description This command has no arguments or keywords.

Command Modes Privileged EXEC

Usage Guidelines This command helps you determine whether the router is sending or receiving ICMP messages. Use it, for example, when you are troubleshooting an end-to-end connection problem.



Note

For more information about the fields in **debug ip icmp** command output, refer to RFC 792, *Internet Control Message Protocol*; Appendix I of RFC 950, *Internet Standard Subnetting Procedure*; and RFC 1256, *ICMP Router Discovery Messages*.

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

```
Router# debug ip icmp

ICMP: rcvd type 3, code 1, from 10.95.192.4
ICMP: src 10.56.0.202, dst 172.69.16.1, echo reply
ICMP: dst (10.120.1.0) port unreachable rcv from 10.120.1.15
ICMP: src 172.69.12.35, dst 172.69.20.7, echo reply
ICMP: dst (255.255.255.255) protocol unreachable rcv from 10.31.7.21
ICMP: dst (10.120.1.0) port unreachable rcv from 10.120.1.15
ICMP: dst (255.255.255.255) protocol unreachable rcv from 10.31.7.21
ICMP: dst (10.120.1.0) port unreachable rcv from 10.120.1.15
ICMP: src 10.56.0.202, dst 172.69.16.1, echo reply
ICMP: dst (10.120.1.0) port unreachable rcv from 10.120.1.15
ICMP: dst (255.255.255.255) protocol unreachable rcv from 10.31.7.21
ICMP: dst (10.120.1.0) port unreachable rcv from 10.120.1.15
```

Table 120 describes the significant fields shown in the display.

Table 120 *debug ip icmp Field Descriptions*

Field	Description
ICMP:	Indication that this message describes an ICMP packet.
rcvd type 3	<p>The type field can be one of the following:</p> <ul style="list-style-type: none"> • 0—Echo Reply • 3—Destination Unreachable • 4—Source Quench • 5—Redirect • 8—Echo • 9—Router Discovery Protocol Advertisement • 10—Router Discovery Protocol Solicitations • 11—Time Exceeded • 12—Parameter Problem • 13—Timestamp • 14—Timestamp Reply • 15—Information Request • 16—Information Reply • 17—Mask Request • 18—Mask Reply

Table 120 debug ip icmp Field Descriptions (continued)

Field	Description
code 1	<p>This field is a code. The meaning of the code depends upon the type field value, as follows:</p> <ul style="list-style-type: none"> • Echo and Echo Reply—The code field is always zero. • Destination Unreachable—The code field can have the following values: <ul style="list-style-type: none"> 0—Network unreachable 1—Host unreachable 2—Protocol unreachable 3—Port unreachable 4—Fragmentation needed and DF bit set 5—Source route failed • Source Quench—The code field is always 0. • Redirect—The code field can have the following values: <ul style="list-style-type: none"> 0—Redirect datagrams for the network 1—Redirect datagrams for the host 2—Redirect datagrams for the command mode of service and network 3—Redirect datagrams for the command mode of service and host • Router Discovery Protocol Advertisements and Solicitations—The code field is always zero. • Time Exceeded—The code field can have the following values: <ul style="list-style-type: none"> 0—Time to live exceeded in transit 1—Fragment reassembly time exceeded • Parameter Problem—The code field can have the following values: <ul style="list-style-type: none"> 0—General problem 1—Option is missing 2—Option missing, no room to add • Timestamp and Timestamp Reply—The code field is always zero. • Information Request and Information Reply—The code field is always zero. • Mask Request and Mask Reply—The code field is always zero.
from 10.95.192.4	Source address of the ICMP packet.

Table 121 describes the significant fields shown in the second line of the display.

Table 121 *debug ip icmp Field Descriptions*

Field	Description
ICMP:	Indicates that this message describes an ICMP packet.
src 10.56.10.202	Address of the sender of the echo.
dst 172.69.16.1	Address of the receiving router.
echo reply	Indicates that the router received an echo reply.

Other messages that the **debug ip icmp** command can generate follow.

When an IP router or host sends out an ICMP mask request, the following message is generated when the router sends a mask reply:

```
ICMP: sending mask reply (255.255.255.0) to 172.69.80.23 via Ethernet0
```

The following two lines are examples of the two forms of this message. The first form is generated when a mask reply comes in after the router sends out a mask request. The second form occurs when the router receives a mask reply with a nonmatching sequence and ID. Refer to Appendix I of RFC 950, *Internet Standard Subnetting Procedures*, for details.

```
ICMP: mask reply 255.255.255.0 from 172.69.80.31
ICMP: unexpected mask reply 255.255.255.0 from 172.69.80.32
```

The following output indicates that the router sent a redirect packet to the host at address 172.69.80.31, instructing that host to use the gateway at address 172.69.80.23 in order to reach the host at destination address 172.69.1.111:

```
ICMP: redirect sent to 172.69.80.31 for dest 172.69.1.111 use gw 172.69.80.23
```

The following message indicates that the router received a redirect packet from the host at address 172.69.80.23, instructing the router to use the gateway at address 172.69.80.28 in order to reach the host at destination address 172.69.81.34:

```
ICMP: redirect rcvd from 172.69.80.23 -- for 172.69.81.34 use gw 172.69.80.28
```

The following message is displayed when the router sends an ICMP packet to the source address (172.69.94.31 in this case), indicating that the destination address (172.69.13.33 in this case) is unreachable:

```
ICMP: dst (172.69.13.33) host unreachable sent to 172.69.94.31
```

The following message is displayed when the router receives an ICMP packet from an intermediate address (172.69.98.32 in this case), indicating that the destination address (172.69.13.33 in this case) is unreachable:

```
ICMP: dst (172.69.13.33) host unreachable rcv from 172.69.98.32
```

Depending on the code received (as Table 120 describes), any of the unreachable messages can have any of the following “strings” instead of the “host” string in the message:

```
net
protocol
port
frag. needed and DF set
source route failed
prohibited
```

The following message is displayed when the TTL in the IP header reaches zero and a time exceed ICMP message is sent. The fields are self-explanatory.

```
ICMP: time exceeded (time to live) send to 10.95.1.4 (dest was 172.69.1.111)
```

The following message is generated when parameters in the IP header are corrupted in some way and the parameter problem ICMP message is sent. The fields are self-explanatory.

```
ICMP: parameter problem sent to 128.121.1.50 (dest was 172.69.1.111)
```

Based on the preceding information, the remaining output can be easily understood:

```
ICMP: parameter problem rcvd 172.69.80.32
ICMP: source quench rcvd 172.69.80.32
ICMP: source quench sent to 128.121.1.50 (dest was 172.69.1.111)
ICMP: sending time stamp reply to 172.69.80.45
ICMP: sending info reply to 172.69.80.12
ICMP: rdp advert rcvd type 9, code 0, from 172.69.80.23
ICMP: rdp solicit rcvd type 10, code 0, from 172.69.80.43
```

debug ip igmp

To display Internet Group Management Protocol (IGMP) packets received and sent, and IGMP-host related events, use the **debug ip igmp** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

```
debug ip igmp [vrf vrf-name] [group-address]
```

```
no debug ip igmp [vrf vrf-name] [group-address]
```

Syntax Description

vrf	(Optional) Supports the multicast Virtual Private Network (VPN) routing and forwarding (VRF) instance.
<i>vrf-name</i>	(Optional) Name assigned to the VRF.
<i>group-address</i>	(Optional) Address of a particular group about which to display IGMP information.

Defaults

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
10.2	This command was introduced.
12.1(3)T	Fields were added to the output of this command to support the Source Specific Multicast (SSM) feature.
12.0(23)S	The vrf keyword and <i>vrf-name</i> argument were added.
12.2(13)T	This command was integrated into Cisco IOS Release 12.2(13)T.
12.3(2)T	Fields were added to the output of this command to support the SSM Mapping feature. The <i>group-address</i> attribute was added.

Usage Guidelines

This command helps discover whether the IGMP processes are functioning. In general, if IGMP is not working, the router process never discovers that another host is on the network that is configured to receive multicast packets. In dense mode, this situation will result in packets being delivered intermittently (a few every 3 minutes). In sparse mode, packets will never be delivered.

Use this command in conjunction with the **debug ip pim** and **debug ip mrouting** commands to observe additional multicast activity and to learn the status of the multicast routing process, or why packets are forwarded out of particular interfaces.

When SSM mapping is enabled, a debug message is displayed to indicate that the router is converting an IGMP version 2 report from the group (G) into an IGMP version 3 join. After SSM mapping has generated the appropriate IGMP version 3 report, any debug output that follows is seen as if the router had received the same IGMP version 3 report directly.

Examples

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

```
Router# debug ip igmp
IGMP: Received Host-Query from 172.16.37.33 (Ethernet1)
IGMP: Received Host-Report from 172.16.37.192 (Ethernet1) for 224.0.255.1
IGMP: Received Host-Report from 172.16.37.57 (Ethernet1) for 224.2.127.255
IGMP: Received Host-Report from 172.16.37.33 (Ethernet1) for 225.2.2.2
```

The messages displayed by the **debug ip igmp** command show query and report activity received from other routers and multicast group addresses.

The following is sample output from the **debug ip igmp** command when SSM is enabled. Because IGMP version 3 lite (IGMP v3lite) requires the host to send IGMP version 2 (IGMPv2) packets, IGMPv2 host reports also will be displayed in response to the router IGMPv2 queries. If SSM is disabled, the word “ignored” will be displayed in the **debug ip igmp** command output.

```
IGMP:Received v3-lite Report from 10.0.119.142 (Ethernet3/3), group count 1
IGMP:Received v3 Group Record from 10.0.119.142 (Ethernet3/3) for 232.10.10.10
IGMP:Update source 224.1.1.1
IGMP:Send v2 Query on Ethernet3/3 to 224.0.0.1
IGMP:Received v2 Report from 10.0.119.142 (Ethernet3/3) for 232.10.10.10
IGMP:Update source 224.1.1.1
```

The following is sample output from the **debug ip igmp** command when SSM static mapping is enabled. The following output indicates that the router is converting an IGMP version 2 join for group (G) into an IGMP version 3 join:

```
IGMP(0): Convert IGMPv2 report (*,232.1.2.3) to IGMPv3 with 2 source(s) using STATIC.
```

The following is sample output from the **debug ip igmp** command when SSM DNS-based mapping is enabled. The following output indicates that a DNS lookup has succeeded:

```
IGMP(0): Convert IGMPv2 report (*,232.1.2.3) to IGMPv3 with 2 source(s) using DNS.
```

The following is sample output from the **debug ip igmp** command when SSM DNS-based mapping is enabled and a DNS lookup has failed:

```
IGMP(0): DNS source lookup failed for (*, 232.1.2.3), IGMPv2 report failed
```

Related Commands

Command	Description
debug ip mrm	Displays MRM control packet activity.
debug ip mrouting	Displays changes to the mroute table.
debug ip pim	Displays PIM packets received and sent and PIM-related events.

debug ip igmp snooping

To display debugging messages about Internet Group Management Protocol (IGMP) snooping services, use the **debug ip igmp snooping** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip igmp snooping { **group** | **management** | **router** | **timer** }

no debug ip igmp snooping { **group** | **management** | **router** | **timer** }

Syntax Description

group	Displays debugging messages related to multicast groups.
management	Displays debugging messages related to IGMP management services.
router	Displays debugging messages related to the local router.
timer	Displays debugging messages related to the IGMP timer.

