

# debug dmsp doc-to-fax



## Note

Effective with release 12.3(8)T, the **debug dmsp doc-to-fax** command is replaced by the **debug fax dmsp** command. See the **debug fax dmsp** command for more information.

To display debugging messages for the doc Media Service Provider (docMSP) TIFF or text2Fax engine, use the **debug dmsp doc-to-fax** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

```
debug dmsp doc-to-fax [text-to-fax | tiff-reader]
```

```
no debug dmsp doc-to-fax [text-to-fax | tiff-reader]
```

## Syntax Description

<b>text-to-fax</b>	(Optional) Displays debugging messages that occur while the DocMSP Component is receiving text packets and producing T4 fax data.
<b>tiff-reader</b>	(Optional) Displays debugging messages that occur while the DocMSP Component is receiving TIFF packets and producing T4 fax data.

## Defaults

No default behavior or values

## Command Modes

Privileged EXEC

## Command History

Release	Modification
12.1(3)XI	This command was introduced on the Cisco AS5300 access server.
12.3(8)T	This command was replaced by the <b>debug fax dmsp</b> command.

## Examples

The following is sample output from the **debug dmsp doc-to-fax** command:

```
Router# debug dmsp doc-to-fax

Jan  1 04:58:39.898: docmsp_call_setup_request: callid=18
Jan  1 04:58:39.902: docmsp_call_setup_request():  ramp data dir=OFFRAMP, conf dir=SRC
Jan  1 04:58:39.902: docmsp_caps_ind: call id=18, src=17
Jan  1 04:58:39.902: docmsp_bridge cfid=5, srccid=18, dstcid=17

Jan  1 04:58:39.902: docmsp_bridge():  ramp data dir=OFFRAMP, conf dir=SRC, encode out=2
Jan  1 04:58:39.902: docmsp_rcv_msp_ev: call id =18, evID = 42
Jan  1 04:58:39.902: docmsp_bridge cfid=6, srccid=18, dstcid=15

Jan  1 04:58:39.902: docmsp_bridge():  ramp data dir=OFFRAMP, conf dir=DEST, encode out=2
Jan  1 04:58:39.902: docmsp_process_rcv_data: call id src=0, dst=18
Jan  1 04:58:39.902: docmsp_generate_page:
Jan  1 04:58:39.902: docmsp_generate_page: new context for Call 18
Jan  1 04:58:39.922: docmsp_get_msp_event_buffer:
Jan  1 04:58:42.082: docmsp_xmit: call id src=15, dst=18
Jan  1 04:58:42.082: docmsp_process_rcv_data: call id src=15, dst=18
Jan  1 04:58:42.082: offramp_data_process:
Jan  1 04:58:42.102: docmsp_xmit: call id src=15, dst=18
```

```
Jan 1 04:58:42.106: docmsp_process_rcv_data: call id src=15, dst=18
Jan 1 04:58:42.106: offramp_data_process:
Jan 1 04:58:42.122: docmsp_xmit: call id src=15, dst=18
Jan 1 04:58:42.126: docmsp_process_rcv_data: call id src=15, dst=18
Jan 1 04:58:42.126: offramp_data_process:
Jan 1 04:58:42.142: docmsp_xmit: call id src=15, dst=18
Jan 1 04:58:42.146: docmsp_xmit: call id src=15, dst=18
```

**Related Commands**

<b>Command</b>	<b>Description</b>
<b>debug dmosp fax-to-doc</b>	Displays debugging messages for doc MPS fax-to-doc.

# debug dmsp fax-to-doc



## Note

Effective with release 12.3(8)T, the **debug dmsp fax-to-doc** command is replaced by the **debug fax dmsp** command. See the **debug fax dmsp** command for more information.

To display debugging messages for doc MSP (docMSP) fax-to-doc, use the **debug dmsp fax-to-doc** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug dmsp fax-to-doc [tiff-writer]**

**no debug dmsp fax-to-doc [tiff-writer]**

## Syntax Description

<b>tiff-writer</b>	(Optional) Displays debug messages that occur while the DocMSP Component is receiving T4 fax data and producing TIFF packets.
--------------------	---

## Defaults

No default behavior or values

## Command Modes

Privileged EXEC

## Command History

Release	Modification
12.1(3)XI	This command was introduced on the Cisco AS5300 access server.
12.3(8)T	This command was replaced by the <b>debug fax dmsp</b> command.

## Examples

The following is sample output from the **debug dmsp fax-to-doc** command:

```
Router# debug dmsp fax-to-doc

*Oct 16 08:29:54.487: docmsp_call_setup_request: callid=22
*Oct 16 08:29:54.487: docmsp_call_setup_request(): ramp data dir=OFFRAMP, conf dir=SRC
*Oct 16 08:29:54.487: docmsp_caps_ind: call id=22, src=21
*Oct 16 08:29:54.487: docmsp_bridge cfid=15, srccid=22, dstcid=21

*Oct 16 08:29:54.487: docmsp_bridge(): ramp data dir=OFFRAMP, conf dir=SRC, encode out=2
*Oct 16 08:29:54.487: docmsp_bridge cfid=16, srccid=22, dstcid=17

*Oct 16 08:29:54.487: docmsp_bridge(): ramp data dir=OFFRAMP, conf dir=DEST, encode out=2
*Oct 16 08:29:54.487: docmsp_xmit: call id src=17, dst=22
*Oct 16 08:29:54.487: docmsp_process_rcv_data: call id src=17, dst=22
*Oct 16 08:29:54.487: offramp_data_process:
*Oct 16 08:29:54.515: docmsp_get_msp_event_buffer:
*Oct 16 08:29:56.115: docmsp_call_setup_request: callid=24
*Oct 16 08:29:56.115: docmsp_call_setup_request(): ramp data dir=ONRAMP, conf dir=DEST
*Oct 16 08:29:56.115: docmsp_caps_ind: call id=24, src=20
*Oct 16 08:29:56.115: docmsp_bridge cfid=17, srccid=24, dstcid=20
```

---

**Related Commands**

---

<b>Command</b>	<b>Description</b>
<b>debug dmsp doc-to-fax</b>	Displays debugging messages for the doc Media Service Provider TIFF or text2Fax engine.

---

# debug dot1x

To display 802.1X debugging information, use the **debug dot1x** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug dot1x** [aaa | all | process | rxdata | state-machine | supplicant | txdata | vlan]

**no debug dot1x** [aaa | all | process | rxdata | state-machine | supplicant | txdata | vlan]

Syntax Description	
<b>aaa</b>	(Optional) Provides information for authentication, authorization, and accounting (AAA) communications.
<b>all</b>	(Optional) Enables all 802.1X debugging messages.
<b>process</b>	(Optional) Provides information regarding the 802.1X process.
<b>rxdata</b>	(Optional) Provides information for packets that have been received from clients.
<b>state-machine</b>	(Optional) Provides information regarding the 802.1X state machine.
<b>supplicant</b>	(Optional) Provides information about the supplicant.
<b>txdata</b>	(Optional) Provides information regarding packets that have been transmitted to clients.
<b>vlan</b>	(Optional) Provides information regarding the MAC address-based VLAN operation.
	<b>Note</b> VLAN interfaces are currently not supported.

**Command Modes** Privileged EXEC

Command History	Release	Modification
	12.3(2)XA	This command was introduced.
	12.3(4)T	This command was integrated into Cisco IOS Release 12.3(4)T.
	12.3(11)T	The <b>supplicant</b> keyword was added.

## Examples

The following is sample output for the **debug dot1x** command:

```
Router# debug dot1x

*Mar 31 03:42:27.190: dot1x_hold_enqueue: Enqueued the packet to global hold queue

*Mar 31 03:42:27.206: dot1x_process: dequeued a hold client from DOT1X_HOLDQ for
macaddr=0001.024b.87ce, hwidb=Ethernet0
*Mar 31 03:42:27.210: dot1x_aaa_add_client_entry: 0001.024b.87ce is added to the client
list
*Mar 31 03:42:27.210: dot1x_aaa_run_auth_methods: Start auth method EAP
*Mar 31 03:42:27.210: dot1x_aaa_start: in the dot1x_aaa_start
*Mar 31 03:42:27.210: dot1x_process: DOT1X_EVENT_BEING_HANDLED
*Mar 31 03:42:27.214: dot1x_run_rfsm: current state INIT, received EAP_START, mac:
0001.024b.87ce
```

```

*Mar 31 03:42:27.214: dot1x_send_id_req_to_client: sending identity request for
0001.024b.87ce
*Mar 31 03:42:27.214: dot1x_client_send_packet: Sending out an eapol packet
*Mar 31 03:42:27.214: EAPOL pak dump tx
*Mar 31 03:42:27.214: EAPOL Version: 0x1 type: 0x0 length: 0x0005
*Mar 31 03:42:27.218: EAP code: 0x1 id: 0x1 length: 0x0005 type: 0x1
01E01E90: 01000005 01010005 01 .....
*Mar 31 03:42:27.226: mac_identity_check: DOT1X packet, can't drop it
*Mar 31 03:42:27.226: dot1x_eapol_enqueue: Enqueued the eapol packet to global eapol queue

*Mar 31 03:42:27.218: dot1x_send_id_req_to_client: Starting timer client_timeout
*Mar 31 03:42:27.226: dot1x_start_timer: Started the Timer for client 0001.024b.87ce, 30
seconds
*Mar 31 03:42:27.226: dot1x_process: dequeued a packet from EAPOL Q link-type 0x8A
*Mar 31 03:42:27.226: dot1x_parse_client_pak: Received EAPOL packet from 0001.024b.87ce
*Mar 31 03:42:27.226: EAPOL pak dump rx
*Mar 31 03:42:27.226: EAPOL Version: 0x1 type: 0x0 length: 0x000A
*Mar 31 03:42:27.226: EAP code: 0x2 id: 0x1 length: 0x000A type: 0x1
01E01D60: 0100000A 0201000A 01636973 636F .....cisco
*Mar 31 03:42:27.230: dot1x_run_rfsm: current state CLIENT_WAIT, received CLIENT_REPLY,
mac: 0001.024b.87ce
*Mar 31 03:42:27.230: dot1x_stop_timer: Stopped the Timer for client 0001.024b.87ce
*Mar 31 03:42:27.234: dot1x_send_response_to_server: Sending client data to server
*Mar 31 03:42:27.234: dot1x_send_aaa_request: Trying to send the aaa request to auth
server

*Mar 31 03:42:27.238: EAP pak dump tx
*Mar 31 03:42:27.238: EAP code: 0x2 id: 0x1 length: 0x000A type: 0x1
01E01D60: 0201000A 01636973 636F ....cisco
*Mar 31 03:42:27.238: dot1x_send_aaa_request: aaa request sent successfully
*Mar 31 03:42:27.242: dot1x_send_response_to_server: Starting server timer
*Mar 31 03:42:27.242: dot1x_start_timer: Started the Timer for client 0001.024b.87ce, 30
seconds
*Mar 31 03:42:49.294: dot1x_parse_aaa_resp: received AAA response for 0001.024b.87ce
*Mar 31 03:42:49.294: dot1x_parse_aaa_resp: Received server response: FAIL
*Mar 31 03:42:49.298: dot1x_run_rfsm: current state SERVER_WAIT, received SERVER_FAIL,
mac: 0001.024b.87ce
*Mar 31 03:42:49.298: dot1x_stop_timer: Stopped the Timer for client 0001.024b.87ce
*Mar 31 03:42:49.298: dot1x_send_client_fail: Authentication failed for 0001.024b.87ce
*Mar 31 03:42:49.298: dot1x_client_send_packet: Sending out an eapol packet
*Mar 31 03:42:49.302: EAPOL pak dump tx
*Mar 31 03:42:49.302: EAPOL Version: 0x1 type: 0x0 length: 0x0004
*Mar 31 03:42:49.302: EAP code: 0x4 id: 0x1 length: 0x0004
01F22460: 01000004 .....
01F22470: 04010004 ....
*Mar 31 03:42:49.306: dot1x_send_client_fail: Starting the quietWhile timer for
0001.024b.87ce as the client supports dot1x
*Mar 31 03:42:49.306: dot1x_start_timer: Started the Timer for client 0001.024b.87ce, 120
seconds
*Mar 31 03:42:49.310: 0001.024b.87ce(Ethernet0->Ethernet0): Processing started
*Mar 31 03:42:49.310: 0001.024b.87ce(Ethernet0->Ethernet0): Unable to determine IP address

```

Table 64 describes significant fields shown in the display.

**Table 64** *debug dot1x Field Descriptions*

Field	Description
dot1x_aaa_run_auth_methods	Starts the authentication process for the user.
dot1x_run_rfsm	Makes the state machine transition. Various states include CLIENT_WAIT (waiting for a response from the client), SERVER_WAIT, SERVER_PASS (a positive authentication result has been received from the server), and SERVER_FAIL.
mac_identity_check	Checks the authentication result for the user.

#### Related Commands

Command	Description
<b>clear dot1x</b>	Clears 802.1X interface information.
<b>identity profile default</b>	Creates an identity profile and enters dot1x profile configuration mode.
<b>show dot1x</b>	Shows details for an identity profile.