Defaults

Debugging is disabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.1(6)EA2	This command was introduced.
12.2(15)ZJ	This command was implemented on the following platforms: Cisco 2600 series, Cisco 3600 series, and Cisco 3700 series routers.
12.3(4)T	This command was integrated into Cisco IOS Release 12.3(4)T on the following platforms: Cisco 2600 series, Cisco 3600 series, and Cisco 3700 series routers.

Examples

The following example shows debugging messages for the IGMP snooping services being displayed:

```
Router# debug ip igmp snooping
```

```
IGMP snooping enabled
```

Related Commands

Command	Description
show ip igmp snooping	Displays the IGMP snooping configuration.

debug ip igrp events

To display summary information on Interior Gateway Routing Protocol (IGRP) routing messages that indicate the source and destination of each update, and the number of routes in each update, use the **debug ip igrp events** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip igrp events [*ip-address*]

no debug ip igrp events [*ip-address*]

Syntax Description	<i>ip-address</i> (Optional) The IP address of an IGRP neighbor.
Command Modes	Privileged EXEC
Usage Guidelines	<p>If the IP address of an IGRP neighbor is specified, the resulting debug ip igrp events output includes messages describing updates from that neighbor and updates that the router broadcasts toward that neighbor. Messages are not generated for each route.</p> <p>This command is particularly useful when there are many networks in your routing table. In this case, using debug ip igrp transactions could flood the console and make the router unusable. Use debug ip igrp events instead to display summary routing information.</p>
Examples	The following is sample output from the debug ip igrp events command:

```

router# debug ip igrp events

Updates sent to these two destination addresses
----- IGRP: sending update to 255.255.255.255 via Ethernet1 (160.89.33.8)
IGRP: Update contains 26 interior, 40 system, and 3 exterior routes.
IGRP: Total routes in update: 69
----- IGRP: sending update to 255.255.255.255 via Ethernet0 (160.89.32.8)
IGRP: Update contains 1 interior, 0 system, and 0 exterior routes.
IGRP: Total routes in update: 1
Updates received from these source addresses
----- IGRP: received update from 160.89.32.24 on Ethernet0
IGRP: Update contains 17 interior, 1 system, and 0 exterior routes.
IGRP: Total routes in update: 18
----- IGRP: received update from 160.89.32.7 on Ethernet0
IGRP: Update contains 5 interior, 1 system, and 0 exterior routes.
IGRP: Total routes in update: 6

```

S2548

This shows that the router has sent two updates to the broadcast address 255.255.255.255. The router also received two updates. Three lines of output describe each of these updates.

The first line indicates whether the router sent or received the update packet, the source or destination address, and the interface through which the update was sent or received. If the update was sent, the IP address assigned to this interface is shown (in parentheses).

```
IGRP: sending update to 255.255.255.255 via Ethernet1 (160.89.33.8)
```

The second line summarizes the number and types of routes described in the update:

```
IGRP: Update contains 26 interior, 40 system, and 3 exterior routes.
```

The third line indicates the total number of routes described in the update:

```
IGRP: Total routes in update: 69
```

debug ip igrp transactions

To display transaction information on Interior Gateway Routing Protocol (IGRP) routing transactions, use the **debug ip igrp transactions** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip igrp transactions [*ip-address*]

no debug ip igrp transactions [*ip-address*]

Syntax Description	<i>ip-address</i> (Optional) The IP address of an IGRP neighbor.
Command Modes	Privileged EXEC
Usage Guidelines	<p>If the IP address of an IGRP neighbor is specified, the resulting debug ip igrp transactions output includes messages describing updates from that neighbor and updates that the router broadcasts toward that neighbor.</p> <p>When many networks are in your routing table, the debug ip igrp transactions command can flood the console and make the router unusable. In this case, use the debug ip igrp events command instead to display summary routing information.</p>
Examples	<p>The following is sample output from the debug ip igrp transactions command:</p> <pre style="margin-left: 40px;"> Router# debug ip igrp transactions Updates received from these two source addresses ----- IGRP: received update from 160.89.80.240 on Ethernet subnet 160.89.66.0, metric 1300 (neighbor 1200) subnet 160.89.56.0, metric 8676 (neighbor 8576) subnet 160.89.48.0, metric 1200 (neighbor 1100) subnet 160.89.50.0, metric 1300 (neighbor 1200) subnet 160.89.40.0, metric 8676 (neighbor 8576) network 192.82.152.0, metric 158550 (neighbor 158450) network 192.68.151.0, metric 1115511 (neighbor 1115411) network 150.136.0.0, metric 16777215 (inaccessible) exterior network 129.140.0.0, metric 9676 (neighbor 9576) exterior network 140.222.0.0, metric 9676 (neighbor 9576) IGRP: received update from 160.89.80.28 on Ethernet subnet 160.89.95.0, metric 180671 (neighbor 180571) subnet 160.89.81.0, metric 1200 (neighbor 1100) subnet 160.89.15.0, metric 16777215 (inaccessible) Updates sent to these two destination addresses ----- IGRP: sending update to 255.255.255.255 via Ethernet0 (160.89.64.31) subnet 160.89.94.0, metric=847 IGRP: sending update to 255.255.255.255 via Serial1 (160.89.94.31) subnet 160.89.80.0, metric=16777215 subnet 160.89.64.0, metric=1100 </pre>

S2549

The output shows that the router being debugged has received updates from two other routers on the network. The router at source address 160.89.80.240 sent information about ten destinations in the update; the router at source address 160.89.80.28 sent information about three destinations in its update. The router being debugged also sent updates—in both cases to the broadcast address 255.255.255.255 as the destination address.

On the second line the first field refers to the type of destination information: “subnet” (interior), “network” (system), or “exterior” (exterior). The second field is the Internet address of the destination network. The third field is the metric stored in the routing table and the metric advertised by the neighbor sending the information. “Metric... inaccessible” usually means that the neighbor router has put the destination in a hold down state.

The entries show that the router is sending updates that are similar, except that the numbers in parentheses are the source addresses used in the IP header. A metric of 16777215 is inaccessible.

Other examples of output that the **debug ip igrp transactions** command can produce follow.

The following entry indicates that the routing table was updated and shows the new edition number (97 in this case) to be used in the next IGRP update:

```
IGRP: edition is now 97
```

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

```
IGRP: broadcasting request on Ethernet0  
IGRP: broadcasting request on Ethernet1
```

The following type of entry can result when routing updates become corrupted between sending and receiving routers:

```
IGRP: bad checksum from 172.69.64.43
```

An entry such as the following should never appear. If it does, the receiving router has a bug in the software or a problem with the hardware. In either case, contact your technical support representative.

```
IGRP: system 45 from 172.69.64.234, should be system 109
```

debug ip inspect

To display messages about Cisco IOS Firewall events, use the **debug ip inspect** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip inspect { **function-trace** | **object-creation** | **object-deletion** | **events** | **timers** | *protocol* | **detailed** }

no debug ip inspect

Syntax	Description
function-trace	Displays messages about software functions called by the Cisco IOS Firewall.
object-creation	Displays messages about software objects being created by the Cisco IOS Firewall. Object creation corresponds to the beginning of Cisco IOS Firewall-inspected sessions.
object-deletion	Displays messages about software objects being deleted by the Cisco IOS Firewall. Object deletion corresponds to the closing of Cisco IOS Firewall-inspected sessions.
events	Displays messages about Cisco IOS Firewall software events, including information about Cisco IOS Firewall packet processing.
timers	Displays messages about Cisco IOS Firewall timer events such as when the Cisco IOS Firewall idle timeout is reached.
<i>protocol</i>	Displays messages about Cisco IOS Firewall-inspected protocol events, including details about the packets of the protocol. Table 122 provides a list of <i>protocol</i> keywords.
detailed	Displays detailed information to be displayed for all the other enabled Cisco IOS Firewall debugging. Use this form of the command in conjunction with other Cisco IOS Firewall debug commands.

Table 122 Protocol Keywords for the debug ip inspect Command

Application Protocol	Protocol Keyword
Transport-layer protocols	
ICMP	icmp
TCP	tcp
User Datagram Protocol (UDP)	udp
Application-layer protocols	
CU-SeeMe	cuseeme
FTP commands and responses	ftp-cmd
FTP tokens (enables tracing of the FTP tokens parsed)	ftp-tokens
H.323 (version 1 and version 2)	h323
HTTP	http
IMAP	imap
Microsoft NetShow	netshow

Table 122 Protocol Keywords for the debug ip inspect Command (continued)

Application Protocol	Protocol Keyword
POP3	pop3
RealAudio	realaudio
Remote procedure call (RPC)	rpc
Real Time Streaming Protocol (RTSP)	rtsp
Session Initiation Protocol (SIP)	sip
Simple Mail Transfer Protocol (SMTP)	smtp
Skinny Client Control Protocol (SCCP)	skinny
Structured Query Language*Net (SQL*Net)	sqlnet
StreamWorks	streamworks
TFTP	tftp
UNIX r-commands (rlogin, rexec, rsh)	rcmd
VDOLive	vdolive

Command Modes

Privileged EXEC

Command History

Release	Modification
11.2 P	This command was introduced.
12.0(5)T	NetShow support was added.
12.0(7)T	H.323 V2 and RTSP protocol support were added.
12.2(11)YU	Support for the ICMP and SIP protocols was added.
12.2(15)T	This command was integrated into Cisco IOS Release 12.2(15)T.
12.3(1)	Support for the skinny protocol was added.
12.3(14)T	Support for the IMAP and POP3 protocols was added.

Examples

The following is sample output from the **debug ip inspect function-trace** command:

```
Router# debug ip inspect function-trace

*Mar  2 01:16:16: CBAC FUNC: insp_inspection
*Mar  2 01:16:16: CBAC FUNC: insp_pre_process_sync
*Mar  2 01:16:16: CBAC FUNC: insp_find_tcp_host_entry addr 40.0.0.1 bucket 41
*Mar  2 01:16:16: CBAC FUNC: insp_find_pregen_session
*Mar  2 01:16:16: CBAC FUNC: insp_get_idbsb
*Mar  2 01:16:16: CBAC FUNC: insp_get_idbsb
*Mar  2 01:16:16: CBAC FUNC: insp_get_irc_of_idb
*Mar  2 01:16:16: CBAC FUNC: insp_get_idbsb
*Mar  2 01:16:16: CBAC FUNC: insp_create_sis
*Mar  2 01:16:16: CBAC FUNC: insp_inc_halfopen_sis
*Mar  2 01:16:16: CBAC FUNC: insp_link_session_to_hash_table
*Mar  2 01:16:16: CBAC FUNC: insp_inspect_pak
*Mar  2 01:16:16: CBAC FUNC: insp_l4_inspection
*Mar  2 01:16:16: CBAC FUNC: insp_process_tcp_seg
```

```

*Mar 2 01:16:16: CBAC FUNC: insp_listen_state
*Mar 2 01:16:16: CBAC FUNC: insp_ensure_return_traffic
*Mar 2 01:16:16: CBAC FUNC: insp_add_acl_item
*Mar 2 01:16:16: CBAC FUNC: insp_ensure_return_traffic
*Mar 2 01:16:16: CBAC FUNC: insp_add_acl_item
*Mar 2 01:16:16: CBAC FUNC: insp_process_syn_packet
*Mar 2 01:16:16: CBAC FUNC: insp_find_tcp_host_entry addr 40.0.0.1 bucket 41
*Mar 2 01:16:16: CBAC FUNC: insp_create_tcp_host_entry
*Mar 2 01:16:16: CBAC* FUNC: insp_fast_inspection
*Mar 2 01:16:16: CBAC* FUNC: insp_inspect_pak
*Mar 2 01:16:16: CBAC* FUNC: insp_l4_inspection
*Mar 2 01:16:16: CBAC* FUNC: insp_process_tcp_seg
*Mar 2 01:16:16: CBAC* FUNC: insp_synrcvd_state
*Mar 2 01:16:16: CBAC* FUNC: insp_fast_inspection
*Mar 2 01:16:16: CBAC* FUNC: insp_inspect_pak
*Mar 2 01:16:16: CBAC* FUNC: insp_l4_inspection
*Mar 2 01:16:16: CBAC* FUNC: insp_process_tcp_seg
*Mar 2 01:16:16: CBAC* FUNC: insp_synrcvd_state
*Mar 2 01:16:16: CBAC FUNC: insp_dec_halfopen_sis
*Mar 2 01:16:16: CBAC FUNC: insp_remove_sis_from_host_entry
*Mar 2 01:16:16: CBAC FUNC: insp_find_tcp_host_entry addr 40.0.0.1 bucket 41

```

This output shows the functions called by the Cisco IOS Firewall as a session is inspected. Entries with an asterisk (*) after the word “CBAC” are entries when the fast path is used; otherwise, the process path is used.

The following is sample output from the **debug ip inspect object-creation** and **debug ip inspect object-deletion** commands:

```

Router# debug ip inspect object-creation
Router# debug ip inspect object-deletion

*Mar 2 01:18:30: CBAC OBJ_CREATE: create pre-gen sis 25A3574
*Mar 2 01:18:30: CBAC OBJ_CREATE: create acl wrapper 25A36FC -- acl item 25A3634
*Mar 2 01:18:30: CBAC OBJ_CREATE: create sis 25C1CC4
*Mar 2 01:18:30: CBAC OBJ_DELETE: delete pre-gen sis 25A3574
*Mar 2 01:18:30: CBAC OBJ_CREATE: create host entry 25A3574 addr 10.0.0.1 bucket 31
*Mar 2 01:18:30: CBAC OBJ_DELETE: delete sis 25C1CC4
*Mar 2 01:18:30: CBAC OBJ_DELETE: delete create acl wrapper 25A36FC -- acl item 25A3634
*Mar 2 01:18:31: CBAC OBJ_DELETE: delete host entry 25A3574 addr 10.0.0.1

```

The following is sample output from the **debug ip inspect object-creation**, **debug ip inspect object-deletion**, and **debug ip inspect events** commands:

```

Router# debug ip inspect object-creation
Router# debug ip inspect object-deletion
Router# debug ip inspect events

*Mar 2 01:18:51: CBAC OBJ_CREATE: create pre-gen sis 25A3574
*Mar 2 01:18:51: CBAC OBJ_CREATE: create acl wrapper 25A36FC -- acl item 25A3634
*Mar 2 01:18:51: CBAC Src 10.1.0.1 Port [1:65535]
*Mar 2 01:18:51: CBAC Dst 10.0.0.1 Port [46406:46406]
*Mar 2 01:18:51: CBAC Pre-gen sis 25A3574 created: 10.1.0.1[1:65535]
30.0.0.1[46406:46406]
*Mar 2 01:18:51: CBAC OBJ_CREATE: create sis 25C1CC4
*Mar 2 01:18:51: CBAC sis 25C1CC4 initiator_addr (10.1.0.1:20) responder_addr
(30.0.0.1:46406) initiator_alt_addr (40.0.0.1:20) responder_alt_addr (10.0.0.1:46406)
*Mar 2 01:18:51: CBAC OBJ_DELETE: delete pre-gen sis 25A3574
*Mar 2 01:18:51: CBAC OBJ_CREATE: create host entry 25A3574 addr 10.0.0.1 bucket 31
*Mar 2 01:18:51: CBAC OBJ_DELETE: delete sis 25C1CC4
*Mar 2 01:18:51: CBAC OBJ_DELETE: delete create acl wrapper 25A36FC -- acl item 25A3634
*Mar 2 01:18:51: CBAC OBJ_DELETE: delete host entry 25A3574 addr 10.0.0.1

```

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

```
Router# debug ip inspect timers

*Mar  2 01:19:15: CBAC Timer Init Leaf: Pre-gen sis 25A3574
*Mar  2 01:19:15: CBAC Timer Start: Pre-gen sis 25A3574 Timer: 25A35D8 Time: 30000
milisecs
*Mar  2 01:19:15: CBAC Timer Init Leaf: sis 25C1CC4
*Mar  2 01:19:15: CBAC Timer Stop: Pre-gen sis 25A3574 Timer: 25A35D8
*Mar  2 01:19:15: CBAC Timer Start: sis 25C1CC4 Timer: 25C1D5C Time: 30000 milisecs
*Mar  2 01:19:15: CBAC Timer Start: sis 25C1CC4 Timer: 25C1D5C Time: 3600000 milisecs
*Mar  2 01:19:15: CBAC Timer Start: sis 25C1CC4 Timer: 25C1D5C Time: 5000 milisecs
*Mar  2 01:19:15: CBAC Timer Stop: sis 25C1CC4 Timer: 25C1D5C
```

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

```
Router# debug ip inspect tcp

*Mar  2 01:20:43: CBAC* sis 25A3604 pak 2541C58 TCP P ack 4223720032 seq 4200176225(22)
(10.0.0.1:46409) => (10.1.0.1:21)
*Mar  2 01:20:43: CBAC* sis 25A3604 ftp L7 inspect result: PROCESS-SWITCH packet
*Mar  2 01:20:43: CBAC sis 25A3604 pak 2541C58 TCP P ack 4223720032 seq 4200176225(22)
(10.0.0.1:46409) => (10.1.0.1:21)
*Mar  2 01:20:43: CBAC sis 25A3604 ftp L7 inspect result: PASS packet
*Mar  2 01:20:43: CBAC* sis 25A3604 pak 2544374 TCP P ack 4200176247 seq 4223720032(30)
(10.0.0.1:46409) <= (10.1.0.1:21)
*Mar  2 01:20:43: CBAC* sis 25A3604 ftp L7 inspect result: PASS packet
*Mar  2 01:20:43: CBAC* sis 25A3604 pak 25412F8 TCP P ack 4223720062 seq 4200176247(15)
(10.0.0.1:46409) => (10.1.0.1:21)
*Mar  2 01:20:43: CBAC* sis 25A3604 ftp L7 inspect result: PASS packet
*Mar  2 01:20:43: CBAC sis 25C1CC4 pak 2544734 TCP S seq 4226992037(0) (10.1.0.1:20) =>
(10.0.0.1:46411)
*Mar  2 01:20:43: CBAC* sis 25C1CC4 pak 2541E38 TCP S ack 4226992038 seq 4203405054(0)
(10.1.0.1:20) <= (10.0.0.1:46411)
```

This sample shows TCP packets being processed and lists the corresponding acknowledge (ACK) packet numbers and sequence (SEQ) numbers. The number of data bytes in the TCP packet is shown in parentheses—for example, (22). For each packet shown, the addresses and port numbers are shown separated by a colon. For example, (10.1.0.1:21) indicates an IP address of 10.1.0.1 and a TCP port number of 21.

Entries with an asterisk (*) after the word “CBAC” are entries when the fast path is used; otherwise, the process path is used.

The following is sample output from the **debug ip inspect tcp** and **debug ip inspect detailed** commands:

```
Router# debug ip inspect tcp
Router# debug ip inspect detailed

*Mar  2 01:20:58: CBAC* Pak 2541E38 Find session for (30.0.0.1:46409) (40.0.0.1:21) tcp
*Mar  2 01:20:58: P ack 4223720160 seq 4200176262(22)
*Mar  2 01:20:58: CBAC* Pak 2541E38 Addr:port pairs to match: (30.0.0.1:46409)
(40.0.0.1:21)
*Mar  2 01:20:58: CBAC* sis 25A3604 SIS_OPEN
*Mar  2 01:20:58: CBAC* Pak 2541E38 IP: s=30.0.0.1 (Ethernet0), d=40.0.0.1 (Ethernet1),
len 76,proto=6
*Mar  2 01:20:58: CBAC sis 25A3604 Saving State: SIS_OPEN/ESTAB iisn 4200176160 i_rcvnxt
4223720160 i_sndnxt 4200176262 i_rcvwnd 8760 risn 4223719771 r_rcvnxt 4200176262 r_sndnxt
4223720160 r_rcvwnd 8760
*Mar  2 01:20:58: CBAC* sis 25A3604 pak 2541E38 TCP P ack 4223720160 seq 4200176262(22)
(30.0.0.1:46409) => (40.0.0.1:21)
*Mar  2 01:20:58: CBAC* sis 25A3604 pak 2541E38 SIS_OPEN/ESTAB TCP seq 4200176262(22)
Flags: ACK 4223720160 PSH
```

```

*Mar 2 01:20:58: CBAC* sis 25A3604 pak 2541E38 --> SIS_OPEN/ESTAB iisn 4200176160
i_rcvnxt 4223720160 i_sndnxt 4200176284 i_rcvwnd 8760 risn 4223719771 r_rcvnxt 4200176262
r_sndnxt 4223720160 r_rcvwnd 8760
*Mar 2 01:20:58: CBAC* sis 25A3604 L4 inspect result: PASS packet 2541E38
(30.0.0.1:46409) (40.0.0.1:21) bytes 22 ftp
*Mar 2 01:20:58: CBAC sis 25A3604 Restoring State: SIS_OPEN/ESTAB iisn 4200176160
i_rcvnxt 4223
720160 i_sndnxt 4200176262 i_rcvwnd 8760 risn 4223719771 r_rcvnxt 4200176262 r_sndnxt
4223720160 r_rcvwnd 8760
*Mar 2 01:20:58: CBAC* sis 25A3604 ftp L7 inspect result: PROCESS-SWITCH packet
*Mar 2 01:20:58: CBAC* sis 25A3604 ftp L7 inspect result: PROCESS-SWITCH packet
*Mar 2 01:20:58: CBAC* Bump up: inspection requires the packet in the process
path(30.0.0.1) (40.0.0.1)
*Mar 2 01:20:58: CBAC Pak 2541E38 Find session for (30.0.0.1:46409) (40.0.0.1:21) tcp
*Mar 2 01:20:58: P ack 4223720160 seq 4200176262(22)
*Mar 2 01:20:58: CBAC Pak 2541E38 Addr:port pairs to match: (30.0.0.1:46409)
(40.0.0.1:21)
*Mar 2 01:20:58: CBAC sis 25A3604 SIS_OPEN
*Mar 2 01:20:58: CBAC Pak 2541E38 IP: s=30.0.0.1 (Ethernet0), d=40.0.0.1 (Ethernet1), len
76, proto=6