# debug dot1x (EtherSwitch)

To enable debugging of the 802.1x protocol when an Ethernet switch network module is installed, use the **debug dot1x** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug dot1x** {all | authsm | backend | besm | core | reauthsm }

**no debug dot1x** {all | authsm | backend | besm | core | reauthsm }

## Syntax Description

<b>all</b>	Enables debugging of all conditions.
<b>authsm</b>	Enables debugging of the authenticator state machine, which is responsible for controlling access to the network through 802.1x-enabled ports.
<b>backend</b>	Enables debugging of the interaction between the 802.1x process and the router RADIUS client.
<b>besm</b>	Enables debugging of the backend state machine, which is responsible for relaying authentication request between the client and the authentication server.
<b>core</b>	Enables debugging of the 802.1x process, which includes 802.1x initialization, configuration, and the interaction with the port manager module.
<b>reauthsm</b>	Enables debugging of the reauthentication state machine, which manages periodic reauthentication of the client.

## 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.

## Usage Guidelines

The **undebug dot1x** command is the same as the **no debug dot1x** command.

**Related Commands**

<b>Command</b>	<b>Description</b>
<b>show debugging</b>	Displays information about the types of debugging that are enabled.
<b>show dot1x</b>	Displays 802.1x statistics, administrative status, and operational status for the router or for the specified interface.

# debug drip event

To display debugging messages for Duplicate Ring Protocol (DRiP) events, use the **debug drip event** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug drip event**

**no debug drip event**

**Syntax Description** This command has no arguments or keywords.

**Defaults** Debugging is disabled for DRiP events.

**Command Modes** Privileged EXEC

Command History	Release	Modification
	11.3(4)T	This command was introduced.

**Usage Guidelines** When a TrBRF interface is configured on the Remote Switch Module (RSM), the DRiP protocol is activated. The DRiP protocol adds the VLAN ID specified in the router command to its database and recognizes the VLAN as a locally configured, active VLAN.

**Examples** The following is sample output from the **debug drip event** command:

```
Router# debug drip event

DRiP gets a packet from the network:
612B92C0: 01000C00 00000000 0C501900 0000AAAA .....P....**
612B92D0: 0300000C 00020000 00000100 0CCCCCCC .....LLL
612B92E0: 00000C50 19000020 AAAA0300 000C0102 ...P... **.....
612B92F0: 01010114 00000002 00000002 00000C50 .....P
612B9300: 19000001 04C00064 04 .....@.d.
```

DRiP gets a packet from the network:

Recv. pak

DRiP recognizes that the VLAN ID it is getting is a new one from the network:

```
6116C840: 0100 0CCCCCCC ...LLL
6116C850: 00102F72 CBF0024 AAAA0300 000C0102 ../rK{.$**.....
6116C860: 01FF0214 0002E254 00015003 00102F72 .....bT..P.../r
6116C870: C8000010 04C00014 044003EB 14 H....@...@.k.
DRIP : remote update - Never heard of this vlan
```

DRiP attempts to resolve any conflicts when it discovers a new VLAN. The value action = 1 means to notify the local platform of change in state.

```
DRIP : resolve remote for vlan 20 in VLAN0
DRIP : resolve remote - action = 1
```

The local platform is notified of change in state:

```
DRIP Change notification active vlan 20
```

Another new VLAN ID was received in the packet:

```
DRIP : resolve remote for vlan 1003 in Vlan0
```

No action is required:

```
DRIP : resolve remote - action = 0
```

Thirty seconds have expired, and DRiP sends its local database entries to all its trunk ports:

```
DRIP : local timer expired
DRIP : transmit on 0000.0c50.1900, length = 24
612B92C0: 01000C00 00000000 0C501900 0000AAAA .....P....**
612B92D0: 0300000C 00020000 00000100 0CCCCCCC .....LLL
612B92E0: 00000C50 19000020 AAAA0300 000C0102 ...P... **.....
612B92F0: 01FF0114 00000003 00000002 00000C50 .....P
612B9300: 19000001 04C00064 04 .....@.d.
```

# debug drip packet

To display debugging messages for Duplicate Ring Protocol (DRiP) packets, use the **debug drip packet** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug drip packet**

**no debug drip packet**

**Syntax Description** This command has no arguments or keywords.

**Defaults** Debugging is not enabled for DRiP packets.

**Command Modes** Privileged EXEC

Command History	Release	Modification
	11.3(4)T	This command was introduced.

**Usage Guidelines** Before you use this command, you can optionally use the **clear drip** command first. As a result the DRiP counters are reset to 0. If the DRiP counters begin to increment, the router is receiving packets.

**Examples** The following is sample output from the **debug drip packet** command:

```
Router# debug drip packet
```

The following type of output is displayed when a packet is entering the router and you use the **show debug** command:

```
039E5FC0:      0100 0CCCCCCC 00E0A39B 3FFB0028      ...LLL.`#.?{(
039E5FD0: AAAA0300 000C0102 01FF0314 0000A5F6      **.....%v
039E5FE0: 00008805 00E0A39B 3C000000 04C00028      ....`#.<....@.(
039E5FF0: 04C00032 044003EB 0F          .@.2.@.k.
039FBD20:                01000C00 00000010      .....
```

The following type of output is displayed when a packet is sent by the router:

```
039FBD30: A6AEB450 0000AAAA 0300000C 00020000      &.4P.**.....
039FBD40: 00000100 0CCCCCCC 0010A6AE B4500020      ....LLL..&.4P.
039FBD50: AAAA0300 000C0102 01FF0114 00000003      **.....
039FBD60: 00000002 0010A6AE B4500001 04C00064      .....&.4P...@.d
039FBD70: 04          .
```

Related Commands	Command	Description
	<b>debug drip event</b>	Displays debugging messages for DRiP events.

# debug dsc clock

To display debugging output for the time-division multiplexing (TDM) clock-switching events on the dial shelf controller (DSC), use the **debug dsc clock** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**[execute-on] debug dsc clock**

**[execute-on] no debug dsc clock**

## Syntax Description

This command has no arguments or keywords; however, it can be used with the **execute-on** command.

## Command Modes

Privileged EXEC

## Command History

Release	Modification
11.3(2)AA	This command was introduced.

## Usage Guidelines

To perform this command from the router shelf on the Cisco AS5800 series platform, use the **execute-on slot slot-number debug dsc clock** form of this command.

The **debug dsc clock** command displays TDM clock-switching events on the dial shelf controller. The information displayed includes the following:

- Clock configuration messages received from trunks via NBUS
- Dial shelf controller clock configuration messages from the router shelf over the dial shelf interface link
- Clock switchover algorithm events

## Examples

The following example shows that the **debug dsc clock** command has been enabled, and that trunk messages are received, and that the configuration message has been received:

```
AS5800# debug dsc clock

Dial Shelf Controller Clock debugging is on
AS5800#
00:02:55: Clock Addition msg of len 12 priority 8 from slot 1 port 1 on line 0
00:02:55: Trunk 1 has reloaded
```

## Related Commands

Command	Description
<b>execute-on</b>	Executes commands remotely on a line card.
<b>show dsc clock</b>	Displays information about the dial shelf controller clock.

# debug dsip

To display debugging output for Distributed System Interconnect Protocol (DSIP) used between a router shelf and a dial shelf, use the **debug dsip** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug dsip** {all | api | boot | console | trace | transport}

**no debug dsip** {all | api | boot | console | trace | transport}

## Syntax Description

<b>all</b>	View all DSIP debugging messages.
<b>api</b>	View DSIP client interface (API) debugging messages.
<b>boot</b>	View DSIP booting messages that are generated when a download of the feature board image is occurring properly.
<b>console</b>	View DSIP console operation while debugging.
<b>trace</b>	Enable logging of header information concerning DSIP packets entering the system into a trace buffer. This logged information can be viewed with the <b>show dsip tracing</b> command.
<b>transport</b>	Debug the DSIP transport layer, the module that interacts with the underlying physical media driver.

## Command Modes

Privileged EXEC

## Command History

Release	Modification
11.3(2)AA	This command was introduced.

## Usage Guidelines

The **debug dsip** command is used to enable the display of debugging messages for DSIP between the router shelf and the dial shelf. Using this command, you can display booting messages generated when the download of an image occurs, view console operation, and trace logging of MAC header information and DSIP transport layer information as modules interact with the underlying physical media driver. This command can be applied to a single modem or a group of modems.

Once the **debug dsip trace** command has been enabled, you can read the information captured in the trace buffer using the **show dsip tracing** command.

**Examples**

The following example indicates the **debug dsip trace** command logs MAC headers of the various classes of DSIP packets. To view the logged information, use the **show dsip tracing** command:

```
AS5800# debug dsip trace

NIP tracing debugging is on

AS5800# show dsip tracing

NIP Control Packet Trace
-----
Dest:00e0.b093.2238 Src:0007.4c72.0058 Type:200B SrcShelf:1 SrcSlot:11
MsgType:0 MsgLen:82 Timestamp: 00:49:14
-----
Dest:00e0.b093.2238 Src:0007.4c72.0028 Type:200B SrcShelf:1 SrcSlot:5
MsgType:0 MsgLen:82 Timestamp: 00:49:14
-----
```

**Related Commands**

Command	Description
<b>debug modem</b>	Displays information about the dial shelf, including clocking information.
<b>show dsip tracing</b>	Displays DSIP media header information logged using the <b>debug dsip trace</b> command.

# debug dspapi



## Note

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

To enable debugging for Digital Signal Processor (DSP) application programming interface (API) message events, use the **debug dspapi** command in privileged EXEC mode. To reset the default value for this feature, use the **no** form of this command.

```
debug dspapi {all | command | detail | error | notification | response}
```

```
no debug dspapi {all | command | detail | error | notification | response}
```

## Syntax Description

<b>all</b>	Enables all <b>debug dspapi</b> options (command, detail, error, notification and response).
<b>command</b>	Displays commands sent to the DSPs.
<b>detail</b>	Displays additional detail for the DSP API debugs enabled.
<b>error</b>	Displays any DSP API errors.
<b>notification</b>	Displays notification messages sent from the DSP (for example, tone detection notification).
<b>response</b>	Displays responses sent by the DSP (for example, responses to statistic requests).

## Defaults

This command is not enabled.

## Command Modes

Privileged EXEC

## Command History

Release	Modification
12.1(5)XM	This command was introduced on the Cisco AS5300 and Cisco AS5800.
12.1(5)XM1	This command was implemented on the Cisco AS5350 and Cisco AS5400.
12.2(2)T	This command was implemented on the Cisco 1700, Cisco 2600 series, Cisco 3600 series, and the Cisco 3810.
12.2(8)T	This command was integrated into Cisco IOS Release 12.2(8)T.
12.3(8)T	This command was replaced by the <b>debug voip dspapi</b> command.

## Usage Guidelines

DSP API message events used to communicate with DSPs are intended for use with Connexant (Nextport) and Texas Instrument (54x) DSPs. This command severely impacts performance and should be used only for single-call debug capture.

---

**Examples**

The following example shows how to enable debugging for all DSP API message events:

```
Router# debug dspapi all
```

---

**Related Commands**

Command	Description
<b>debug hpi</b>	Enables debugging for HPI message events.

---

# debug dspfarm

To display digital signal processor (DSP) farm service debugging information, use the **debug dspfarm** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug dspfarm** {**all** | **errors** | **events** | **packets**}

**no debug dspfarm**

## Syntax Description

<b>all</b>	All DSP-farm debug-trace information.
<b>errors</b>	DSP-farm errors.
<b>events</b>	DSP-farm events.
<b>packets</b>	DSP-farm packets.

## Defaults

No default behavior or values

## Command Modes

Privileged EXEC

## Command History

Release	Modification
12.1(5)YH	This command was introduced on the Cisco VG200.
12.2(13)T	This command was implemented on the Cisco 2600 series, Cisco 3620, Cisco 3640, Cisco 3660, and Cisco 3700 series.

## Usage Guidelines

The router on which this command is used must be equipped with one or more digital T1/E1 packet voice trunk network modules (NM-HDVs) or high-density voice (HDV) transcoding/conferencing DSP farms (NM-HDV-FARMS) to provide DSP resources.

Debugging is turned on for all DSP-farm-service sessions. You can debug multiple sessions simultaneously, with different levels of debugging for each.