```

The following is sample output from the **debug ip inspect icmp** and **debug ip inspect detailed** commands:

```

Router# debug ip inspect icmp
Router# debug ip inspect detailed

lw6d:CBAC sis 81073F0C SIS_CLOSED
lw6d:CBAC Pak 80D2E9EC IP:s=192.168.133.3 (Ethernet1), d=0.0.0.0 (Ethernet0), len 98,
proto=1
lw6d:CBAC ICMP:sis 81073F0C pak 80D2E9EC SIS_CLOSED ICMP packet (192.168.133.3:0) =>
(0.0.0.0:0) datalen 56
lw6d:CBAC ICMP:start session from 192.168.133.3
lw6d:CBAC sis 81073F0C --> SIS_OPENING (192.168.133.3:0) (0.0.0.0:0)
lw6d:CBAC sis 81073F0C L4 inspect result:PASS packet 80D2E9EC (192.168.133.3:0)
(0.0.0.0:0) bytes 56 icmp
lw6d:CBAC sis 81073F0C SIS_OPENING
lw6d:CBAC Pak 80E72BFC IP:s=0.0.0.0 (Ethernet0), d=192.168.133.3 (Ethernet1), len 98,
proto=1
lw6d:CBAC ICMP:sis 81073F0C pak 80E72BFC SIS_OPENING ICMP packet (192.168.133.3:0) <=
(0.0.0.0:0) datalen 56
lw6d:CBAC sis 81073F0C --> SIS_OPEN (192.168.133.3:0) (0.0.0.0:0)
lw6d:CBAC sis 81073F0C L4 inspect result:PASS packet 80E72BFC (0.0.0.0:0)
(192.168.133.3:0) bytes 56 icmp
lw6d:CBAC* sis 81073F0C SIS_OPEN
lw6d:CBAC* Pak 80D2F2C8 IP:s=192.168.133.3 (Ethernet1), d=0.0.0.0 (Ethernet0), len 98,
proto=1
lw6d:CBAC* ICMP:sis 81073F0C pak 80D2F2C8 SIS_OPEN ICMP packet (192.168.133.3:0) =>
(0.0.0.0:0) datalen 56
lw6d:CBAC* sis 81073F0C --> SIS_OPEN (192.168.133.3:0) (0.0.0.0:0)
lw6d:CBAC* sis 81073F0C L4 inspect result:PASS packet 80D2F2C8 (192.168.133.3:0)
(0.0.0.0:0) bytes 56 icmp
lw6d:CBAC* sis 81073F0C SIS_OPEN
lw6d:CBAC* Pak 80E737CC IP:s=0.0.0.0 (Ethernet0), d=192.168.133.3 (Ethernet1), len 98,
proto=1
lw6d:CBAC* ICMP:sis 81073F0C pak 80E737CC SIS_OPEN ICMP packet (192.168.133.3:0) <=
(0.0.0.0:0) datalen 56
lw6d:CBAC* sis 81073F0C --> SIS_OPEN (192.168.133.3:0) (0.0.0.0:0)
lw6d:CBAC* sis 81073F0C L4 inspect result:PASS packet 80E737CC (0.0.0.0:0)
(192.168.133.3:0) bytes 56 icmp
lw6d:CBAC* sis 81073F0C SIS_OPEN
lw6d:CBAC* Pak 80F554F0 IP:s=192.168.133.3 (Ethernet1), d=0.0.0.0 (Ethernet0), len 98,
proto=1

```

```
1w6d:CBAC* ICMP:sis 81073F0C pak 80F554F0 SIS_OPEN ICMP packet (192.168.133.3:0) =>
(0.0.0.0:0) datalen 56
1w6d:CBAC* sis 81073F0C --> SIS_OPEN (192.168.133.3:0) (0.0.0.0:0)
1w6d:CBAC* sis 81073F0C L4 inspect result:PASS packet 80F554F0 (192.168.133.3:0)
(0.0.0.0:0) bytes 56 icmp
1w6d:CBAC* sis 81073F0C SIS_OPEN
1w6d:CBAC* Pak 80E73AC0 IP:s=0.0.0.0 (Ethernet0), d=192.168.133.3 (Ethernet1), len 98,
proto=1
1w6d:CBAC* ICMP:sis 81073F0C pak 80E73AC0 SIS_OPEN ICMP packet (192.168.133.3:0) <=
(0.0.0.0:0) datalen 56
1w6d:CBAC* sis 81073F0C --> SIS_OPEN (192.168.133.3:0) (0.0.0.0:0)
1w6d:CBAC* sis 81073F0C L4 inspect result:PASS packet 80E73AC0 (0.0.0.0:0) (192.168.133.3:0)
bytes 56 icmp
```

debug ip inspect L2-transparent

To enable debugging messages for transparent firewall events, use the **debug ip inspect L2-transparent** command in privileged EXEC mode. To disable debugging messages, use the **no** form of this command.

debug ip inspect L2-transparent {packet | dhcp-passthrough}

no debug ip inspect L2-transparent {packet | dhcp-passthrough}

Syntax Description	packet	Displays messages for all debug packets that are inspected by the transparent firewall.
		Note Only IP packets (TCP, User Datagram Protocol [UDP], and Internet Control Management Protocol [ICMP]) are subjected to inspection by the transparent firewall.
	dhcp-passthrough	Displays debug messages only for DHCP pass-through traffic that the transparent firewall forwards across the bridge.
		To allow a transparent firewall to forward DHCP pass-through traffic, use the ip inspect L2-transparent dhcp-passthrough command.

Command Modes Privileged EXEC

Command History	Release	Modification
	12.3(7)T	This command was introduced.

Usage Guidelines The **debug ip inspect L2-transparent** command can be used to help verify and troubleshoot transparent firewall-related configurations, such as a Telnet connection from the client to the server with inspection configured.

Examples The following example shows how the transparent firewall debug command works in a basic transparent firewall configuration. (Note that each debug message is preceded by an asterisk (*).)

```
! Enable debug commands.
Router# debug ip inspect L2-transparent packet
INSPECT L2 firewall debugging is on

Router# debug ip inspect object-creation
INSPECT Object Creations debugging is on

Router# debug ip inspect object-deletion
INSPECT Object Deletions debugging is on
! Start the transparent firewall configuration process

Router# config terminal
Enter configuration commands, one per line. End with CNTL/Z.
! Configure bridging

Router(config)# bridge 1 protocol ieee
```

```

Router(config)# bridge irb
Router(config)# bridge 1 route ip
Router(config)# interface bv11

*Mar  1 00:06:42.511:%LINK-3-UPDOWN:Interface BV11, changed state to down.

Router(config-if)# ip address 209.165.200.225 255.255.255.254
! Configure inspection

Router(config)# ip inspect name test tcp
! Following debugs show the memory allocated for CBAC rules.
*Mar  1 00:07:21.127:CBAC OBJ_CREATE:create irc 817F04F0 (test)
*Mar  1 00:07:21.127:CBAC OBJ_CREATE:create irt 818AED20 Protocol:tcp Inactivity time:0
test

Router(config)# ip inspect name test icmp
Router(config)#
*Mar  1 00:07:39.211:CBAC OBJ_CREATE:create irt 818AEDCC Protocol:icmp Inactivity time:0
! Configure Bridging on ethernet0 interface

Router(config)# interface ethernet0

Router(config-if)# bridge-group 1
*Mar  1 00:07:49.071:%LINK-3-UPDOWN:Interface BV11, changed state to up
*Mar  1 00:07:50.071:%LINEPROTO-5-UPDOWN:Line protocol on Interface BV11, changed state to
up
! Configure inspection on ethernet0 interface

Router(config-if)# ip inspect test in
Router(config-if)#
*Mar  1 00:07:57.543:CBAC OBJ_CREATE:create idbsb 8189CBFC (Ethernet0)

! Incremented the number of bridging interfaces configured for inspection */
*Mar  1 00:07:57.543:L2FW:Incrementing L2FW i/f count

Router(config-if)# interface ethernet1
! Configure bridging and ACL on interface ethernet1

Router(config-if)# bridge-group 1
Router(config-if)# ip access-group 101 in
*Mar  1 00:08:26.711:%LINEPROTO-5-UPDOWN:Line protocol on Interface Ethernet1, changed
state to up

Router(config-if)# end

```

Related Commands

Command	Description
ip inspect L2-transparent dhcp-passthrough	Allows a transparent firewall to forward DHCP pass-through traffic.

debug ip ips

To enable debugging messages for Cisco IOS Intrusion Prevention System (IPS), use the **debug ip ips** command in privileged EXEC mode. To disable debugging messages, use the **no** form of this command.

debug ip ips [*engine*] [**detailed**]

no debug ip ips [*engine*] [**detailed**]

Syntax Description	
<i>engine</i>	(Optional) Displays debugging messages only for a specific signature engine.
detailed	(Optional) Displays detailed debugging messages for the specified signature engine or for all IPS actions.

Command Modes	
	Privileged EXEC

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

Examples The following example shows how to enable debugging messages for the Cisco IOS IPS:

```
Router# debug ip ips
```

debug ip mbgp dampening

To log route flap dampening activity related to multiprotocol Border Gateway Protocol (BGP), use the **debug ip mbgp dampening** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip mbgp dampening [*access-list-number*]

no debug ip mbgp dampening [*access-list-number*]

Syntax Description	<i>access-list-number</i>	(Optional) The number of an access list in the range from 1 to 99. If an access list number is specified, debugging occurs only for the routes permitted by the access list.
---------------------------	---------------------------	--

Defaults	Logging for route flap dampening activity is not enabled.
-----------------	---

Command Modes	Privileged EXEC
----------------------	-----------------

Command History	Release	Modification
	11.1(20)CC	This command was introduced.

Examples	The following is sample output from the debug ip mbgp dampening command:
-----------------	---

```
Router# debug ip mbgp dampening

BGP: charge penalty for 173.19.0.0/16 path 49 with halflife-time 15 reuse/suppress
750/2000
BGP: flapped 1 times since 00:00:00. New penalty is 1000
BGP: charge penalty for 173.19.0.0/16 path 19 49 with halflife-time 15 reuse/suppress
750/2000
BGP: flapped 1 times since 00:00:00. New penalty is 1000
```

debug ip mbgp updates

To log multiprotocol Border Gateway Protocol (BGP)-related information passed in BGP update messages, use the **debug ip mbgp updates** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip mbgp updates

no debug ip mbgp updates

Syntax Description This command has no arguments or keywords.

Defaults Logging for multiprotocol BGP-related information in BGP update messages is not enabled.

Command Modes Privileged EXEC

Command History	Release	Modification
	11.1(20)CC	This command was introduced.

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

```
Router# debug ip mbgp updates

BGP: NEXT_HOP part 1 net 200.10.200.0/24, neigh 171.69.233.49, next 171.69.233.34
BGP: 171.69.233.49 send UPDATE 200.10.200.0/24, next 171.69.233.34, metric 0, path 33 34
19 49 109 65000 297 3561 6503
BGP: NEXT_HOP part 1 net 200.10.202.0/24, neigh 171.69.233.49, next 171.69.233.34
BGP: 171.69.233.49 send UPDATE 200.10.202.0/24, next 171.69.233.34, metric 0, path 33 34
19 49 109 65000 297 1239 1800 3597
BGP: NEXT_HOP part 1 net 200.10.228.0/22, neigh 171.69.233.49, next 171.69.233.34
BGP: 171.69.233.49 rcv UPDATE about 222.2.2.0/24, next hop 171.69.233.49, path 49 109
metric 0
BGP: 171.69.233.49 rcv UPDATE about 131.103.0.0/16, next hop 171.69.233.49, path 49 109
metric 0
BGP: 171.69.233.49 rcv UPDATE about 206.205.242.0/24, next hop 171.69.233.49, path 49 109
metric 0
BGP: 171.69.233.49 rcv UPDATE about 1.0.0.0/8, next hop 171.69.233.49, path 49 19 metric 0
BGP: 171.69.233.49 rcv UPDATE about 198.1.2.0/24, next hop 171.69.233.49, path 49 19
metric 0
BGP: 171.69.233.49 rcv UPDATE about 171.69.0.0/16, next hop 171.69.233.49, path 49 metric
0
BGP: 171.69.233.49 rcv UPDATE about 172.19.0.0/16, next hop 171.69.233.49, path 49 metric
0
BGP: nettable_walker 172.19.0.0/255.255.0.0 calling revise_route
BGP: revise route installing 172.19.0.0/255.255.0.0 -> 171.69.233.49
BGP: 171.69.233.19 computing updates, neighbor version 267099, table version 267100,
starting at 0.0.0.0
BGP: NEXT_HOP part 1 net 172.19.0.0/16, neigh 171.69.233.19, next 171.69.233.49
BGP: 171.69.233.19 send UPDATE 172.19.0.0/16, next 171.69.233.49, metric 0, path 33 49
BGP: 1 updates (average = 46, maximum = 46)
```

```
BGP: 171.69.233.19 updates replicated for neighbors : 171.69.233.34, 171.69.233.49,  
171.69.233.56  
BGP: 171.69.233.19 1 updates enqueued (average=46, maximum=46)  
BGP: 171.69.233.19 update run completed, ran for 0ms, neighbor version 267099, start  
version 267100, throttled to 267100, check point net 0.0.0.0
```

debug ip mcache

To display IP multicast fast-switching events, use the **debug ip mcache** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

```
debug ip mcache [vrf vrf-name] [hostname | group-address]
```

```
no debug ip mcache [vrf vrf-name] [hostname | group-address]
```

Syntax Description		
vrf	(Optional)	Supports the Multicast Virtual Private Network (VPN) routing and forwarding (VRF) instance.
<i>vrf-name</i>	(Optional)	Name assigned to the VRF.
<i>hostname</i>	(Optional)	The host name.
<i>group-address</i>	(Optional)	The group address.

Defaults No default behavior or values

Command Modes Privileged EXEC

Command History	Release	Modification
	11.0	This command was introduced.
	12.0(23)S	The vrf keyword and <i>vrf-name</i> argument were added.
	12.2(13)T	The vrf keyword and <i>vrf-name</i> argument were added in Cisco IOS Release 12.2T.

Usage Guidelines Use this command when multicast fast switching appears not to be functioning.

Examples The following is sample output from the **debug ip mcache** command when an IP multicast route is cleared:

```
Router# debug ip mcache

IP multicast fast-switching debugging is on

Router# clear ip mroute *

MRC: Build MAC header for (172.31.60.185/32, 224.2.231.173), Ethernet0
MRC: Fast-switch flag for (172.31.60.185/32, 224.2.231.173), off -> on, caller
ip_mroute_replicate-1
MRC: Build MAC header for (172.31.191.10/32, 224.2.127.255), Ethernet0
MRC: Build MAC header for (172.31.60.152/32, 224.2.231.173), Ethernet0
```

Table 123 describes the significant fields shown in the display.

Table 123 *debug ip mcache Field Descriptions*

Field	Description
MRC	Multicast route cache.
Fast-switch flag	Route is fast switched.
(172.31.60.185/32)	Host route with 32 bits of mask.
off -> on	State has changed.
caller ...	The code function that activated the state change.

Related Commands

Command	Description
debug ip dvmrp	Displays information on DVMRP packets received and sent.
debug ip igmp	Displays IGMP packets received and sent, and IGMP-host related events.
debug ip igrp transactions	Displays transaction information on IGRP routing transactions.
debug ip mrm	Displays MRM control packet activity.
debug ip sd	Displays all SD announcements received.

debug ip mds ipc

To debug multicast distributed switching (MDS) interprocessor communication, that is, synchronization between the Multicast Forwarding Information Base (MFIB) on the line card and the multicast routing table in the Route Processor (RP), use the **debug ip mds ipc** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

```
debug ip mds ipc {event | packet}
```

```
no debug ip mds ipc {event | packet}
```

Syntax Description	event	Displays MDS events when there is a problem.
	packet	Displays MDS packets.

Command Modes Privileged EXEC

Usage Guidelines Use this command on the line card or RP.

Examples The following is sample output from the **debug ip mds ipc packet** command:

```
Router# debug ip mds ipc packet
```

```
MDFS ipc packet debugging is on
```

```
Router#
```

```
MDFS: LC sending statistics message to RP with code 0 of size 36
MDFS: LC sending statistics message to RP with code 1 of size 680
MDFS: LC sending statistics message to RP with code 2 of size 200
MDFS: LC sending statistics message to RP with code 3 of size 152
MDFS: LC sending window message to RP with code 36261 of size 8
MDFS: LC received IPC packet of size 60 sequence 36212
```

The following is sample output from the **debug ip mds ipc event** command:

```
Router# debug ip mds ipc event
```

```
MDFS: LC received invalid sequence 21 while expecting 20
```

debug ip mds mevent

To debug Multicast Forwarding Information Base (MFIB) route creation, route updates, and so on, use the **debug ip mds mevent** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip mds mevent

no debug ip mds mevent

Syntax Description This command has no arguments or keywords.

Command Modes Privileged EXEC

Usage Guidelines Use this command on the line card.

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

```
Router# debug ip mds mevent

MDFS mroute event debugging is on
Router#clear ip mdfs for *
Router#
MDFS: Create (*, 239.255.255.255)
MDFS: Create (192.168.1.1/32, 239.255.255.255), RPF POS2/0/0
MDFS: Add OIF for mroute (192.168.1.1/239.255.255.255) on Fddi0/0/0
MDFS: Create (*, 224.2.127.254)
MDFS: Create (192.168.1.1/32, 224.2.127.254), RPF POS2/0/0
MDFS: Add OIF for mroute (192.168.1.1/224.2.127.254) on Fddi0/0/0
MDFS: Create (128.9.160.67/32, 224.2.127.254), RPF POS2/0/0
```

debug ip mds mpacket

To debug multicast distributed switching (MDS) events such as packet drops, interface drops, and switching failures, use the **debug ip mds mpacket** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip mds mpacket

no debug ip mds mpacket

Syntax Description This command has no arguments or keywords.

Command Modes Privileged EXEC

Usage Guidelines Use this command on the line card.

debug ip mds process

To debug line card process level events, use the **debug ip mds process** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip mds process

no debug ip mds process

Syntax Description This command has no arguments or keywords.

Command Modes Privileged EXEC

Usage Guidelines Use this command on the line card or Route Processor (RP).

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

```
Router# debug ip mds process

MDFS process debugging is on
Mar 19 16:15:47.448: MDFS: RP queueing mdb message for (210.115.194.5, 224.2.127.254) to
all linecards
Mar 19 16:15:47.448: MDFS: RP queueing midb message for (210.115.194.5, 224.2.127.254) to
all linecards
Mar 19 16:15:47.628: MDFS: RP servicing low queue for LC in slot 0
Mar 19 16:15:47.628: MDFS: RP servicing low queue for LC in slot 2
Mar 19 16:15:48.229: MDFS: RP queueing mdb message for (171.68.224.10, 224.2.127.254) to
all linecards
Mar 19 16:15:48.229: MDFS: RP queueing mdb message for (171.68.224.10, 224.2.127.254) to
all linecards
Mar 19 16:15:48.229: MDFS: RP queueing mdb message for (171.69.67.106, 224.2.127.254) to
all linecards
Mar 19 16:15:48.229: MDFS: RP queueing mdb message for (171.69.67.106, 224.2.127.254) to
all linecards
Mar 19 16:15:48.229: MDFS: RP queueing mdb message for (206.14.154.181, 224.2.127.254) to
all linecards
Mar 19 16:15:48.229: MDFS: RP queueing mdb message for (206.14.154.181, 224.2.127.254) to
all linecards
Mar 19 16:15:48.233: MDFS: RP queueing mdb message for (210.115.194.5, 224.2.127.254) to
all linecards
```

debug ip mhbeat

To monitor the action of the heartbeat trap, use the **debug ip mhbeat** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip mhbeat

no debug ip mhbeat

Syntax Description This command has no arguments or keywords.

Defaults Debugging is not enabled.

Command Modes Privileged EXEC

Command History	Release	Modification
	12.1(2)XH	This command was introduced.

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

```
Router# debug ip mhbeat

IP multicast heartbeat debugging is on

Router debug snmp packets

SNMP packet debugging is on

!
Router(config)# ip multicast heartbeat intervals-of 10

Dec 23 13:34:21.132: MHBEAT: ip multicast-heartbeat group 224.0.1.53 port 0
      source 0.0.0.0 0.0.0.0 at-least 3 in 5 intervals-of 10 secondsd
Router#
Dec 23 13:34:23: %SYS-5-CONFIG_I: Configured from console by console
Dec 23 13:34:31.136: MHBEAT: timer ticked, t=1,i=1,c=0
Dec 23 13:34:41.136: MHBEAT: timer ticked, t=2,i=2,c=0
Dec 23 13:34:51.136: MHBEAT: timer ticked, t=3,i=3,c=0
Dec 23 13:35:01.136: MHBEAT: timer ticked, t=4,i=4,c=0
Dec 23 13:35:11.136: MHBEAT: timer ticked, t=5,i=0,c=0
Dec 23 13:35:21.135: Send SNMP Trap for missing heartbeat
Dec 23 13:35:21.135: SNMP: Queuing packet to 171.69.55.12
Dec 23 13:35:21.135: SNMP: V1 Trap, ent ciscoExperiment.2.3.1, addr 4.4.4.4, gentrap 6,
spectrap 1
  ciscoIpMRouteHeartBeat.1.0 = 224.0.1.53
  ciscoIpMRouteHeartBeat.2.0 = 0.0.0.0
  ciscoIpMRouteHeartBeat.3.0 = 10
  ciscoIpMRouteHeartBeat.4.0 = 5
  ciscoIpMRouteHeartBeat.5.0 = 0
  ciscoIpMRouteHeartBeat.6.0 = 3
```

Related Commands

Command	Description
ip multicast heartbeat	Monitors the health of multicast delivery, and alerts when the delivery fails to meet certain parameters.

debug ip mobile

To display IP mobility activities, use the **debug ip mobile** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip mobile [**advertise** | **host** [*access-list-number*] | **local-area** | **redundancy**]

no debug ip mobile

Syntax Description	
advertise	(Optional) Advertisement information.
host	(Optional) The mobile node host.
<i>access-list-number</i>	(Optional) The number of an IP access list.
local-area	(Optional) The local area.
redundancy	(Optional) Redundancy activities.

Command Modes Privileged EXEC

Command History	Release	Modification
	12.0(1)T	This command was introduced.
	12.0(2)T	The standby keyword was added.
	12.2(13)T	This command was enhanced to display information about foreign agent reverse tunnels and the mobile networks attached to the mobile router.
	12.3(8)T	This command was enhanced to display information about UDP tunneling. The standby keyword was replaced with the redundancy keyword. This command replaces the debug ip mobile advertise , debug ip mobile host , and debug ip mobile redundancy commands.

Usage Guidelines Use the **debug ip mobile standby** command to troubleshoot redundancy problems.

No per-user debugging output is shown for mobile nodes using the network access identifier (NAI) for the **debug ip mobile host** command. Debugging of specific mobile nodes using an IP address is possible through the access list.