## Examples

The following is sample output from the **debug dspfarm events** command:

```
Router# debug dspfarm events
```

```
DSP Farm service events debugging is on
```

```
*Mar 1 00:45:51: Sent 180 bytes to DSP 4 channel 2
*Mar 1 00:45:53: Sent 180 bytes to DSP 4 channel 3
*Mar 1 00:45:55: Sent 180 bytes to DSP 4 channel 1
*Mar 1 00:45:56: Sent 180 bytes to DSP 4 channel 2
*Mar 1 00:45:58: Sent 180 bytes to DSP 4 channel 3
*Mar 1 00:46:00: Sent 180 bytes to DSP 4 channel 1
*Mar 1 00:46:01: xapi_dspfarm_modify_connection: sess_id 26, conn_id 2705, conn_mode 3,
ripaddr 10.10.1.7, rport 20170
*Mar 1 00:46:01: dspfarm_process_appl_event_queue: XAPP eve 6311C4B0 rcvd
```

```

*Mar 1 00:46:01: dspfarm_find_stream: stream 63121F1C, found in sess 631143CC, cid 2705
*Mar 1 00:46:01: dspfarm_modify_connection: old_mode 4, new_mode 3
*Mar 1 00:46:01: dspfarm_close_local_rtp: stream 63121F1C, local_rtp_port 22656
*Mar 1 00:46:01: xapi_dspfarm_enqueue_event_to_appl: handle 63120634, event 6311C4C8,
eve_id 5, context 6311426C, result 0
*Mar 1 00:46:01: xapi_dspfarm_delete_connection: sess_id 26, conn_id 2705
*Mar 1 00:46:01: dspfarm_process_appl_event_queue: XAPP eve 6311C4E0 rcvd
*Mar 1 00:46:01: dspfarm_find_stream: stream 63121F1C, found in sess 631143CC, cid 2705
*Mar 1 00:46:01: dspfarm_close_local_rtp: stream 63121F1C, local_rtp_port 0
*Mar 1 00:46:01: dspfarm_release_dsp_resource: sess 631143CC, stream 63121F1C, num_stream
3, sess_type 2, sess_dsp_id 2040000, stream_dsp_id 2040002
*Mar 1 00:46:01: dspfarm_drop_conference:slot 2 dsp 4 ch 2
*Mar 1 00:46:01: dspfarm_send_drop_conf: Sent drop_conference to DSP 4 ch 2
*Mar 1 00:46:01: dspfarm_xapp_eng: Sent msg 8 to DSPFARM
*Mar 1 00:46:01: xapi_dspfarm_enqueue_event_to_appl: handle 63120634, event 6311C4F8,
eve_id 9, context 6311426C, result 0
*Mar 1 00:46:01: dspfarm_process_dsp_event_queue: DSP eve 6312078C rcvd
*Mar 1 00:46:01: dspfarm_delete_stream: sess_id 26, conn_id 2705, stream 63121F1C, in
sess 631143CC is freed
*Mar 1 00:46:01: Sent 180 bytes to DSP 4 channel 3
*Mar 1 00:46:04: Sent 180 bytes to DSP 4 channel 3
*Mar 1 00:46:05: xapi_dspfarm_modify_connection: sess_id 26, conn_id 2689, conn_mode 3,
ripaddr 10.10.1.5, rport 19514
*Mar 1 00:46:05: dspfarm_process_appl_event_queue: XAPP eve 6311C510 rcvd
*Mar 1 00:46:05: dspfarm_find_stream: stream 63121E34, found in sess 631143CC, cid 2689
*Mar 1 00:46:05: dspfarm_modify_connection: old_mode 4, new_mode 3
*Mar 1 00:46:05: dspfarm_close_local_rtp: stream 63121E34, local_rtp_port 25834
*Mar 1 00:46:05: xapi_dspfarm_enqueue_event_to_appl: handle 63120634, event 6311C528,
eve_id 5, context 63114244, result 0
*Mar 1 00:46:05: xapi_dspfarm_delete_connection: sess_id 26, conn_id 2689
*Mar 1 00:46:05: dspfarm_process_appl_event_queue: XAPP eve 6311C540 rcvd
*Mar 1 00:46:05: dspfarm_find_stream: stream 63121E34, found in sess 631143CC, cid 2689
*Mar 1 00:46:05: dspfarm_close_local_rtp: stream 63121E34, local_rtp_port 0
*Mar 1 00:46:05: dspfarm_release_dsp_resource: sess 631143CC, stream 63121E34, num_stream
2, sess_type 2, sess_dsp_id 2040000, stream_dsp_id 2040001
*Mar 1 00:46:05: dspfarm_drop_conference:slot 2 dsp 4 ch 1
*Mar 1 00:46:05: dspfarm_send_drop_conf: Sent drop_conference to DSP 4 ch 1
*Mar 1 00:46:05: dspfarm_xapp_eng: Sent msg 8 to DSPFARM
*Mar 1 00:46:05: xapi_dspfarm_enqueue_event_to_appl: handle 63120634, event 6311C558,
eve_id 9, context 63114244, result 0
*Mar 1 00:46:05: dspfarm_process_dsp_event_queue: DSP eve 6311586C rcvd
*Mar 1 00:46:05: dspfarm_delete_stream: sess_id 26, conn_id 2689, stream 63121E34, in
sess 631143CC is freed
*Mar 1 00:46:05: xapi_dspfarm_modify_connection: sess_id 26, conn_id 2721, conn_mode 3,
ripaddr 10.10.1.6, rport 21506
*Mar 1 00:46:05: dspfarm_process_appl_event_queue: XAPP eve 6311C570 rcvd
*Mar 1 00:46:05: dspfarm_find_stream: stream 63122004, found in sess 631143CC, cid 2721
*Mar 1 00:46:05: dspfarm_modify_connection: old_mode 4, new_mode 3
*Mar 1 00:46:05: dspfarm_close_local_rtp: stream 63122004, local_rtp_port 19912
*Mar 1 00:46:05: xapi_dspfarm_enqueue_event_to_appl: handle 63120634, event 6311C588,
eve_id 5, context 63114294, result 0
*Mar 1 00:46:05: xapi_dspfarm_delete_connection: sess_id 26, conn_id 2721
*Mar 1 00:46:05: dspfarm_process_appl_event_queue: XAPP eve 6311C5A0 rcvd
*Mar 1 00:46:05: dspfarm_find_stream: stream 63122004, found in sess 631143CC, cid 2721
*Mar 1 00:46:05: dspfarm_close_local_rtp: stream 63122004, local_rtp_port 0
*Mar 1 00:46:05: dspfarm_release_dsp_resource: sess 631143CC, stream 63122004, num_stream
1, sess_type 2, sess_dsp_id 2040000, stream_dsp_id 2040003
*Mar 1 00:46:05: dspfarm_drop_conference:slot 2 dsp 4 ch 3
*Mar 1 00:46:05: dspfarm_drop_conference: Last conferee - closing the conf session
*Mar 1 00:46:05: dspfarm_send_close_conf: Sent close_conference to DSP 4
*Mar 1 00:46:05: dspfarm_drop_conference: Removed the conf in dsp 4
*Mar 1 00:46:05: dspfarm_xapp_eng: Sent msg 8 to DSPFARM
*Mar 1 00:46:05: xapi_dspfarm_enqueue_event_to_appl: handle 63120634, event 6311C5B8,
eve_id 9, context 63114294, result 0

```

```
*Mar 1 00:46:05: dspfarm_process_dsp_event_queue: DSP eve 6311586C rcvd
*Mar 1 00:46:05: dspfarm_delete_stream: sess_id 26, conn_id 2721, stream 63122004, in
sess 631143CC is freed
```

**Related Commands**

<b>Command</b>	<b>Description</b>
<b>debug frame-relay vc-bundle</b>	Sets debugging for SCCP and its applications at one of four levels.
<b>dspfarm (DSP farm)</b>	Enables DSP-farm service.
<b>sccp</b>	Enables SCCP and its associated transcoding and conferencing applications.
<b>show dspfarm</b>	Displays summary information about DSP resources.

# debug dspu activation

To display information on downstream physical unit (DSPU) activation, use the **debug dspu activation** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug dspu activation** [*name*]

**no debug dspu activation** [*name*]

## Syntax Description

*name* (Optional) The host or physical unit (PU) name designation.

## Command Modes

Privileged EXEC

## Usage Guidelines

The **debug dspu activation** command displays all DSPU activation traffic. To restrict the output to a specific host or PU, include the host or PU *name* argument. You cannot turn off debugging output for an individual PU if that PU has not been named in the **debug dspu activation** command.

## Examples

The following is sample output from the **debug dspu activation** command. Not all intermediate numbers are shown for the “activated” and “deactivated” logical unit (LU) address ranges.

```
Router# debug dspu activation

DSPU: LS HOST3745 connected
DSPU: PU HOST3745 activated
DSPU: LU HOST3745-2 activated
DSPU: LU HOST3745-3 activated
.
.
.
DSPU: LU HOST3745-253 activated
DSPU: LU HOST3745-254 activated

DSPU: LU HOST3745-2 deactivated
DSPU: LU HOST3745-3 deactivated
.
.
.
DSPU: LU HOST3745-253 deactivated
DSPU: LU HOST3745-254 deactivated
DSPU: LS HOST3745 disconnected
DSPU: PU HOST3745 deactivated
```

Table 65 describes the significant fields shown in the display.

**Table 65** *debug dspu activation Field Descriptions*

Field	Description
DSPU	Downstream PU debugging message.
LS	Link station (LS) event triggered the message.
PU	PU event triggered the message.
LU	LU event triggered the message.
HOST3745	Host name or PU name.
HOST3745-253	Host name or PU name and the LU address, separated by a dash.
connected activated disconnected deactivated	Event that occurred to trigger the message.

**Related Commands**

Command	Description
<b>debug dspu packet</b>	Displays information on a DSPU packet.
<b>debug dspu state</b>	Displays information on DSPU FSM state changes.
<b>debug dspu trace</b>	Displays information on DSPU trace activity.

# debug dspu packet

To display information on a downstream physical unit (DSPU) packet, use the **debug dspu packet** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug dspu packet** [*name*]

**no debug dspu packet** [*name*]

## Syntax Description

*name* (Optional) The host or PU name designation.

## Command Modes

Privileged EXEC

## Usage Guidelines

The **debug dspu packet** command displays all DSPU packet data flowing through the router. To restrict the output to a specific host or physical unit (PU), include the host or PU *name* argument. You cannot turn off debugging output for an individual PU if that PU has not been named in the **debug dspu packet** command.

## Examples

The following is sample output from the **debug dspu packet** command:

```
Router# debug dspu packet

DSPU: Rx: PU HOST3745 data length 12 data:
      2D0003002BE16B80 000D0201
DSPU: Tx: PU HOST3745 data length 25 data:
      2D0000032BE1EB80 000D020100850000 000C060000010000 00
DSPU: Rx: PU HOST3745 data length 12 data:
      2D0004002BE26B80 000D0201
DSPU: Tx: PU HOST3745 data length 25 data:
      2D0000042BE2EB80 000D020100850000 000C060000010000 00
```

[Table 66](#) describes the significant fields shown in the display.

**Table 66** *debug dspu packet* Field Descriptions

Field	Description
DSPU: Rx:	Received frame (packet) from the remote PU to the router PU.
DSPU: Tx:	Transmitted frame (packet) from the router PU to the remote PU.
PU HOST3745	Host name or PU associated with the transmit or receive.
data length 12 data:	Number of bytes of data, followed by up to 128 bytes of displayed data.

## Related Commands

Command	Description
<b>debug drip event</b>	Displays debugging messages for DRiP packets.
<b>debug dspu state</b>	Displays information on DSPU FSM state changes.
<b>debug dspu trace</b>	Displays information on DSPU trace activity.

# debug dspu state

To display information on downstream physical unit (DSPU) finite state machine (FSM) state changes, use the **debug dspu state** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug dspu state** [*name*]

**no debug dspu state** [*name*]

## Syntax Description

*name* (Optional) The host or physical unit (PU) name designation.

## Command Modes

Privileged EXEC

## Usage Guidelines

Use the **debug dspu state** command to display only the FSM state changes. To see all FSM activity, use the **debug dspu trace** command. You cannot turn off debugging output for an individual PU if that PU has not been named in the **debug dspu state** command.

## Examples

The following is sample output from the **debug dspu state** command. Not all intermediate numbers are shown for the “activated” and “deactivated” logical unit (LU) address ranges.

```
Router# debug dspu state

DSPU: LS HOST3745: input=StartLs, Reset -> PendConOut
DSPU: LS HOST3745: input=ReqOpn.Cnf, PendConOut -> Xid
DSPU: LS HOST3745: input=Connect.Ind, Xid -> ConnIn
DSPU: LS HOST3745: input=Connected.Ind, ConnIn -> Connected
DSPU: PU HOST3745: input=Actpu, Reset -> Active
DSPU: LU HOST3745-2: input=uActlu, Reset -> upLuActive
DSPU: LU HOST3745-3: input=uActlu, Reset -> upLuActive
.
.
.
DSPU: LU HOST3745-253: input=uActlu, Reset -> upLuActive
DSPU: LU HOST3745-254: input=uActlu, Reset -> upLuActive

DSPU: LS HOST3745: input=PuStopped, Connected -> PendDisc
DSPU: LS HOST3745: input=Disc.Cnf, PendDisc -> PendClose
DSPU: PS HOST3745: input=Close.Cnf, PendClose -> Reset
DSPU: PU HOST3745: input=T2ResetPu, Active -> Reset
DSPU: LU HOST3745-2: input=uStopLu, upLuActive -> Reset
DSPU: LU HOST3745-3: input=uStopLu, upLuActive -> Reset
.
.
.
DSPU: LU HOST3745-253: input=uStopLu, upLuActive -> Reset
DSPU: LU HOST3745-254: input=uStopLu, upLuActive -> Reset
```

Table 67 describes the significant fields shown in the display.

**Table 67** *debug dspu state Field Descriptions*

Field	Description
DSPU	Downstream PU debug message.
LS	Link station (LS) event triggered the message.
PU	PU event triggered the message.
LU	LU event triggered the message.
HOST3745-253	Host name or PU name and LU address.
input=input,	Input received by the FSM.
previous-state, -> current-state	Previous state and current new state as seen by the FSM.

#### Related Commands

Command	Description
<b>debug drip event</b>	Displays debugging messages for DRiP packets.
<b>debug drip packet</b>	Displays information on DSPU packet.
<b>debug dspu trace</b>	Displays information on DSPU trace activity.

# debug dspu trace

To display information on downstream physical unit (DSPU) trace activity, which includes all finite state machine (FSM) activity, use the **debug dspu trace** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug dspu trace** [*name*]

**no debug dspu trace** [*name*]

## Syntax Description

*name* (Optional) The host or physical unit (PU) name designation.

## Command Modes

Privileged EXEC

## Usage Guidelines

Use the **debug dspu trace** command to display all FSM state changes. To see FSM state changes only, use the **debug dspu state** command. You cannot turn off debugging output for an individual PU if that PU has not been named in the **debug dspu trace** command.