Examples The following is sample output from the **debug ip mobile** command when foreign agent reverse tunneling is enabled:

```
Router# debug ip mobile
```

```
MobileIP:MN 10.0.0.10 deleted from ReverseTunnelTable of Ethernet2/1(Entries 0)
```

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

```
Router# debug ip mobile advertise
```

```
MobileIP: Agent advertisement sent out Ethernet1/2: type=16, len=10, seq=1,
lifetime=36000,
flags=0x1400(rbhFmGv-rsv-),
Care-of address: 10.0.0.31
Prefix Length ext: len=1 (8 )
FA Challenge value:769C808D
```

Table 124 describes the significant fields shown in the display.

Table 124 *debug ip mobile advertise Field Descriptions*

Field	Description
type	Type of advertisement.
len	Length of extension (in bytes).
seq	Sequence number of this advertisement.
lifetime	Lifetime (in seconds).
flags	Capital letters represent bits that are set; lowercase letters represent unset bits.
Care-of address	IP address.
Prefix Length ext	Number of prefix lengths advertised. This is the bits in the mask of the interface that sends this advertisement. Used for roaming detection.
FA Challenge value	Foreign agent challenge value (randomly generated by the foreign agent).

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

```
Router# debug ip mobile host
```

```
MobileIP: HA received registration for MN 10.0.0.6 on interface Ethernet1 using COA
10.0.0.31 HA 10.0.0.5 lifetime 30000 options sbdmgvT
MobileIP: Authenticated FA 10.0.0.31 using SPI 110 (MN 10.0.0.6)
MobileIP: Authenticated MN 10.0.0.6 using SPI 300

MobileIP: HA accepts registration from MN 10.0.0.6
MobileIP: Mobility binding for MN 10.0.0.6 updated
MobileIP: Roam timer started for MN 10.0.0.6, lifetime 30000
MobileIP: MH auth ext added (SPI 300) in reply to MN 10.0.0.6
MobileIP: HF auth ext added (SPI 220) in reply to MN 10.0.0.6

MobileIP: HA sent reply to MN 10.0.0.6
```

The following is sample output from the **debug ip mobile standby** command. In this example, the active home agent receives a registration request from mobile node 10.0.0.2 and sends a binding update to peer home agent 1.0.0.2:

```
MobileIP:MN 10.0.0.2 - sent BindUpd to HA 1.0.0.2 HAA 10.0.0.1
MobileIP:HA standby maint started - cnt 1
MobileIP:MN 10.0.0.2 - sent BindUpd id 3780410816 cnt 0 elapsed 0
adjust -0 to HA 1.0.0.2 in grp 1.0.0.10 HAA 10.0.0.1
```

In the following example, the standby home agent receives a binding update for mobile node 10.0.0.2 sent by the active home agent:

```
MobileIP:MN 10.0.0.2 - HA rcv BindUpd from 1.0.0.3 HAA 10.0.0.1
```

UDP Tunneling for NAT Traversal

The following sample output displays the registration, authentication, and establishment of UDP tunneling of a mobile node (MN) with a foreign agent (FA):

```
Dec 31 12:34:25.707: UDP: rcvd src=10.10.10.10(434),dst=10.30.30.1(434), length=54
Dec 31 12:34:25.707: MobileIP: ParseRegExt type MHAЕ(32) addr 2000FEЕC end 2000FF02
Dec 31 12:34:25.707: MobileIP: ParseRegExt skipping 10 to next
Dec 31 12:34:25.707: MobileIP: FA rcv registration for MN 10.10.10.10 on Ethernet2/2 using
COA 10.30.30.1 HA 10.10.10.100 lifetime 65535 options sbdmg-T-identification
C1BC0D4FB01AC0D8
Dec 31 12:34:25.707: MobileIP: Ethernet2/2 glean 10.10.10.10 accepted
Dec 31 12:34:25.707: MobileIP: Registration request byte count = 74
Dec 31 12:34:25.707: MobileIP: FA queued MN 10.10.10.10 in register table
Dec 31 12:34:25.707: MobileIP: Visitor registration timer started for MN 10.10.10.10,
lifetime 120
Dec 31 12:34:25.707: MobileIP: Adding UDP Tunnel req extension
Dec 31 12:34:25.707: MobileIP: Authentication algorithm MD5 and 16 byte key
Dec 31 12:34:25.707: MobileIP: MN 10.10.10.10 FHAЕ added to HA 10.10.10.100 using SPI 1000
Dec 31 12:34:25.707: MobileIP: FA forwarded registration for MN 10.10.10.10 to HA
10.10.10.100
Dec 31 12:34:25.715: UDP: rcvd src=10.10.10.100(434), dst=10.30.30.1(434), length=94
Dec 31 12:34:25.715: MobileIP: ParseRegExt type NVSE(134) addr 20010B28 end 20010B6A
Dec 31 12:34:25.715: MobileIP: ParseRegExt type MN-config NVSE(14) subtype 1 (MN prefix
length) prefix length (24)
Dec 31 12:34:25.715: MobileIP: ParseRegExt skipping 12 to next
Dec 31 12:34:25.715: MobileIP: ParseRegExt type MHAЕ(32) addr 20010B36 end 20010B6A
Dec 31 12:34:25.715: MobileIP: ParseRegExt skipping 10 to next
Dec 31 12:34:25.715: MobileIP: ParseRegExt type UDPTUNREPE(44) addr 20010B4C end 20010B6A
Dec 31 12:34:25.715: Parsing UDP Tunnel Reply Extension - length 6
Dec 31 12:34:25.715: MobileIP: ParseRegExt skipping 6 to next
Dec 31 12:34:25.715: MobileIP: ParseRegExt type FHAЕ(34) addr 20010B54 end 20010B6A
Dec 31 12:34:25.715: MobileIP: ParseRegExt skipping 20 to next
Dec 31 12:34:25.715: MobileIP: FA rcv accept (0) reply for MN 10.10.10.10 on Ethernet2/3
using HA 10.10.10.100 lifetime 65535
Dec 31 12:34:25.719: MobileIP: Authenticating HA 10.10.10.100 using SPI 1000
Dec 31 12:34:25.719: MobileIP: Authentication algorithm MD5 and 16 byte key
Dec 31 12:34:25.719: MobileIP: Authenticated HA 10.10.10.100 using SPI 1000 and 16 byte
key
Dec 31 12:34:25.719: MobileIP: HA accepts UDP Tunneling
Dec 31 12:34:25.719: MobileIP: Update visitor table for MN 10.10.10.10
Dec 31 12:34:25.719: MobileIP: Enabling UDP Tunneling
Dec 31 12:34:25.719: MobileIP: Tunnel0 (MIPUDP/IP) created with src 10.30.30.1 dst
10.10.10.100
Dec 31 12:34:25.719: MobileIP: Setting up UDP Keep-Alive Timer for tunnel 10.30.30.1:0 -
10.10.10.100:0 with keep-alive 30
Dec 31 12:34:25.719: MobileIP: Starting the tunnel keep-alive timer
Dec 31 12:34:25.719: MobileIP: ARP entry for MN 10.10.10.10 using 10.10.10.10 inserted on
Ethernet2/2
Dec 31 12:34:25.719: MobileIP: FA route add 10.10.10.10 successful. Code = 0
Dec 31 12:34:25.719: MobileIP: MN 10.10.10.10 added to ReverseTunnelTable of Ethernet2/2
(Entries 1)
Dec 31 12:34:25.719: MobileIP: FA dequeued MN 10.10.10.10 from register table
Dec 31 12:34:25.719: MobileIP: MN 10.10.10.10 using 10.10.10.10 visiting on Ethernet2/2
Dec 31 12:34:25.719: MobileIP: Reply in for MN 10.10.10.10 using 10.10.10.10, accepted
Dec 31 12:34:25.719: MobileIP: registration reply byte count = 84
Dec 31 12:34:25.719: MobileIP: FA forwarding reply to MN 10.10.10.10 (10.10.10.10 mac
0060.70ca.f021)
Dec 31 12:34:26.095: MobileIP: agent advertisement byte count = 48
```

```

Dec 31 12:34:26.095: MobileIP: Agent advertisement sent out Ethernet2/2: type=16, len=10,
  seq=55, lifetime=65535, flags=0x1580(rbhFmG-TU),
Dec 31 12:34:26.095: Care-of address: 10.30.30.1
Dec 31 12:34:26.719: MobileIP: swif coming up Tunnel0
!
Dec 31 12:34:35.719: UDP: sent src=10.30.30.1(434), dst=10.10.10.100(434)
Dec 31 12:34:35.719: UDP: rcvd src=10.10.10.100(434), dst=10.30.30.1(434), length=32d0

```

The following sample output shows the registration, authentication, and establishment of UDP tunneling of an MN with a home agent (HA):

```

Dec 31 12:34:26.167: MobileIP: ParseRegExt skipping 20 to next
Dec 31 12:34:26.167: MobileIP: ParseRegExt type UDPTUNREQE(144) addr 2001E762 end 2001E780
Dec 31 12:34:26.167: MobileIP: Parsing UDP Tunnel Request Extension - length 6
Dec 31 12:34:26.167: MobileIP: ParseRegExt skipping 6 to next
Dec 31 12:34:26.167: MobileIP: ParseRegExt type FHAE(34) addr 2001E76A end 2001E780
Dec 31 12:34:26.167: MobileIP: ParseRegExt skipping 20 to next
Dec 31 12:34:26.167: MobileIP: HA 167 rcv registration for MN 10.10.10.10 on Ethernet2/1
  using HomeAddr 10.10.10.10 COA 10.30.30.1 HA 10.10.10.100 lifetime 65535 options
  sbdmg-T-identification C1BC0D4FB01AC0D8
Dec 31 12:34:26.167: MobileIP: NAT detected SRC:10.10.10.50 COA: 10.30.30.1
Dec 31 12:34:26.167: MobileIP: UDP Tunnel Request accepted 10.10.10.50:434
Dec 31 12:34:26.167: MobileIP: Authenticating FA 10.30.30.1 using SPI 1000
Dec 31 12:34:26.167: MobileIP: Authentication algorithm MD5 and 16 byte key
Dec 31 12:34:26.167: MobileIP: Authentication algorithm MD5 and truncated key
Dec 31 12:34:26.167: MobileIP: Authentication algorithm MD5 and 16 byte key
Dec 31 12:34:26.167: MobileIP: Authenticated FA 10.30.30.1 using SPI 1000 and 16 byte key
Dec 31 12:34:26.167: MobileIP: Authenticating MN 10.10.10.10 using SPI 1000
Dec 31 12:34:26.167: MobileIP: Authentication algorithm MD5 and 16 byte key
Dec 31 12:34:26.167: MobileIP: Authentication algorithm MD5 and truncated key
Dec 31 12:34:26.167: MobileIP: Authentication algorithm MD5 and 16 byte key
Dec 31 12:34:26.167: MobileIP: Authenticated MN 10.10.10.10 using SPI 1000 and 16 byte key
Dec 31 12:34:26.167: MobileIP: Mobility binding for MN 10.10.10.10 created
Dec 31 12:34:26.167: MobileIP: NAT detected for MN 10.10.10.10. Terminating tunnel on
  10.10.10.50
Dec 31 12:34:26.167: MobileIP: Tunnel0 (MIPUDP/IP) created with src 10.10.10.100 dst
  10.10.10.50
Dec 31 12:34:26.167: MobileIP: Setting up UDP Keep-Alive Timer for tunnel 10.10.10.100:0 -
  10.10.10.50:0 with keep-alive 30
Dec 31 12:34:26.167: MobileIP: Starting the tunnel keep-alive timer
Dec 31 12:34:26.167: MobileIP: MN 10.10.10.10 Insert route for 10.10.10.10/255.255.255.255
  via gateway 10.10.10.50 on Tunnel0
Dec 31 12:34:26.167: MobileIP: MN 10.10.10.10 is now roaming
Dec 31 12:34:26.171: MobileIP: Gratuitous ARPs sent for MN 10.10.10.10 MAC 0002.fca5.bc39
Dec 31 12:34:26.171: MobileIP: Mask for address is 24
Dec 31 12:34:26.171: MobileIP: HA accepts registration from MN 10.10.10.10
Dec 31 12:34:26.171: MobileIP: Dynamic and Static Network Extension Length 0 - 0
Dec 31 12:34:26.171: MobileIP: Composed mobile network extension length:0
Dec 31 12:34:26.171: MobileIP: Added prefix length vse in reply
Dec 31 12:34:26.171: MobileIP: Authentication algorithm MD5 and 16 byte key
Dec 31 12:34:26.171: MobileIP: MN 10.10.10.10 MHAE added to MN 10.10.10.10 using SPI 1000
Dec 31 12:34:26.171: MobileIP: Authentication algorithm MD5 and 16 byte key
Dec 31 12:34:26.171: MobileIP: MN 10.10.10.10 FHAE added to FA 10.10.10.50 using SPI 1000
Dec 31 12:34:26.171: MobileIP: MN 10.10.10.10 - HA sent reply to 10.10.10.50
Dec 31 12:34:26.171: MobileIP: Authentication algorithm MD5 and 16 byte key
Dec 31 12:34:26.171: MobileIP: MN 10.10.10.10 HHAE added to HA 10.10.10.3 using SPI 1000
Dec 31 12:34:26.175: MobileIP: ParseRegExt type CVSE(38) addr 2000128C end 200012AE
Dec 31 12:34:26.175: MobileIP: ParseRegExt type HA red. version CVSE(6)
Dec 31 12:34:26.175: MobileIP: ParseRegExt skipping 8 to next
Dec 31 12:34:26.175: MobileIP: ParseRegExt type HHAE(35) addr 20001298 end 200012AE
Dec 31 12:34:26.175: MobileIP: ParseRegExt skipping 20 to next
Dec 31 12:34:26.175: MobileIP: Authenticating HA 10.10.10.3 using SPI 1000
Dec 31 12:34:26.175: MobileIP: Authentication algorithm MD5 and 16 byte key
Dec 31 12:34:26.175: MobileIP: Authentication algorithm MD5 and truncated key

```

```
Dec 31 12:34:26.175: MobileIP: Authentication algorithm MD5 and 16 byte key
Dec 31 12:34:26.175: MobileIP: Authenticated HA 10.10.10.3 using SPI 1000 and 16 byte key
Dec 31 12:34:27.167: MobileIP: swif coming up Tunnel0d0
```

Related Commands

Command	Description
ip mobile foreign-agent nat traversal	Enables NAT UDP traversal support for MIP FAs.
ip mobile home-agent nat traversal	Enables NAT UDP traversal support for MIP HAs.
show ip mobile binding	Displays the mobility binding table.
show ip mobile globals	Displays global information about MIP HAs, FAs, and MNs.
show ip mobile tunnel	Displays information about UDP tunneling.
show ip mobile visitor	Displays the table that contains a visitor list of FAs.

debug ip mobile advertise



Note

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

To display advertisement information, use the **debug ip mobile advertise** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip mobile advertise

no debug ip mobile advertise

Syntax Description

This command has no arguments or keywords.

Defaults

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(1)T	This command was introduced.
12.3(4)T	This command was integrated into Cisco IOS Release 12.3(4)T.
12.3(8)T	This command was replaced by the debug ip mobile command.

Examples

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

```
Router# debug ip mobile advertise
```

```
MobileIP: Agent advertisement sent out Ethernet1/2: type=16, len=10, seq=1,
lifetime=36000,
flags=0x1400 (rbhFmGv-rsv-),
Care-of address: 68.0.0.31
Prefix Length ext: len=1 (8 )
```

[Table 125](#) describes significant fields shown in the display.

Table 125 *debug ip mobile advertise Field Descriptions*

Field	Description
type	Type of advertisement.
len	Length of extension in bytes.
seq	Sequence number of this advertisement.
lifetime	Lifetime in seconds.

Table 125 *debug ip mobile advertise Field Descriptions (continued)*

Field	Description
flags	Capital letters represent bits that are set, lowercase letters represent unset bits.
Care-of address	IP address.
Prefix Length ext	Number of prefix lengths advertised. This number is the bits in the mask of the interface sending this advertisement. Used for roaming detection.

debug ip mobile host



Note

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

To display IP mobility events, use the **debug ip mobile host** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip mobile host [*acl*]

no debug ip mobile host

Syntax Description

acl (Optional) Access list.

Defaults

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(1)T	This command was introduced.
12.3(4)T	This command was integrated into Cisco IOS Release 12.3(4)T.
12.3(8)T	This command was replaced by the debug ip mobile command.

Examples

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

```
Router# debug ip mobile host
```

```
MobileIP: HA received registration for MN 20.0.0.6 on interface Ethernet1 using COA
68.0.0.31 HA 66.0.0.5 lifetime 30000 options sbdmgvT
```

```
MobileIP: Authenticated FA 68.0.0.31 using SPI 110 (MN 20.0.0.6)
```

```
MobileIP: Authenticated MN 20.0.0.6 using SPI 300
```

```
MobileIP: HA accepts registration from MN 20.0.0.6
```

```
MobileIP: Mobility binding for MN 20.0.0.6 updated
```

```
MobileIP: Roam timer started for MN 20.0.0.6, lifetime 30000
```

```
MobileIP: MH auth ext added (SPI 300) in reply to MN 20.0.0.6
```

```
MobileIP: HF auth ext added (SPI 220) in reply to MN 20.0.0.6
```

```
MobileIP: HA sent reply to MN 20.0.0.6
```

debug ip mobile mib

To display debugging messages for mobile networks, use the **debug ip mobile mib** command in privileged EXEC mode. To disabled debugging, use the **no** form of this command.

debug ip mobile mib

no debug ip mobile mib

Syntax Description This command has no arguments or keywords.

Defaults Disabled

Command Modes Privileged EXEC

Command History	Release	Modification
	12.3(4)T	This command was introduced.

Usage Guidelines This command is useful for customers deploying mobile networks functionality that need to monitor and debug mobile router information via the Simple Network Management Protocol (SNMP).

Examples The following mobile networks deployment MIB debugging messages are displayed only on certain conditions or when a certain condition fails:

```
Router# debug ip mobile mib

! Mobile router is not enabled
MIPMIB: Mobile Router is not enabled

! Care-of-interface can be set as transmit-only only if its a Serial interface
MIPMIB: Serial interfaces can only be set as transmit-only

! The Care of address can be configured only if foreign agent is running
MIPMIB: FA cannot be started

! Check if home agent is active
MIPMIB: HA is not enabled

! For mobile router configuration, host configuration must have been done already
MIPMIB: MN is not configured

! Mobile Network does not match the existing mobile network
MIPMIB: Conflict with existing mobile networks

! Mobile router present
MIPMIB: MR %i is not configured
```

```
! Static mobile networks can be configured only for single member mobilenetgroups
MIPMIB: MR is part of group %s, network cannot be configured

! If a binding exists for this mobile router, then delete the route for this unconfigured
! mobile network
MIBMIB: Delete static mobile net for MR

! Check if its a dynamically registered mobile network
nMIPMIB: Mobile network %i/%i is dynamically registered, cannot be removed

! Check if the mobile network has already been configured for another group
nMIPMIB: Mobile network already configured for MR

! Check if the network has been dynamically registered
nMIPMIB: Deleted dynamic mobnet %i/%i for MR %i"

! Check if the redundancy group exists
MIPMIB: Redundancy group %s does not exist

! CCoA configuration, use primary interface address as the CCoA
MIPMIB: No IP address on this interface

! CCoA configuration, CCoA address shouldn't be the same as the Home Address
nMIPMIB: Collocated CoA is the same as the Home Address, registrations will fail

ip dhcp class CLASS3
  relay agent information
```

debug ip mobile redundancy



Note

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

To display IP mobility events, use the **debug ip mobile redundancy** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip mobile redundancy

no debug ip mobile redundancy

Syntax Description

This command has no arguments or keywords.

Defaults

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(1)T	This command was introduced.
12.3(4)T	This command was integrated into Cisco IOS Release 12.3(4)T.
12.3(8)T	This command was replaced by the debug ip mobile command.

Examples

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

```
Router# debug ip mobile redundancy

00:19:21: MobileIP: Adding MN service flags to bindupdate
00:19:21: MobileIP: Adding MN service flags 0 init registration flags 1
00:19:21: MobileIP: Adding a hared version cvse - bindupdate
00:19:21: MobileIP: HARElayBindUpdate version number 2MobileIP: MN 40.0.0.20 - sent
BindUpd to HA 7.0.0.3 HAA 7.0.0.4
00:19:21: MobileIP: HA standby maint started - cnt 1
00:19:21: MobileIP: MN 40.0.0.20 - HA rcv BindUpdAck accept from 7.0.0.3 HAA 7.0.0.4
00:19:22: MobileIP: HA standby maint started - cnt 1
```

debug ip mobile router

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

debug ip mobile router [detail]

no debug ip mobile router [detail]

Syntax Description	detail (Optional) Displays detailed mobile router debug messages.
---------------------------	--

Defaults	No default behavior or values
-----------------	-------------------------------

Command Modes	Privileged EXEC
----------------------	-----------------

Command History	Release	Modification
	12.2(4)T	This command was introduced.
	12.2(13)T	This command was enhanced to display information about the addition and deletion of mobile networks.

Usage Guidelines	<p>The mobile router operations can be debugged. The following conditions trigger debugging messages:</p> <ul style="list-style-type: none"> • Agent discovery • Registration • Mobile router state change • Routes and tunnels created or deleted • Roaming information
-------------------------	---

Debugging messages are prefixed with “MobRtr” and detail messages are prefixed with “MobRtrX”.

Examples	The following is sample output from the debug ip mobile router command:
-----------------	--

```
Router# debug ip mobile router

MobileRouter: New FA 27.0.0.12 coa 27.0.0.12 int Ethernet0/1 MAC 0050.50c1.c855
2w2d: MobileRouter: Register reason: isolated
2w2d: MobileRouter: Snd reg request agent 27.0.0.12 coa 27.0.0.12 home 9.0.0.1 ha 29.0.0.4
lifetime 36000 int Ethernet0/1 flag sbdmgtv cnt 0 id B496B69C.55E77974
2w2d: MobileRouter: Status Isolated -> Pending
```

The following is sample output from the **debug ip mobile router detail** command:

```
Router# debug ip mobile router detail
```

```
1d09h: MobRtr: New agent 20.0.0.2 coa 30.0.0.2 int Ethernet3/1 MAC 00b0.8e35.a055
1d09h: MobRtr: Register reason: left home
1d09h: MobRtrX: Extsize 18 add 1 delete 0
1d09h: MobRtrX: Add network 20.0.0.0/8
MobileIP: MH auth ext added (SPI 100) to HA 100.0.0.3
1d09h: MobRtr: Register to fa 20.0.0.2 coa 30.0.0.2 home 100.0.0.1 ha 100.0.0.3 life 120
int Ethernet3/1 flag sbdmgvt cnt 0 id BE804340.447F50A4
1d09h: MobRtr: Status Isolated -> Pending
1d09h: MobRtr: MN rcv accept (0) reply on Ethernet3/1 from 20.0.0.2 lifetime 120
MobileIP: MN 100.0.0.3 - authenticating HA 100.0.0.3 using SPI 100
MobileIP: MN 100.0.0.3 - authenticated HA 100.0.0.3 using SPI 100
1d09h: MobRtr: Status Pending -> Registered
1d09h: MobRtr: Add default gateway 20.0.0.2 (Ethernet3/1)
1d09h: MobRtr: Add default route via 20.0.0.2 (Ethernet3/1)
```

Related Commands

Command	Description
debug ip mobile	Displays Mobile IP information.

debug ip mpacket

To display IP multicast packets received and sent, use the **debug ip mpacket** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

```
debug ip mpacket [vrf vrf-name] [detail | fastswitch] [access-list] [group]
```

```
no debug ip mpacket [vrf vrf-name] [detail | fastswitch] [access-list] [group]
```

Syntax Description

vrf	(Optional) Supports the Multicast Virtual Private Network (VPN) routing and forwarding (VRF) instance.
<i>vrf-name</i>	(Optional) Name assigned to the VRF.
detail	(Optional) Displays IP header information and MAC address information.
fastswitch	(Optional) Displays IP packet information in the fast path.
<i>access-list</i>	(Optional) The access list number.
<i>group</i>	(Optional) The group name or address.