## Examples

The following is sample output from the **debug dspu trace** command:

```
Router# debug dspu trace

DSPU: LS HOST3745 input = 0 ->(1,a1)
DSPU: LS HOST3745 input = 5 ->(5,a6)
DSPU: LS HOST3745 input = 7 ->(5,a9)
DSPU: LS HOST3745 input = 9 ->(5,a28)
DSPU: LU HOST3745-2 in:0 s:0->(2,a1)
DSPU: LS HOST3745 input = 19 ->(8,a20)
DSPU: LS HOST3745 input = 18 ->(8,a17)
DSPU: LU HOST3745-3 in:0 s:0->(2,a1)
DSPU: LS HOST3745 input = 19 ->(8,a20)
DSPU: LS HOST3745 input = 18 ->(8,a17)
DSPU: LU HOST3745-252 in:0 s:0->(2,a1)
DSPU: LS HOST3745 input = 19 ->(8,a20)
DSPU: LS HOST3745 input = 18 ->(8,a17)
DSPU: LU HOST3745-253 in:0 s:0->(2,a1)
DSPU: LS HOST3745 input = 19 ->(8,a20)
DSPU: LS HOST3745 input = 18 ->(8,a17)
DSPU: LU HOST3745-254 in:0 s:0->(2,a1)
DSPU: LS HOST3745 input = 19 ->(8,a20)
```

Table 68 describes significant fields shown in the output.

**Table 68** *debug dspu trace Field Descriptions*

Field	Description
7:23:57	Time stamp.
DSPU	Downstream PU debug message.
LS	Link station (LS) event triggered the message.
PU	A PU event triggered the message.
LU	LU event triggered the message.
HOST3745-253	Host name or PU name and LU address.
in:input s:state ->(new-state, action)	String describing the following: <ul style="list-style-type: none"> <li>input—LU FSM input</li> <li>state—Current FSM state</li> <li>new-state—New FSM state</li> <li>action—FSM action</li> </ul>
input=input -> (new-state, action)	String describing the following: <ul style="list-style-type: none"> <li>input—PU or LS FSM input</li> <li>new-state—New PU or LS FSM state</li> <li>action—PU or LS FSM action</li> </ul>

#### Related Commands

Command	Description
<b>debug drip event</b>	Displays debugging messages for DRiP packets.
<b>debug drip packet</b>	Displays information on DSPU packet.
<b>debug dspu state</b>	Displays information on DSPU FSM state changes.

# debug dss ipx event

To display debugging messages for route change events that affect Internetwork Packet Exchange (IPX) Multilayer Switching (MLS), use the **debug dss ipx event** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug dss ipx event**

**no debug dss ipx event**

**Syntax Description** This command has no arguments or keywords.

**Defaults** Debugging is not enabled.

**Command Modes** Privileged EXEC

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

**Examples** The following is sample output from the **debug dss ipx event** command:

```
Router# debug dss ipx event

DSS IPX events debugging is on

Router# configure terminal

Enter configuration commands, one per line. End with CNTL/Z.

Router(config)# interface vlan 22

Router(config-if)# ipx access-group 800 out

05:51:36:DSS-feature:dss_ipxcache_version():idb:NULL, reason:42,
prefix:0, mask:FFFFFFF
05:51:36:DSS-feature:dss_ipx_access_group():idb:Vlan22
05:51:36:DSS-feature:dss_ipx_access_list()
05:51:36:DSS-base 05:51:33.834 dss_ipx_invalidate_interface V122
05:51:36:DSS-base 05:51:33.834 dss_set_ipx_flowmask_reg 2
05:51:36:%IPX mls flowmask transition from 1 to 2 due to new status of
simple IPX access list on interfaces
```

Related Commands	Command	Description
	<b>debug mls rp</b>	Displays various MLS debugging elements.

# debug eap

To display information about Extensible Authentication Protocol (EAP), use the **debug eap** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug eap** {all | errors | events | packets | sm}

**no debug eap** {all | errors | events | packets | sm}

## Syntax Description

<b>all</b>	Turns on debugging for all EAP information.
<b>errors</b>	Displays information about EAP packet errors.
<b>events</b>	Displays information about EAP events.
<b>packets</b>	Displays EAP packet-related information.
<b>sm</b>	Displays EAP state machine transitions.

## Command Modes

Privileged EXEC

## Command History

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

## Examples

The following sample output from the **debug eap all** command shows all EAP information:

```
Router# debug eap all

*Apr  9 19:19:05.475: eapauth_start_timer: eap-ev:Starting txWhen timer 3(10.0.0.1)
*Apr  9 19:19:06.108: eapauth_post_msg_to_process: eap-ev:10.0.0.1: msg =
6(eventEapResponse)
*Apr  9 19:19:06.108:      eap_auth 10.0.0.1: during state eapauth_connecting, got event
4(eapResponse)
*Apr  9 19:19:06.108: @@@ eap_auth 10.0.0.1: eapauth_connecting -> eapauth_authenticating

*Apr  9 19:20:11.907: eap_parse_aaa_resp: eap-packets:10.0.0.1: AAA Resp Status =PASS
*Apr  9 19:20:11.907: eap_parse_aaa_resp: eap-packets:10.0.0.1: Status Query Timeout=30
*Apr  9 19:20:11.907: eap_dump_packet: eap-packets:
EAP code: 0x3 id: 0x8 length: 0x0004
*Apr  9 19:20:11.907: eap_parse_aaa_resp: eap-packets:10.0.0.1: Received MPPE_RECV_KEY
(32)
819C2BC0: 3E >
819C2BD0: 138EC4CB FCACC44F EA4F56C8 4853BA0F ..DK|,DOjOVHHS:.
819C2BE0: 9CB08871 B8ABC03A 3E97A96C EBF6DE .0.q8+@:>.)lkv^

*Apr  9 19:20:11.911:      eap_bend 10.0.0.1: during state eapbend_response, got event
2(eapBendSuccess)
*Apr  9 19:20:11.911: @@@ eap_bend 10.0.0.1: eapbend_response -> eapbend_success
*Apr  9 19:20:11.911:      eap_bend 10.0.0.1: idle during state eapbend_success
*Apr  9 19:20:11.911: @@@ eap_bend 10.0.0.1: eapbend_success -> eapbend_idle
*Apr  9 19:20:11.911:      eap_auth 10.0.0.1: during state eapauth_authenticating, got
event 10(eapSuccess)
```

---

**Related Commands**

<b>Command</b>	<b>Description</b>
<b>debug eou</b>	Displays information about EAPoUDP.

# debug eigrp fsm

To display debugging information about Enhanced Interior Gateway Routing Protocol (EIGRP) feasible successor metrics (FSM), use the **debug eigrp fsm** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug eigrp fsm**

**no debug eigrp fsm**

**Syntax Description** This command has no arguments or keywords.

**Defaults** No default behavior or values

**Command Modes** Privileged EXEC

**Usage Guidelines** This command helps you observe EIGRP feasible successor activity and to determine whether route updates are being installed and deleted by the routing process.

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

```
Router# debug eigrp fsm

DUAL: dual_rcvupdate(): 172.25.166.0 255.255.255.0 via 0.0.0.0 metric 750080/0
DUAL: Find FS for dest 172.25.166.0 255.255.255.0. FD is 4294967295, RD is 42949
67295 found
DUAL: RT installed 172.25.166.0 255.255.255.0 via 0.0.0.0
DUAL: dual_rcvupdate(): 192.168.4.0 255.255.255.0 via 0.0.0.0 metric 4294967295/
4294967295
DUAL: Find FS for dest 192.168.4.0 255.255.255.0. FD is 2249216, RD is 2249216
DUAL: 0.0.0.0 metric 4294967295/4294967295not found Dmin is 4294967295
DUAL: Dest 192.168.4.0 255.255.255.0 not entering active state.
DUAL: Removing dest 192.168.4.0 255.255.255.0, nexthop 0.0.0.0
DUAL: No routes. Flushing dest 192.168.4.0 255.255.255.0
```

In the first line, DUAL stands for diffusing update algorithm. It is the basic mechanism within EIGRP that makes the routing decisions. The next three fields are the Internet address and mask of the destination network and the address through which the update was received. The metric field shows the metric stored in the routing table and the metric advertised by the neighbor sending the information. If shown, the term “Metric... inaccessible” usually means that the neighbor router no longer has a route to the destination, or the destination is in a hold-down state.

In the following output, EIGRP is attempting to find a feasible successor for the destination. Feasible successors are part of the DUAL loop avoidance methods. The FD field contains more loop avoidance state information. The RD field is the reported distance, which is the metric used in update, query, or reply packets.

The indented line with the “not found” message means a feasible successor (FS) was not found for 192.168.4.0 and EIGRP must start a diffusing computation. This means it begins to actively probe (sends query packets about destination 192.168.4.0) the network looking for alternate paths to 192.164.4.0.

```
DUAL: Find FS for dest 192.168.4.0 255.255.255.0. FD is 2249216, RD is 2249216
DUAL: 0.0.0.0 metric 4294967295/4294967295not found Dmin is 4294967295
```

The following output indicates the route DUAL successfully installed into the routing table:

```
DUAL: RT installed 172.25.166.0 255.255.255.0 via 0.0.0.0
```

The following output shows that no routes to the destination were discovered and that the route information is being removed from the topology table:

```
DUAL: Dest 192.168.4.0 255.255.255.0 not entering active state.
DUAL: Removing dest 192.168.4.0 255.255.255.0, nexthop 0.0.0.0
DUAL: No routes. Flushing dest 192.168.4.0 255.255.255.0
```

# debug eigrp neighbor

To display neighbors discovered by the Enhanced Interior Gateway Routing Protocol (EIGRP), use the **debug eigrp neighbor** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug eigrp neighbor** [**siatimer**] [**static**]

**no debug eigrp neighbor** [**siatimer**] [**static**]

Syntax Description	Parameter	Description
	<b>siatimer</b>	(Optional) Stuck-in-active (SIA) timer messages.
	<b>static</b>	(Optional) Static routes.

**Defaults** Debugging for EIGRP neighbors is not enabled.

**Command Modes** Privileged EXEC

Command History	Release	Modification
	12.0(7)T	This command was introduced.

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

```
Router# debug eigrp neighbor static

EIGRP Static Neighbors debugging is on

Router# configure terminal

Router(config)# router eigrp 100

Router(config-router)# neighbor 10.1.1.1 e3/1

Router(config-router)#
22:40:07:EIGRP:Multicast Hello is disabled on Ethernet3/1!
22:40:07:EIGRP:Add new static nbr 10.1.1.1 to AS 100 Ethernet3/1

Router(config-router)# no neighbor 10.1.1.1 e3/1

Router(config-router)#
22:41:23:EIGRP:Static nbr 10.1.1.1 not in AS 100 Ethernet3/1 dynamic list
22:41:23:EIGRP>Delete static nbr 10.1.1.1 from AS 100 Ethernet3/1
22:41:23:EIGRP:Multicast Hello is enabled on Ethernet3/1!
```

**Related Commands**

<b>Command</b>	<b>Description</b>
<b>neighbor</b>	Defines a neighboring router with which to exchange routing information.
<b>show ip eigrp neighbors</b>	Displays EIGRP neighbors.

# debug eigrp nsf

To display nonstop forwarding (NSF) events in the console of the router, use the **debug eigrp nsf** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug eigrp nsf**

**no debug eigrp nsf**

**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 eigrp nsf** command displays NSF-specific events. This command can be issued on an NSF-capable or NSF-aware router.

**Examples** The following example enables Enhanced Interior Gateway Routing Protocol (EIGRP) NSF debugging:

```
Router# debug eigrp nsf
```

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

# debug eigrp packet

To display general debugging information, use the **debug eigrp packet** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug eigrp packet**

**no debug eigrp packet**

**Syntax Description** This command has no arguments or keywords.

**Defaults** No default behavior or values

**Command Modes** Privileged EXEC

**Usage Guidelines** If a communication session is closing when it should not be, an end-to-end connection problem can be the cause. The **debug eigrp packet** command is useful for analyzing the messages traveling between the local and remote hosts.

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

```
Router# debug eigrp packet

EIGRP: Sending HELLO on Ethernet0/1
        AS 109, Flags 0x0, Seq 0, Ack 0
EIGRP: Sending HELLO on Ethernet0/1
        AS 109, Flags 0x0, Seq 0, Ack 0
EIGRP: Sending HELLO on Ethernet0/1
        AS 109, Flags 0x0, Seq 0, Ack 0
EIGRP: Received UPDATE on Ethernet0/1 from 192.195.78.24,
        AS 109, Flags 0x1, Seq 1, Ack 0
EIGRP: Sending HELLO/ACK on Ethernet0/1 to 192.195.78.24,
        AS 109, Flags 0x0, Seq 0, Ack 1
EIGRP: Sending HELLO/ACK on Ethernet0/1 to 192.195.78.24,
        AS 109, Flags 0x0, Seq 0, Ack 1
EIGRP: Received UPDATE on Ethernet0/1 from 192.195.78.24,
        AS 109, Flags 0x0, Seq 2, Ack 0
```

The output shows transmission and receipt of Enhanced Interior Gateway Routing Protocol (EIGRP) packets. These packet types may be hello, update, request, query, or reply packets. The sequence and acknowledgment numbers used by the EIGRP reliable transport algorithm are shown in the output. Where applicable, the network-layer address of the neighboring router is also included.

Table 69 describes the significant fields shown in the display.