Defaults

The **debug ip mpacket** command displays all IP multicast packets switched at the process level.

Command Modes

Privileged EXEC

Command History

Release	Modification
10.2	This command was introduced.
12.1(2)T	The fastswitch keyword was added.
12.0(23)S	The vrf keyword and <i>vrf-name</i> argument were added.
12.2(13)T	The vrf keyword and <i>vrf-name</i> argument were added in Cisco IOS Release 12.2T.

Usage Guidelines

This command displays information for multicast IP packets that are forwarded from this router. Use the *access-list* or *group* argument to limit the display to multicast packets from sources described by the access list or a specific multicast group.

Use this command with the **debug ip packet** command to display additional packet information.



Note

The **debug ip mpacket** command generates many messages. Use this command with care so that performance on the network is not affected by the debug message traffic.

Examples

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

```
Router# debug ip mpacket 224.2.0.1

IP: s=10.188.34.54 (Ethernet1), d=224.2.0.1 (Tunnel0), len 88, mforward
IP: s=10.188.34.54 (Ethernet1), d=224.2.0.1 (Tunnel0), len 88, mforward
IP: s=10.188.34.54 (Ethernet1), d=224.2.0.1 (Tunnel0), len 88, mforward
IP: s=10.162.3.27 (Ethernet1), d=224.2.0.1 (Tunnel0), len 68, mforward
```

Table 126 describes the significant fields shown in the display.

Table 126 *debug ip mpacket Field Descriptions*

Field	Description
IP	IP packet.
s=10.188.34.54	Source address of the packet.
(Ethernet1)	Name of the interface that received the packet.
d=224.2.0.1	Multicast group address that is the destination for this packet.
(Tunnel0)	Outgoing interface for the packet.
len 88	Number of bytes in the packet. This value will vary depending on the application and the media.
mforward	Packet has been forwarded.

Related Commands

Command	Description
debug ip dvmrp	Displays information on DVMRP packets received and sent.
debug ip igmp	Displays IGMP packets received and sent, and IGMP host-related events.
debug ip mrm	Displays MRM control packet activity.
debug ip packet	Displays general IP debugging information and IPSO security transactions.
debug ip sd	Displays all SD announcements received.

debug ip mrm

To display Multicast Routing Monitor (MRM) control packet activity, use the **debug ip mrm** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip mrm

no debug ip mrm

Syntax Description This command has no arguments or keywords.

Defaults Debugging for MRM is not enabled.

Command Modes Privileged EXEC

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

Examples The following is sample output from the **debug ip mrm** command on the different devices:

On Manager

```
*Feb 28 16:25:44.009: MRM: Send Beacon for group 239.1.1.1, holdtime 86100 seconds
*Feb 28 16:26:01.095: MRM: Receive Status Report from 10.1.4.2 on Ethernet0
*Feb 28 16:26:01.099: MRM: Send Status Report Ack to 10.1.4.2 for group 239.1.1.1
```

On Test-Sender

```
MRM: Receive Test-Sender Request/Local trigger from 1.1.1.1 on Ethernet0
MRM: Send TS request Ack to 1.1.1.1 for group 239.1.2.3
MRM: Send test packet src:2.2.2.2 dst:239.1.2.3 manager:1.1.1.1
```

On Test-Receiver

```
MRM: Receive Test-Receiver Request/Monitor from 1.1.1.1 on Ethernet0
MRM: Send TR request Ack to 1.1.1.1 for group 239.1.2.3
MRM: Receive Beacon from 1.1.1.1 on Ethernet0
MRM: Send Status Report to 1.1.1.1 for group 239.1.2.3
MRM: Receive Status Report Ack from 1.1.1.1 on Ethernet0
```

debug ip mrouting

To display changes to the multicast route (mroute) table, use the **debug ip mrouting** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

```
debug ip mrouting [vrf vrf-name] [rpf-events] [group]
```

```
no debug ip mrouting [vrf vrf-name] [rpf-events] [group]
```

Syntax Description		
vrf	(Optional)	Supports the multicast Virtual Private Network (VPN) routing and forwarding (VRF) instance.
<i>vrf-name</i>	(Optional)	Name assigned to the VRF.
rpf-events	(Optional)	Checks the Reverse Path Forwarding (RPF) events of a specified group.
<i>group</i>	(Optional)	Group name or address to monitor packet activity of a single group.

Defaults No default behavior or values

Command Modes Privileged EXEC

Command History	Release	Modification
	10.2	This command was introduced.
	12.0(22)S	The rpf-events keyword was added.
	12.2(14)S	The vrf keyword and <i>vrf-name</i> argument were added.
	12.2(15)T	The vrf keyword and <i>vrf-name</i> argument were added in Cisco IOS Release 12.2T.

Usage Guidelines This command indicates when the router has made changes to the mroute table. Use the **debug ip pim** and **debug ip mrouting** commands consecutively to obtain additional multicast routing information. In addition, use the **debug ip igmp** command to learn why an mroute message is being displayed.

This command generates a substantial amount of output. Use the optional *group* argument to limit the output to a single multicast group.

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

```
Router# debug ip mrouting 224.2.0.1

MRT: Delete (10.0.0.0/8, 224.2.0.1)
MRT: Delete (10.4.0.0/16, 224.2.0.1)
MRT: Delete (10.6.0.0/16, 224.2.0.1)
MRT: Delete (10.9.0.0/16, 224.2.0.1)
MRT: Delete (10.16.0.0/16, 224.2.0.1)
```

```

MRT: Create (*, 224.2.0.1), if_input NULL
MRT: Create (224.69.15.0/24, 225.2.2.4), if_input Ethernet0, RPF nbr 224.69.61.15
MRT: Create (224.69.39.0/24, 225.2.2.4), if_input Ethernet1, RPF nbr 0.0.0.0
MRT: Create (10.0.0.0/8, 224.2.0.1), if_input Ethernet1, RPF nbr 224.0.0.0
MRT: Create (10.4.0.0/16, 224.2.0.1), if_input Ethernet1, RPF nbr 224.0.0.0
MRT: Create (10.6.0.0/16, 224.2.0.1), if_input Ethernet1, RPF nbr 224.0.0.0
MRT: Create (10.9.0.0/16, 224.2.0.1), if_input Ethernet1, RPF nbr 224.0.0.0
MRT: Create (10.16.0.0/16, 224.2.0.1), if_input Ethernet1, RPF nbr 224.0.0.0

```

The following lines show that multicast IP routes were deleted from the routing table:

```

MRT: Delete (10.0.0.0/8, 224.2.0.1)
MRT: Delete (10.4.0.0/16, 224.2.0.1)
MRT: Delete (10.6.0.0/16, 224.2.0.1)

```

The (*, G) entries are generally created by receipt of an Internet Group Management Protocol (IGMP) host report from a group member on the directly connected LAN or by a Protocol Independent Multicast (PIM) join message (in sparse mode) that this router receives from a router that is sending joins toward the RP. This router will in turn send a join toward the Route Processor (RP) that creates the shared tree (or RP tree).

```

MRT: Create (*, 224.2.0.1), if_input NULL

```

The following lines are an example of creating an (S, G) entry that shows that an IP multicast packet (mpacket) was received on Ethernet interface 0. The second line shows a route being created for a source that is on a directly connected LAN. The RPF means “Reverse Path Forwarding,” whereby the router looks up the source address of the multicast packet in the unicast routing table and determines which interface will be used to send a packet to that source.

```

MRT: Create (224.69.15.0/24, 225.2.2.4), if_input Ethernet0, RPF nbr 224.69.61.15
MRT: Create (224.69.39.0/24, 225.2.2.4), if_input Ethernet1, RPF nbr 224.0.0.0

```

The following lines show that multicast IP routes were added to the routing table. Note the 224.0.0.0 as the RPF, which means the route was created by a source that is directly connected to this router.

```

MRT: Create (10.9.0.0/16, 224.2.0.1), if_input Ethernet1, RPF nbr 224.0.0.0
MRT: Create (10.16.0.0/16, 224.2.0.1), if_input Ethernet1, RPF nbr 224.0.0.0

```

If the source is not directly connected, the neighbor address shown in these lines will be the address of the router that forwarded the packet to this router.

The shortest path tree state maintained in routers consists of source (S), multicast address (G), outgoing interface (OIF), and incoming interface (IIF). The forwarding information is referred to as the multicast forwarding entry for (S, G).

An entry for a shared tree can match packets from any source for its associated group if the packets come through the proper incoming interface as determined by the RPF lookup. Such an entry is denoted as (*, G). A (*, G) entry keeps the same information a (S, G) entry keeps, except that it saves the rendezvous point address in place of the source address in sparse mode or as 24.0.0.0 in dense mode.

[Table 127](#) describes the significant fields shown in the display.

Table 127 *debug ip mrouting Field Descriptions*

Field	Description
MRT	Multicast route table.
RPF	Reverse Path Forwarding.
nbr	Neighbor.

Related Commands

Command	Description
debug ip dvmrp	Displays information on DVMRP packets received and sent.
debug ip igmp	Displays IGMP packets received and sent, and IGMP host-related events.
debug ip packet	Displays general IP debugging information and IPSO security transactions.
debug ip pim	Displays all PIM announcements received.
debug ip sd	Displays all SD announcements received.

debug ip mrouting limit

To debug mroute limiting and to determine the reason for the limiting, use the **debug ip mrouting limit** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip mrouting limit

no debug ip mrouting limit

Syntax Description This command has no arguments or keywords.

Command Modes Privileged EXEC

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

Usage Guidelines This command debugs all mroute entries limited by the **ip multicast limit** command, and it displays the reason for the limiting.

Examples The following example shows a limit increment, a decrement, and a denial to add an mroute in which the maximum for a standard access list was reached:

```
Router# debug ip mrouting limit

MRL(0): incr-ed acl 'rpf-list' to (13 < max 32), [n:0,p:0], (main) Ethernet1/0,
(40.202.60.41, 225.30.200.60)
MRL(0): decr-ed acl 'rpf-list' to (10 < max 32), [n:0,p:0], (main) Ethernet1/0, (*,
225.40.202.60)
MRL(0): Add mroute (42.43.0.43, 225.30.200.60) denied for Ethernet0/2, acl std-list, (16 =
max 16)
```