**Table 69** *debug eigrp packet Field Descriptions*

Field	Description
EIGRP:	EIGRP packet information.
AS n	Autonomous system number.
Flags <i>nxn</i>	<p>A flag of 1 means the sending router is indicating to the receiving router that this is the first packet it has sent to the receiver.</p> <p>A flag of 2 is a multicast that should be conditionally received by routers that have the conditionally receive (CR) bit set. This bit gets set when the sender of the multicast has previously sent a sequence packet explicitly telling it to set the CR bit.</p>
HELLO	<p>Hello packets are the neighbor discovery packets. They are used to determine whether neighbors are still alive. As long as neighbors receive the hello packets the router is sending, the neighbors validate the router and any routing information sent. If neighbors lose the hello packets, the receiving neighbors invalidate any routing information previously sent. Neighbors also send hello packets.</p>

# debug eigrp transmit

To display transmittal messages sent by the Enhanced Interior Gateway Routing Protocol (EIGRP), use the **debug eigrp transmit** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

```
debug eigrp transmit [ack] [build] [detail] [link] [packetize] [peerdown] [sia] [startup]
                    [strange]
```

```
no debug eigrp transmit [ack] [build] [detail] [link] [packetize] [peerdown] [sia] [startup]
                        [strange]
```

## Syntax Description

<b>ack</b>	(Optional) Information for acknowledgment (ACK) messages sent by the system.
<b>build</b>	(Optional) Build information messages (messages that indicate that a topology table was either successfully built or could not be built).
<b>detail</b>	(Optional) Additional detail for debug output.
<b>link</b>	(Optional) Information regarding topology table linked-list management.
<b>packetize</b>	(Optional) Information regarding topology table linked-list management.
<b>peerdown</b>	(Optional) Information regarding the impact on packet generation when a peer is down.
<b>sia</b>	(Optional) Stuck-in-active (SIA) messages.
<b>startup</b>	(Optional) Information regarding peer startup and initialization packets that have been transmitted.
<b>strange</b>	(Optional) Unusual events relating to packet processing.

## Defaults

Debugging for EIGRP transmittal messages is not enabled.

## Command Modes

Privileged EXEC

## Command History

Release	Modification
12.1	This command was introduced.

## Examples

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

```
Router# debug eigrp transmit

EIGRP Transmission Events debugging is on
  (ACK, PACKETIZE, STARTUP, PEERDOWN, LINK, BUILD, STRANGE, SIA, DETAIL)

Router# configure terminal

Enter configuration commands, one per line.  End with CNTL/Z.
Router#(config)# router eigrp 100
Router#(config-router)# network 10.4.9.0 0.0.0.255
Router#(config-router)#
5d22h: DNDB UPDATE 10.0.0.0/8, serno 0 to 1, refcount 0
Router#(config-router)#
```

# debug eou

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

```
debug eou {all | eap | errors | events | packets | ratelimit | sm}
```

```
no debug eou {all | eap | errors | events | packets | ratelimit | sm}
```

## Syntax Description

<b>all</b>	Displays all EAPoUDP information.
<b>eap</b>	Displays EAPoUDP packets.
<b>errors</b>	Displays information about EAPoUDP packet errors.
<b>events</b>	Displays information about EAPoUDP events.
<b>packets</b>	Displays EAPoUDP packet-related information.
<b>ratelimit</b>	Displays EAPoUDP posture-validation information.
<b>sm</b>	Displays EAPoUDP state machine transitions.

## Defaults

If you do not enter any keywords, debugging is turned on for all EAPoUDP messages.

## Command Modes

Privileged EXEC

## Command History

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

## Examples

The following sample output from the **debug eou all** command shows all EAPoUDP information:

```
Router# debug eou all

*Apr  9 19:30:40.782: eou-ev:EOU Init Validation for idb= FastEthernet0/0.420 src_mac=
0001.027c.f364 src_ip= 10.0.0.1
*Apr  9 19:30:40.786:      eou_auth 10.0.0.1: initial state eou_initialize has enter
*Apr  9 19:30:40.786: @@@ eou_auth 10.0.0.1: eou_initialize -> eou_hello
*Apr  9 19:30:40.786: eou-ev:eou_send_hello_request: Send Hello Request host= 10.0.0.15
eou_port= 5566 (hex)

*Apr  9 19:30:40.790: EAPoUDP (tx) Flags:0 Ver=1 opcode=2 Len=8 MsgId=3839857370 Assoc
ID=0
*Apr  9 19:30:40.790: Dumping TLV contents
*Apr  9 19:30:40.790: TLV M:1 R:0 Type=ASSOCIATION ID Length=4 Association=-1994800267
*Apr  9 19:30:40.999: EAPoUDP (rx) Flags:128 Ver=1 opcode=2 Len=24 MsgId=3839857370 Assoc
ID=2300167029
*Apr  9 19:30:40.999: Dumping TLV contents
*Apr  9 19:30:40.999: TLV M:1 R:0 Type=COOKIE PAYLOAD Length=12
07167CE0:      8919C375 259B6D41 5FEA5D27      ..Cu%.mA_jj'
07167CF0:
*Apr  9 19:30:40.999: TLV M:1 R:0 Type=ASSOCIATION ID Length=4 Association=1016688999
```

```
*Apr 9 19:31:50.048: @@@ eou_auth 10.0.0.1: eou_eap -> eou_eap
*Apr 9 19:31:50.048: eou-ev:10.0.0.1: msg = 24 (eventEouEapSuccess)
*Apr 9 19:31:50.048: eou_auth 10.0.0.1: during state eou_eap, got event
14 (eouEapSuccess)
*Apr 9 19:31:50.048: @@@ eou_auth 10.0.0.1: eou_eap -> eou_result
*Apr 9 19:31:50.052: eou-ev:Starting RESULT timer 3 (10.0.0.1)
```

**Related Commands**

<b>Command</b>	<b>Description</b>
<b>debug eap</b>	Displays information about EAP messages.
<b>debug ip admission eapudp</b>	Displays information about EAPoUDP network admission control events.

# debug ephone alarm

To set SkinnyStation alarm messages debugging for the Cisco IP phone, use the **debug ephone alarm** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug ephone alarm** [**mac-address** *mac-address*]

**no debug ephone alarm** [**mac-address** *mac-address*]

## Syntax Description

<b>mac-address</b>	(Optional) Defines the MAC address of the Cisco IP phone.
<i>mac-address</i>	(Optional) Specifies the MAC address of the Cisco IP phone.

## Defaults

No default behavior or values

## Command Modes

Privileged EXEC

## Command History

Release	Modification
12.2(2)XT	This command was introduced on the following platforms: Cisco 1750, Cisco 1751, Cisco 2600 series and Cisco 3600 series multiservice routers; and Cisco IAD2420 series Integrated Access Devices (IADs).
12.2(8)T	This command was implemented on the Cisco 3725 and Cisco 3745 routers.
12.2(8)T1	This command was implemented on the Cisco 2600-XM and Cisco 2691 routers.
12.2(11)T	This command was implemented on the Cisco 1760 routers.

## Usage Guidelines

The **debug ephone alarm** command shows all the SkinnyStation alarm messages sent by the Cisco IP phone. Under normal circumstances, this message is sent by the Cisco IP phone just before it registers, and the message has the severity level for the alarm set to “Informational” and contains the reason for the phone reboot or re-register. This type of message is entirely benign and does not indicate an error condition.

If the **mac-address** keyword is not used, the **debug ephone alarm** command debugs all Cisco IP phones that are registered to the router. You can remove debugging for the Cisco IP phones that you do not want to debug by using the **mac-address** keyword with the **no** form of this command.

You can enable or disable debugging on any number of Cisco IP phones. To see the Cisco IP phones that have debugging enabled, enter the **show ephone** command and look at the debug field in the output. When debugging is enabled for a Cisco IP phone, the debug output is displayed for the directory numbers associated with the Cisco IP phone.

**Examples**

The following example shows a SkinnyStation alarm message that is sent before the Cisco IP phone registers:

```
Router# debug ephone alarm

phone keypad reset
CM-closed-TCP
CM-bad-state
```

**Related Commands**

Command	Description
<b>debug ephone detail</b>	Sets detail debugging for the Cisco IP phone.
<b>debug ephone error</b>	Sets error debugging for the Cisco IP phone.
<b>debug ephone keepalive</b>	Sets keepalive debugging for the Cisco IP phone.
<b>debug ephone loopback</b>	Sets MWI debugging for the Cisco IP phone.
<b>debug ephone pak</b>	Provides voice packet level debugging and prints the contents of one voice packet in every 1024 voice packets.
<b>debug ephone raw</b>	Provides raw low-level protocol debugging display for all SCCP messages.
<b>debug ephone register</b>	Sets registration debugging for the Cisco IP phone.
<b>debug ephone state</b>	Sets state debugging for the Cisco IP phone.
<b>debug ephone statistics</b>	Sets statistics debugging for the Cisco IP phone.
<b>show debugging</b>	Displays information about the types of debugging that are enabled for your router.

# debug ephone ccm-compatible

To display Cisco CallManager notification updates for calls between Cisco CallManager and Cisco CallManager Express, use the **debug ephone ccm-compatible** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug ephone ccm-compatible** [*mac-address mac-address*]

**no debug ephone ccm-compatible** [*mac-address mac-address*]

## Syntax Description

**mac-address mac-address** (Optional) Specifies the MAC address of a Cisco IP phone for debugging.

## Command Modes

Privileged EXEC

## Command History

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

## Usage Guidelines

This command displays call flow notification information for all calls between Cisco CallManager and Cisco CallManager Express, but it is most useful for filtering out specific information for transfer and forward cases. For basic call information, use the **debug ephone state** command.

If you do not specify the **mac-address** keyword, the **debug ephone ccm-compatible** command debugs all Cisco IP phones that are registered to the router. You can remove debugging for the Cisco IP phones that you do not want to debug by using the **no** form of this command with the **mac-address** keyword.

Debugging can be enabled or disabled on any number of Cisco IP phones. Cisco IP phones that have debugging enabled are listed in the debug field of the **show ephone** command output. When debugging is enabled for a Cisco IP phone, debug output is displayed for all phone extensions (virtual voice ports) associated with that phone.

## Examples

The following sample output displays call flow notifications between Cisco CallManager and Cisco CallManager Express:

```
Router# debug ephone ccm-compatible

*May  1 04:30:02.650:ephone-2[2]:DtAlertingTone/DtHoldTone - mediaActive reset during
CONNECT
*May  1 04:30:02.654:ephone-2[2]:DtHoldTone - force media STOP state
*May  1 04:30:02.654://93/xxxxxxxxxxxx/CCAPI/ccCallNotify:(callID=0x5D,nData->
bitmask=0x00000007)
*May  1 04:30:02.654://93/xxxxxxxxxxxx/VTSP:(50/0/3):-1:0:5/vtsp_process_event:
vtsp:[50/0/3 (93), S_CONNECT, E_CC_SERVICE_MSG]
*May  1 04:30:02.654://93/xxxxxxxxxxxx/VTSP:(50/0/3):-1:0:5/act_service_msg_dow
n:.
*May  1 04:30:02.658:dn_callerid_update DN 3 number= 12009 name= CCM7960 in state
CONNECTED
*May  1 04:30:02.658:dn_callerid_update (incoming) DN 3 info updated to
*May  1 04:30:02.658:calling= 12009 called= 13003 origCalled=
```

```

*May 1 04:30:02.658:callingName= CCM7960, calledName= , redirectedTo =
*May 1 04:30:02.658:ephone-2[2][SEP003094C2999A]:refreshDisplayLine for line 1
  DN 3 chan 1
*May 1 04:30:03.318:ephone-2[2]:DisplayCallInfo incoming call
*May 1 04:30:03.318:ephone-2[2]:Call Info DN 3 line 1 ref 24 called 13003 calling 12009
origcalled 13003 calltype 1
*May 1 04:30:03.318:ephone-2[2]:Original Called Name UUT4PH3
*May 1 04:30:03.318:ephone-2[2]:CCM7960 calling
*May 1 04:30:03.318:ephone-2[2]:UUT4PH3

```

**Related Commands**

Command	Description
<b>debug ephone state</b>	Displays call state information.
<b>show debugging</b>	Displays information about the types of debugging that are enabled for your router.
<b>show ephone</b>	Displays information about registered Cisco IP phones.

# debug ephone detail

To set detail debugging for the Cisco IP phone, use the **debug ephone detail** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug ephone detail** [**mac-address** *mac-address*]

**no debug ephone detail** [**mac-address** *mac-address*]

## Syntax Description

<b>mac-address</b>	(Optional) Defines the MAC address of the Cisco IP phone.
<i>mac-address</i>	(Optional) Specifies the MAC address of the Cisco IP phone.

## Defaults

No default behavior or values

## Command Modes

Privileged EXEC

## Command History

Release	Modification
12.1(5)YD	This command was introduced on the following platforms: Cisco 2600 series and Cisco 3600 series multiservice routers, and Cisco IAD2420 series Integrated Access Devices (IADs).
12.2(2)XT	This command was implemented on the Cisco 1750 and Cisco 1751 multiservice routers.
12.2(8)T	This command was integrated into Cisco IOS Release 12.2(8)T and implemented on the Cisco 3725 and Cisco 3745 routers.
12.2(8)T1	This command was implemented on the Cisco 2600-XM and Cisco 2691 routers.
12.2(11)T	This command was implemented on the Cisco 1760 routers.

## Usage Guidelines

The **debug ephone detail** command includes the error and state levels.

If the **mac-address** keyword is not used, the **debug ephone detail** command debugs all Cisco IP phones that are registered to the router. You can remove debugging for the Cisco IP phones that you do not want to debug by using the **mac-address** keyword with the **no** form of this command.

You can enable or disable debugging on any number of Cisco IP phones. To see the Cisco IP phones that have debugging enabled, enter the **show ephone** command and look at the debug field in the output.

When debugging is enabled for a Cisco IP phone, the debug output is displayed for the directory numbers associated with the Cisco IP phone.

**Examples**

The following is sample output of detail debugging of the Cisco IP phone with MAC address 0030.94c3.8724. The sample is an excerpt of some of the activities that takes place during call setup, connected state, active call, and the call being disconnected.

```
Router# debug ephone detail mac-address 0030.94c3.8724
```

```
Ephone detail debugging is enabled

1d04h: ephone-1[1]:OFFHOOK
.
.
1d04h: Skinny Call State change for DN 1 SIEZE
.
.
1d04h: ephone-1[1]:SetCallState line 1 DN 1 TsOffHook
.
.
1d04h: ephone-1[1]:SetLineLamp 1 to ON
.
.
1d04h: ephone-1[1]:KeypadButtonMessage 5
.
.
1d04h: ephone-1[1]:KeypadButtonMessage 0
.
.
1d04h: ephone-1[1]:KeypadButtonMessage 0
.
.
1d04h: ephone-1[1]:KeypadButtonMessage 2
.
.
1d04h: ephone-1[1]:Store ReDial digit: 5002
.
SkinnyTryCall to 5002 instance 1
.
.
1d04h: ephone-1[1]:Store ReDial digit: 5002
1d04h: ephone-1[1]:
SkinnyTryCall to 5002 instance 1
.
.
1d04h: Skinny Call State change for DN 1 ALERTING
.
.
1d04h: ephone-1[1]:SetCallState line 1 DN 1 TsRingOut
.
.
1d04h: ephone-1[1]:SetLineLamp 1 to ON
1d04h: SetCallInfo calling dn 1 dn 1
calling [5001] called [5002]
.
.
1d04h: ephone-1[1]: Jane calling
1d04h: ephone-1[1]: Jill
.
.
1d04h: SkinnyUpdateDnState by EFXS_RING_GENERATE
for DN 2 to state RINGING
.
.
1d04h: SkinnyGetCallState for DN 2 CONNECTED
.
```

```

.
1d04h: ephone-1[1]:SetLineLamp 3 to ON
1d04h: ephone-1[1]:UpdateCallState DN 1 state 4 calleddn 2
.
.
1d04h: Skinny Call State change for DN 1 CONNECTED
.
.
1d04h: ephone-1[1]:OpenReceive DN 1 codec 4:G711Ulaw64k duration 10 ms bytes 80
.
.
1d04h: ephone-1[1]:OpenReceiveChannelAck 1.2.172.21 port=20180
1d04h: ephone-1[1]:Outgoing calling DN 1 Far-ephone-2 called DN 2
1d04h: SkinnyGetCallState for DN 1 CONNECTED
.
.
1d04h: ephone-1[1]:SetCallState line 3 DN 2 TsOnHook
.
.
1d04h: ephone-1[1]:SetLineLamp 3 to OFF
.
.
1d04h: ephone-1[1]:SetCallState line 1 DN 1 TsOnHook
.
.
1d04h: ephone-1[1]:Clean Up Speakerphone state
1d04h: ephone-1[1]:SpeakerPhoneOnHook
1d04h: ephone-1[1]:Clean up activeline 1
1d04h: ephone-1[1]:StopTone sent to ephone
1d04h: ephone-1[1]:Clean Up phone offhook state
1d04h: SkinnyGetCallState for DN 1 IDLE
1d04h: called DN -1, calling DN -1 phone -1
1d04h: ephone-1[1]:SetLineLamp 1 to OFF
1d04h: UnBinding ephone-1 from DN 1
1d04h: UnBinding called DN 2 from DN 1
1d04h: ephone-1[1]:ONHOOK
1d04h: ephone-1[1]:SpeakerPhoneOnHook
1d04h: ephone-1[1]:ONHOOK NO activeline
.
.
.

```

**Related Commands**

Command	Description
<b>debug ephone alarm</b>	Sets SkinnyStation alarm messages debugging for the Cisco IP phone.
<b>debug ephone error</b>	Sets error debugging for the Cisco IP phone.
<b>debug ephone keepalive</b>	Sets keepalive debugging for the Cisco IP phone.
<b>debug ephone loopback</b>	Sets MWI debugging for the Cisco IP phone.
<b>debug ephone pak</b>	Provides voice packet level debugging and prints the contents of one voice packet in every 1024 voice packets.
<b>debug ephone raw</b>	Provides raw low-level protocol debugging display for all SCCP messages.
<b>debug ephone register</b>	Sets registration debugging for the Cisco IP phone.
<b>debug ephone state</b>	Sets state debugging for the Cisco IP phone.

<b>Command</b>	<b>Description</b>
<b>debug ephone statistics</b>	Sets statistics debugging for the Cisco IP phone.
<b>show debugging</b>	Displays information about the types of debugging that are enabled for your router.

# debug ephone error

To set error debugging for the Cisco IP phone, use the **debug ephone error** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug ephone error** [**mac-address** *mac-address*]

**no debug ephone error** [**mac-address** *mac-address*]

## Syntax Description

<b>mac-address</b>	(Optional) Defines the MAC address of the Cisco IP phone.
<i>mac-address</i>	(Optional) Specifies the MAC address of the Cisco IP phone.

## Defaults

No default behavior or values

## Command Modes

Privileged EXEC

## Command History

Release	Modification
12.1(5)YD	This command was introduced on the following platforms: Cisco 2600 series and Cisco 3600 series multiservice routers, and Cisco IAD2420 series Integrated Access Devices (IADs).
12.2(2)XT	This command was implemented on the Cisco 1750 and Cisco 1751 multiservice routers.
12.2(8)T	This command was integrated into Cisco IOS Release 12.2(8)T and implemented on the Cisco 3725 and Cisco 3745 routers.
12.2(8)T1	This command was implemented on the Cisco 2600-XM and Cisco 2691 routers.
12.2(11)T	This command was implemented on the Cisco 1760 routers.

## Usage Guidelines

The **debug ephone error** command cancels debugging at the detail and state level.

If the **mac-address** keyword is not used, the **debug ephone error** command debugs all Cisco IP phones that are registered to the router. You can remove debugging for the Cisco IP phones that you do not want to debug by using the **mac-address** keyword with the **no** form of this command.

You can enable or disable debugging on any number of Cisco IP phones. To see the Cisco IP phones that have debugging enabled, enter the **show ephone** command and look at the debug field in the output.

When debugging is enabled for a Cisco IP phone, the debug output is displayed for the directory numbers associated with the Cisco IP phone.

**Examples**

The following is sample output of error debugging for the Cisco IP phone with MAC address 0030.94c3.8724:

```
Router# debug ephone error mac-address 0030.94c3.8724
```

```
EPHONE error debugging is enabled
```

```
socket [2] send ERROR 11
```

```
Skinny Socket [2] retry failure
```

**Related Commands**

Command	Description
<b>debug ephone alarm</b>	Sets SkinnyStation alarm messages debugging for the Cisco IP phone.
<b>debug ephone detail</b>	Sets detail debugging for the Cisco IP phone.
<b>debug ephone keepalive</b>	Sets keepalive debugging for the Cisco IP phone.
<b>debug ephone loopback</b>	Sets MWI debugging for the Cisco IP phone.
<b>debug ephone pak</b>	Provides voice packet level debugging and prints the contents of one voice packet in every 1024 voice packets.
<b>debug ephone raw</b>	Provides raw low-level protocol debugging display for all SCCP messages.
<b>debug ephone register</b>	Sets registration debugging for the Cisco IP phone.
<b>debug ephone state</b>	Sets state debugging for the Cisco IP phone.
<b>debug ephone statistics</b>	Sets statistics debugging for the Cisco IP phone.
<b>show debugging</b>	Displays information about the types of debugging that are enabled for your router.

# debug ephone keepalive

To set keepalive debugging for the Cisco IP phone, use the **debug ephone keepalive** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug ephone keepalive** [**mac-address** *mac-address*]

**no debug ephone keepalive** [**mac-address** *mac-address*]

## Syntax Description

<b>mac-address</b>	(Optional) Defines the MAC address of the Cisco IP phone.
<i>mac-address</i>	(Optional) Specifies the MAC address of the Cisco IP phone.

## Defaults

No default behavior or values

## Command Modes

Privileged EXEC

## Command History

Release	Modification
12.1(5)YD	This command was introduced on the following platforms: Cisco 2600 series and Cisco 3600 series multiservice routers, and Cisco IAD2420 series Integrated Access Devices (IADs).
12.2(2)XT	This command was implemented on the Cisco 1750 and Cisco 1751 multiservice routers.
12.2(8)T	This command was implemented on the Cisco 3725 and Cisco 3745 routers.
12.2(8)T1	This command was implemented on the Cisco 2600-XM and Cisco 2691 routers.
12.2(11)T	This command was implemented on the Cisco 1760 routers.

## Usage Guidelines

The **debug ephone keepalive** command sets keepalive debugging.

If the **mac-address** keyword is not used, the **debug ephone keepalive** command debugs all Cisco IP phones that are registered to the router. You can remove debugging for the Cisco IP phones that you do not want to debug by using the **mac-address** keyword with the **no** form of this command.

You can enable or disable debugging on any number of Cisco IP phones. To see the Cisco IP phones that have debugging enabled, enter the **show ephone** command and look at the debug field in the output.

When debugging is enabled for a Cisco IP phone, the debug output is displayed for the directory numbers associated with the Cisco IP phone.

**Examples**

The following is sample output of the keepalive status for the Cisco IP phone with MAC address 0030.94C3.E1A8:

```
Router# debug ephone keepalive mac-address 0030.94c3.E1A8

EPHONE keepalive debugging is enabled for phone 0030.94C3.E1A8

1d05h: ephone-1 Set interface FastEthernet0/0 ETHERNET
1d05h: ephone-1[1]:Keepalive socket[1] SEP003094C3E1A8
1d05h: ephone-1 Set interface FastEthernet0/0 ETHERNET
1d05h: ephone-1[1]:Keepalive socket[1] SEP003094C3E1A8
1d05h: Skinny Checking for stale sockets
1d05h: ephone-1 Set interface FastEthernet0/0 ETHERNET
1d05h: ephone-1[1]:Keepalive socket[1] SEP003094C3E1A8
1d05h: ephone-1 Set interface FastEthernet0/0 ETHERNET
1d05h: ephone-1[1]:Keepalive socket[1] SEP003094C3E1A8
1d05h: Skinny active socket list (3/96): 1 2 4
```

**Related Commands**

Command	Description
<b>debug ephone alarm</b>	Sets SkinnyStation alarm messages debugging for the Cisco IP phone.
<b>debug ephone detail</b>	Sets detail debugging for the Cisco IP phone.
<b>debug ephone error</b>	Sets error debugging for the Cisco IP phone.
<b>debug ephone loopback</b>	Sets MWI debugging for the Cisco IP phone.
<b>debug ephone pak</b>	Provides voice packet level debugging and prints the contents of one voice packet in every 1024 voice packets.
<b>debug ephone raw</b>	Provides raw low-level protocol debugging display for all SCCP messages.
<b>debug ephone register</b>	Sets registration debugging for the Cisco IP phone.
<b>debug ephone state</b>	Sets state debugging for the Cisco IP phone.
<b>debug ephone statistics</b>	Sets statistics debugging for the Cisco IP phone.
<b>show debugging</b>	Displays information about the types of debugging that are enabled for your router.

# debug ephone loopback

To set debugging for loopback calls, use the **debug ephone loopback** command in privileged EXEC mode. To disable debugging, use the **no** form of this command.

```
debug ephone loopback [mac-address mac-address]
```

```
no debug ephone loopback [mac-address mac-address]
```

## Syntax Description

**mac-address** *mac-address* (Optional) Specifies the MAC address of a Cisco IP phone for debugging.

## Command Modes

Privileged EXEC

## Command History

Release	Modification
12.2(2)XT	This command was introduced for Cisco IOS Telephony Services (now known as Cisco CallManager Express) Version 2.0 on the Cisco 1750, Cisco 1751, Cisco 2600 series, Cisco 3600 series, and Cisco IAD2420 series.
12.2(8)T	This command was integrated into Cisco IOS Release 12.2(8)T and implemented on the Cisco 3725 and Cisco 3745.
12.2(8)T1	This command was implemented on the Cisco 2600-XM and Cisco 2691.
12.3(4)T	This command was integrated into Cisco IOS Release 12.3(4)T.

## Usage Guidelines

The **debug ephone loopback** command sets debugging for incoming and outgoing calls on all loopback-dn pairs or on the single loopback-dn pair that is associated with the IP phone that has the MAC address specified in this command.

If you enable the **debug ephone loopback** command and the **debug ephone pak** command at the same time, the output displays packet debug output for the voice packets that are passing through the loopback-dn pair.

You can enable or disable debugging on any number of Cisco IP phones. To see the Cisco IP phones that have debugging enabled, enter the **show ephone** command and look at the debug field in the output. When debugging is enabled for a Cisco IP phone, the debug output is displayed for the directory numbers associated with that Cisco IP phone.

## Examples

The following example contains two excerpts of output for a call that is routed through a loopback. The first excerpt is output from the **show running-config** command and displays the loopback configuration used for this example. The second excerpt is output from the **debug ephone loopback** command.

```
Router# show running-config
.
.
.
```

```

ephone-dn 14
  number 1514
!
!
ephone-dn 42
  number 17181..
  loopback-dn 43 forward 4
  no huntstop
!
!
ephone-dn 43
  number 19115..
  loopback-dn 42 forward 4
!
.
.
.

```

A loopback call is started. An incoming call to 1911514 (ephone-dn 43) uses the loopback pair of ephone-dns to become an outgoing call to extension 1514. The number in the outgoing call has only four digits because the **loopback-dn** command specifies forwarding of four digits. The outgoing call uses ephone-dn 42, which is also specified in the **loopback-dn** command under ephone-dn 43. When the extension at 1514 rings, the following debug output is displayed:

```
Router# debug ephone loopback
```

```

Mar 7 00:57:25.376:Pass processed call info to special DN 43 chan 1
Mar 7 00:57:25.376:SkinnySetCallInfoLoopback DN 43 state IDLE to DN 42 state IDLE
Mar 7 00:57:25.376:Called Number = 1911514 Called Name =
Mar 7 00:57:25.376:Calling Number = 8101 Calling Name =
  orig Called Number =
Copy Caller-ID info from Loopback DN 43 to DN 42
Mar 7 00:57:25.376:DN 43 Forward 1514
Mar 7 00:57:25.376:PredictTarget match 1514 DN 14 is idle
Mar 7 00:57:25.380:SkinnyUpdateLoopbackState DN 43 state RINGING calledDn -1
Mar 7 00:57:25.380:Loopback DN 42 state IDLE
Mar 7 00:57:25.380:Loopback DN 43 calledDN -1 callingDn -1 G711Ulaw64k
Mar 7 00:57:25.380:SkinnyUpdateLoopbackState DN 43 to DN 42 signal OFFHOOK
Mar 7 00:57:25.380:SetDnCodec Loopback DN 43 codec 4:G711Ulaw64k vad 0 size 160
Mar 7 00:57:25.380:SkinnyDnToneLoopback DN 42 state SIEZE to DN 43 state RINGING
Mar 7 00:57:25.380:TONE ON DtInsideDialTone
Mar 7 00:57:25.380:SkinnyDnToneLoopback called number = 1911514
Mar 7 00:57:25.380:DN 43 Forward 1514
Mar 7 00:57:25.380:DN 42 from 43 Dial 1514
Mar 7 00:57:25.384:SkinnyDnToneLoopback DN 42 state ALERTING to DN 43 state RINGING
Mar 7 00:57:25.384:TONE OFF
Mar 7 00:57:25.384:SkinnyDnToneLoopback DN 42 state ALERTING to DN 43 state RINGING
Mar 7 00:57:25.384:SkinnyUpdateLoopbackState DN 42 state ALERTING calledDn -1
Mar 7 00:57:25.384:Loopback DN 43 state RINGING
Mar 7 00:57:25.384:Loopback Alerting DN 42 calledDN -1 callingDn -1 G711Ulaw64k
Mar 7 00:57:25.388:ephone-5[7]:DisplayCallInfo incoming call
Mar 7 00:57:25.388:SkinnyDnToneLoopback DN 42 state ALERTING to DN 43 state RINGING
Mar 7 00:57:25.388:TONE ON DtAlertingTone
Mar 7 00:57:25.388:SkinnyDnToneLoopback DN 42 to DN 43 deferred alerting by
DtAlertingTone
Mar 7 00:57:25.388:EFXS_STATE_ONHOOK_RINGING already done for DN 43 chan 1
Mar 7 00:57:25.388:Set prog_ind 0 for DN 42 chan 1
.
.
.

```

When extension 1514 answers the call, the following debug output is displayed:

```
.
.
.
Mar 7 00:57:32.158:SkinnyDnToneLoopback DN 42 state ALERTING to DN 43 state RINGING
Mar 7 00:57:32.158:TONE OFF
Mar 7 00:57:32.158:dn_support_g729 true DN 42 chan 1 (loopback)
Mar 7 00:57:32.158:SetDnCodec Loopback DN 43 codec 4:G711Ulaw64k vad 0 size 160
Mar 7 00:57:32.158:SkinnyUpdateLoopbackState DN 42 state CALL_START calledDn 14
Mar 7 00:57:32.158:Loopback DN 43 state RINGING
Mar 7 00:57:32.158:SkinnyUpdateLoopbackState DN 42 to DN 43 deferred alerting by
CALL_START already sent
Mar 7 00:57:32.158:SetDnCodec reassert defer_start for DN 14 chan 1
Mar 7 00:57:32.158:Delay media until loopback DN 43 is ready
Mar 7 00:57:32.158:SkinnyUpdateLoopbackCodec check for DN 14 chan 1 from DN 42 loopback
DN 43
Mar 7 00:57:32.158:SkinnyUpdateLoopbackCodec DN chain is 14 1, other=42, lb=43, far=-1 1,
final=43 1
Mar 7 00:57:32.158:SkinnyUpdateLoopbackCodec DN 14 chan 1 DN 43 chan 1 codec 4 match
Mar 7 00:57:32.162:SkinnyUpdateLoopbackState DN 42 state CONNECTED calledDn 14
Mar 7 00:57:32.162:Loopback DN 43 state RINGING
Mar 7 00:57:32.162:SkinnyUpdateLoopbackState DN 42 to DN 43 signal ANSWER
Mar 7 00:57:32.162:Loopback DN 42 calledDN 14 callingDn -1 G711Ulaw64k
Mar 7 00:57:32.162:Loopback DN 43 calledDN -1 callingDn -1 incoming G711Ulaw64k
Mar 7 00:57:32.162:ephone-5[7][SEP000DBDBEF37D]:refreshDisplayLine for line 1 DN 14 chan
1
Mar 7 00:57:32.162:dn_support_g729 true DN 43 chan 1 (loopback)
Mar 7 00:57:32.162:SetDnCodec Loopback DN 42 codec 4:G711Ulaw64k vad 0 size 160
Mar 7 00:57:32.162:SkinnyUpdateLoopbackState DN 43 state CALL_START calledDn -1
Mar 7 00:57:32.162:Loopback DN 42 state CONNECTED
Mar 7 00:57:32.162:SkinnyUpdateLoopbackState DN 43 has defer_dn 14 chan 1 set
Mar 7 00:57:32.162:SkinnyUpdateLoopbackState DN 43 has defer_dn 14 chan 1:
  -invoke SkinnyOpenReceive
Mar 7 00:57:32.162:SkinnyUpdateLoopbackCodec check for DN 14 chan 1 from DN 42 loopback
DN 43
Mar 7 00:57:32.162:SkinnyUpdateLoopbackCodec DN chain is 14 1, other=42, lb=43, far=-1 1,
final=43 1
Mar 7 00:57:32.162:SkinnyUpdateLoopbackCodec DN 14 chan 1 DN 43 chan 1 codec 4 match
Mar 7 00:57:32.162:SkinnyUpdateLoopbackState DN 43 state CALL_START calledDn -1
Mar 7 00:57:32.162:Loopback DN 42 state CONNECTED
Mar 7 00:57:32.454:SkinnyGetDnAddrInfo DN 43 LOOPBACK
update media address to 10.0.0.6 25390 from DN 14
Mar 7 00:57:33.166:ephone-5[7]:DisplayCallInfo incoming call
.
.
.
```

When the called extension, 1514, goes back on-hook, the following debug output is displayed:

```
.
.
.
Mar 7 00:57:39.224:Loopback DN 42 disc reason 16 normal state CONNECTED
Mar 7 00:57:39.224:SkinnyUpdateLoopbackState DN 42 state CALL_END calledDn -1
Mar 7 00:57:39.224:Loopback DN 43 state CONNECTED
Mar 7 00:57:39.224:SkinnyUpdateLoopbackState DN 42 to DN 43 signal ONHOOK
Mar 7 00:57:39.236:SkinnyDnToneLoopback DN 42 state IDLE to DN 43 state IDLE
Mar 7 00:57:39.236:TONE OFF
Mar 7 00:57:39.236:SkinnyDnToneLoopback DN 43 state IDLE to DN 42 state IDLE
Mar 7 00:57:39.236:TONE OFF
```

Table 70 describes the significant fields shown in the display.

**Table 70** *debug ephone loopback Field Descriptions*

Field	Description
Called Number	Original called number as presented to the incoming side of the loopback-dn.
Forward	Outgoing number that is expected to be dialed by the outgoing side of the loopback-dn pair.
PredictTarget Match	Extension (ephone-dn) that is anticipated by the loopback-dn to be the far-end termination for the call.
signal OFFHOOK	Indicates that the outgoing side of the loopback-dn pair is going off-hook prior to placing the outbound call leg.
Dial	Outbound side of the loopback-dn that is actually dialing the outbound call leg.
deferred alerting	Indicates that the alerting, or ringing, tone is returning to the original inbound call leg in response to the far-end ephone-dn state.
DN chain	Chain of ephone-dns that has been detected, starting from the far-end that terminates the call. Each entry in the chain indicates an ephone-dn tag and channel number. Entries appear in the following order, from left to right: <ul style="list-style-type: none"> <li>• Ephone-dn tag and channel of the far-end call terminator (in this example, ephone-dn 14 is extension 1514).</li> <li>• other—Ephone-dn tag of the outgoing side of the loopback.</li> <li>• lb—Ephone-dn tag of the incoming side of the loopback.</li> <li>• far—Ephone-dn tag and channel of the far-end call originator, or -1 for a nonlocal number.</li> <li>• final—Ephone-dn tag for the originator of the call on the incoming side of the loopback. If the originator is not a local ephone-dn, this is set to -1. This number represents the final ephone-dn tag in the chain, looking toward the originator.</li> </ul>
codec match	Indicates that there is no codec conflict between the two calls on either side of the loopback-dn.
GetDnAddrInfo	IP address of the IP phone at the final destination extension (ephone-dn), after resolving the chain of ephone-dns involved.
disc_reason	Disconnect cause code, in decimal. These are normal CC_CAUSE code values that are also used in call control API debugging. Common cause codes include the following: <ul style="list-style-type: none"> <li>• 16—Normal disconnect.</li> <li>• 17—User busy.</li> <li>• 19—No answer.</li> <li>• 28—Invalid number.</li> </ul>

**Related Commands**

<b>Command</b>	<b>Description</b>
<b>debug ephone pak</b>	Provides voice packet level debugging.
<b>loopback-dn</b>	Configures loopback-dn virtual loopback voice ports used to establish demarcation points for VoIP voice calls and supplementary services.
<b>show ephone</b>	Displays information about registered Cisco IP phones.
<b>show ephone-dn loopback</b>	Displays information for ephone-dns that have been set up for loopback calls.

# debug ephone moh

To set debugging for music on hold (MOH), use the **debug ephone moh** command in privileged EXEC mode. To disable debugging, use the **no** form of this command.

**debug ephone moh** [**mac-address** *mac-address*]

**no debug ephone moh** [**mac-address** *mac-address*]

Syntax Description	<b>mac-address</b> <i>mac-address</i>	(Optional) Specifies the MAC address of a Cisco IP phone for debugging.
--------------------	---------------------------------------	---

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

Command History	Release	Modification
	12.2(2)XT	This command was introduced for Cisco IOS Telephony Services (now known as Cisco CallManager Express) Version 2.0 and Cisco Survivable Remote Site Telephony (SRST) Version 2.0 on the Cisco 1750, Cisco 1751, Cisco 2600 series, Cisco 3600 series, and Cisco IAD2420 series.
	12.2(8)T	This command was integrated into Cisco IOS Release 12.2(8)T and implemented on the Cisco 3725 and Cisco 3745.
	12.2(8)T1	This command was implemented on the Cisco 2600-XM and Cisco 2691.
	12.3(4)T	This command was integrated into Cisco IOS Release 12.3(4)T.

**Usage Guidelines** Always use the **no moh** command before modifying or replacing the MOH file in Flash memory.

When a configuration using the **multicast moh** command is used and the **debug ephone moh** command is enabled, if you delete or modify the MOH file in the router's Flash memory, the debug output can be excessive and can flood the console. The multicast MOH configuration should be removed before using the **no moh** command when the **debug ephone moh** command is enabled.

**Examples** The following sample output shows MOH activity prior to the first MOH session. Note that if you enable multicast MOH, that counts as the first session.

```
Router# debug ephone moh

Mar  7 00:52:33.817:MOH AU file
Mar  7 00:52:33.817:skinny_open_moh_play set type to 3
Mar  7 00:52:33.825: 2E73 6E64 0000 0018 0007 3CCA 0000 0001
Mar  7 00:52:33.825: 0000 1F40 0000 0001 FFFF FFFF FFFF FFFF
Mar  7 00:52:33.825: FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF
Mar  7 00:52:33.825: FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF
Mar  7 00:52:33.825: FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF
Mar  7 00:52:33.825: FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF
Mar  7 00:52:33.825: FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF
```

```

Mar 7 00:52:33.825: FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF
Mar 7 00:52:33.825: FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF
Mar 7 00:52:33.825: FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF
Mar 7 00:52:33.825: FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF
Mar 7 00:52:33.825: FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF
Mar 7 00:52:33.825: FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF
Mar 7 00:52:33.825: FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF
Mar 7 00:52:33.825: FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF
Mar 7 00:52:33.825: FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF
Mar 7 00:52:33.825:
Mar 7 00:52:33.825:AU file processing Found .snd
Mar 7 00:52:33.825:AU file data start at 24 end at 474338
Mar 7 00:52:33.825:AU file codec Media_Payload_G711Ulaw64k
Mar 7 00:52:33.825:MOH read file header type AU start 24 end 474338
Mar 7 00:52:33.825:MOH pre-read block 0 at write-offset 0 from 24
Mar 7 00:52:33.833:MOH pre-read block 1 at write-offset 8000 from 8024
Mar 7 00:52:33.845:Starting read server with play-offset 0 write-offset 16000

```

Table 71 describes the significant fields shown in the display.

**Table 71 debug ephone moh Field Descriptions**

Field	Description
type	0—invalid 1—raw file 2—wave format file (.wav) 3—AU format (.au) 4—live feed
AU file processing Found .snd	A .snd header was located in the AU file.
AU file data start at, end at	Data start and end file offset within the MOH file, as indicated by the file header.
read file header type	File format found (AU, WAVE, or RAW).
pre-read block, write-offset	Location in the internal MOH buffer to which data is being written, and location from which that data was read in the file.
play-offset, write-offset	Indicates the relative positioning of MOH file read-ahead buffering. Data is normally written from a Flash file into the internal circular buffer, ahead of the location from which data is being played or output.

**Related Commands**

Command	Description
<b>moh</b> <b>(telephony-service)</b>	Generates an audio stream from a file for MOH in a Cisco CME system.
<b>multicast moh</b>	Uses the MOH audio stream as a multicast source in a Cisco CME system.

# debug ephone mwi

To set message waiting indication (MWI) debugging for the Cisco IOS Telephony Service router, use the **debug ephone mwi** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug ephone mwi**

**no debug ephone mwi**

## Syntax Description

This command has no arguments or keywords.

## Defaults

No default behavior or values

## Command Modes

Privileged EXEC

## Command History

Release	Modification
12.2(2)XT	This command was introduced on the following platforms: Cisco 1750, Cisco 1751, Cisco 2600 series and Cisco 3600 series multiservice routers; and Cisco IAD2420 series Integrated Access Devices (IADs).
12.2(8)T	This command was implemented on the Cisco 3725 and Cisco 3745 routers.
12.2(8)T1	This command was implemented on the Cisco 2600-XM and Cisco 2691 routers.
12.2(11)T	This command was implemented on the Cisco 1760 routers.

## Usage Guidelines

The **debug ephone mwi** command sets message waiting indication debugging for the Cisco IOS Telephony Service router. Because the MWI protocol activity is not specific to any individual Cisco IP phone, setting the MAC address keyword qualifier for this command is not useful.



### Note

Unlike the other related **debug ephone** commands, the **mac-address** keyword does not help debug a particular Cisco IP phone.

## Examples

The following is sample output of the message waiting indication status for the Cisco IOS Telephony Service router:

```
Router# debug ephone mwi
```

**Related Commands**

<b>Command</b>	<b>Description</b>
<b>debug ephone alarm</b>	Sets SkinnyStation alarm messages debugging for the Cisco IP phone.
<b>debug ephone detail</b>	Sets detail debugging for the Cisco IP phone.
<b>debug ephone error</b>	Sets error debugging for the Cisco IP phone.
<b>debug ephone keepalive</b>	Sets keepalive debugging for the Cisco IP phone.
<b>debug ephone pak</b>	Provides voice packet level debugging and prints the contents of one voice packet in every 1024 voice packets.
<b>debug ephone raw</b>	Provides raw low-level protocol debugging display for all SCCP messages.
<b>debug ephone register</b>	Sets registration debugging for the Cisco IP phone.
<b>debug ephone state</b>	Sets state debugging for the Cisco IP phone.
<b>debug ephone statistics</b>	Sets statistics debugging for the Cisco IP phone.
<b>show debugging</b>	Displays information about the types of debugging that are enabled for your router.

# debug ephone pak

To provide voice packet level debugging and to print the contents of one voice packet in every 1024 voice packets, use the **debug ephone pak** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug ephone pak** [**mac-address** *mac-address*]

**no debug ephone pak** [**mac-address** *mac-address*]

## Syntax Description

<b>mac-address</b>	(Optional) Defines the MAC address of the Cisco IP phone.
<i>mac-address</i>	(Optional) Specifies the MAC address of the Cisco IP phone.

## Defaults

No default behavior or values

## Command Modes

Privileged EXEC

## Command History

Release	Modification
12.1(5)YD	This command was introduced on the following platforms: Cisco 2600 series and Cisco 3600 series multiservice routers, and Cisco IAD2420 series Integrated Access Devices (IADs).
12.2(2)XT	This command was implemented on the Cisco 1750 and Cisco 1751 multiservice routers.
12.2(8)T	This command was integrated into Cisco IOS Release 12.2(8)T and implemented on the Cisco 3725 and Cisco 3745 routers.
12.2(8)T1	This command was implemented on the Cisco 2600-XM and Cisco 2691 routers.
12.2(11)T	This command was implemented on the Cisco 1760 routers.

## Usage Guidelines

The **debug ephone pak** command provides voice packet level debugging and prints the contents of one voice packet in every 1024 voice packets.

If the **mac-address** keyword is not used, the **debug ephone pak** command debugs all Cisco IP phones that are registered to the router. You can remove debugging for the Cisco IP phones that you do not want to debug by using the **mac-address** keyword with the **no** form of this command.

You can enable or disable debugging on any number of Cisco IP phones. To see the Cisco IP phones that have debugging enabled, enter the **show ephone** command and look at the debug field in the output. When debugging is enabled for a Cisco IP phone, the debug output is displayed for the directory numbers associated with the Cisco IP phone.

**Examples**

The following is sample output of packet debugging for the Cisco IP phone with MAC address 0030.94c3.8724:

```
Router# debug ephone pak mac-address 0030.94c3.8724

EPHONE packet debugging is enabled for phone 0030.94c3.8724

01:29:14: ***ph_xmit_ephone DN 3 tx_pkts 5770 dest=10.2.1.1 orig len=32
  pakcopy=0 discards 27 ip_enctype 0 0 last discard: unsupported payload type
01:29:14: to_skinny_duration 130210 offset -30 last -40 seq 0 adj 0
01:29:14: IP: 45B8 003C 0866 0000 3F11 3F90 2800 0001 0A02 0101
01:29:14: TTL 63 TOS B8 prec 5
01:29:14: UDP: 07D0 6266 0028 0000
01:29:14: sport 2000 dport 25190 length 40 checksum 0
01:29:14: RTP: 8012 16AF 9170 6409 0E9F 0001
01:29:14: is_rtp:1 is_frfl1:0 vlen:0 delta_t:160 vofr1:0 vofr2:0
scodec:11 rtp_bits:8012 rtp_codec:18 last_bad_payload 19
01:29:14: vencap FAILED
01:29:14: PROCESS SWITCH
01:29:15: %SYS-5-CONFIG-I: Configured from console by console
01:29:34: ***SkinnyPktIp DN 3 10.2.1.1 to 40.0.0.1 pkts 4880 FAST sw
01:29:34: from_skinny_duration 150910
01:29:34: nw 3BBC2A8 addr 3BBC2A4 mac 3BBC2A4 dg 3BBC2C4 dgs 2A
01:29:34: MAC: 1841 0800
01:29:34: IP: 45B8 0046 682E 0000 3E11 E0BD 0A02 0101 2800 0001
01:29:34: TTL 62 TOS B8 prec 5
01:29:34: UDP: 6266 07D0 0032 0000
01:29:34: sport 25190 dport 2000 length 50 checksum 0
01:29:34: RTP: 8012 55FF 0057 8870 3AF4 C394
01:29:34: RTP: rtp_bits 8012 seq 55FF ts 578870 ssrc 3AF4C394
01:29:34: PAYLOAD:
01:29:34: 1409 37C9 54DE 449C 3B42 0446 3AAB 182E
01:29:34: 56BC 5184 58E5 56D3 13BE 44A7 B8C4
01:29:34:
01:29:37: ***ph_xmit_ephone DN 3 tx_pkts 6790 dest=10.2.1.1 orig len=32
  pakcopy=0 discards 31 ip_enctype 0 0 last discard: unsupported payload type
01:29:37: to_skinny_duration 153870 offset -150 last -40 seq 0 adj 0
01:29:37: IP: 45B8 003C 0875 0000 3F11 3F81 2800 0001 0A02 0101
01:29:37: TTL 63 TOS B8 prec 5
01:29:37: UDP: 07D0 6266 0028 0000
01:29:37: sport 2000 dport 25190 length 40 checksum 0
01:29:37: RTP: 8012 1AAF 9173 4769 0E9F 0001
01:29:37: is_rtp:1 is_frfl1:0 vlen:0 delta_t:160 vofr1:0 vofr2:0
```

**Related Commands**

Command	Description
<b>debug ephone alarm</b>	Sets SkinnyStation alarm messages debugging for the Cisco IP phone.
<b>debug ephone detail</b>	Sets detail debugging for the Cisco IP phone.
<b>debug ephone error</b>	Sets error debugging for the Cisco IP phone.
<b>debug ephone keepalive</b>	Sets keepalive debugging for the Cisco IP phone.
<b>debug ephone loopback</b>	Sets MWI debugging for the Cisco IP phone.
<b>debug ephone raw</b>	Provides raw low-level protocol debugging display for all SCCP messages.
<b>debug ephone register</b>	Sets registration debugging for the Cisco IP phone.
<b>debug ephone state</b>	Sets state debugging for the Cisco IP phone.

<b>Command</b>	<b>Description</b>
<b>debug ephone statistics</b>	Sets statistics debugging for the Cisco IP phone.
<b>show debugging</b>	Displays information about the types of debugging that are enabled for your router.

# debug ephone qov

To display quality of voice (QOV) statistics for calls when preset limits are exceeded, use the **debug ephone qov** command in privileged EXEC mode. To disable debugging, use the **no** form of this command.

```
debug ephone qov [mac-address mac-address]
```

```
no debug ephone qov [mac-address mac-address]
```

## Syntax Description

**mac-address** *mac-address* (Optional) Specifies the MAC address of a Cisco IP phone for debugging.

## Command Modes

Privileged EXEC

## Command History

Release	Modification
12.2(15)ZJ2	This command was introduced for Cisco CallManager Express 3.0 and Cisco Survivable Remote Site Telephony (SRST) Version 3.0.
12.3(4)T	This command was integrated into Cisco IOS Release 12.3(4)T.

## Usage Guidelines

Once enabled, the **debug ephone qov** command produces output only when the QOV statistics reported by phones exceed preset limits. Phones are polled every few seconds for QOV statistics on VoIP calls only, not on local PSTN calls. An output report is produced when limits are surpassed for either or both of the following:

- Lost packets—A report is triggered when two adjacent QOV samples show an increase of four or more lost packets between samples. The report is triggered by an increase of lost packets in a short period of time, not by the total number of lost packets.
- Jitter and latency—A report is triggered when either jitter or latency exceeds 100 milliseconds.

To receive a QOV report at the end of each call regardless of whether the QOV limits have been exceeded, enable the **debug ephone alarm** command in addition to the **debug ephone qov** command.

The **debug ephone statistics** command displays the raw statistics that are polled from phones and used to generate QOV reports.

**Examples**

The following sample output describes QOV statistics for a call on ephone 5:

```
Router# debug ephone qov

Mar  7 00:54:57.329:ephone-5[7]:QOV DN 14 chan 1 (1514) ref 4 called=1514 calling=8101
Mar  7 00:54:57.329:ephone-5[7][SEP000DBDBEF37D]:Lost 91 Jitter 0 Latency 0
Mar  7 00:54:57.329:ephone-5[7][SEP000DBDBEF37D]:previous Lost 0 Jitter 0 Latency 0
Mar  7 00:54:57.329:ephone-5[7][SEP000DBDBEF37D]:Router sent 1153 pkts, current phone got
1141
received by all (shared) phones 0
Mar  7 00:54:57.329:ephone-5[7]:worst jitter 0 worst latency 0
Mar  7 00:54:57.329:ephone-5[7]:Current phone sent 1233 packets

Mar  7 00:54:57.329:ephone-5[7]:Signal Level to phone 3408 (-15 dB) peak 3516 (-15 dB)
```

Table 72 describes the significant fields shown in the display.

**Table 72** *debug ephone qov Field Descriptions*

Field	Description
Lost	Number of lost packets reported by the IP phone.
Jitter, Latency	The most recent jitter and latency parameters reported by the IP phone.
previous Lost, Jitter, Latency	Values from the previous QOV statistics report that were used as the comparison points against which the current statistics triggered generation of the current report.
Router sent pkts	Number of packets sent by the router to the IP phone. This number is the total for the entire call, even if the call is moved from one phone to another during a call, which can happen with shared lines.
current phone got	Number of packets received by the phone currently terminating the call. This number is the total for the entire call, even if the call is moved from one phone to another during a call, which can happen with shared lines.
worst jitter, worst latency	Highest value reported by the phone during the call.
Current phone sent packets	Number of packets that the current phone claims it sent during the call.
Signal Level to phone	Signal level seen in G.711 voice packet data prior to the sending of the most recent voice packet to the phone. The first number is the raw sample value, converted from G.711 to 16-bit linear format and left-justified. The number in parentheses is the value in decibels (dB), assuming that 32,767 is about +3 dB.  <b>Note</b> This value is meaningful only if the call uses a G.711 codec.

**Related Commands**

Command	Description
<b>debug ephone alarm</b>	Displays alarm messages for IP phones.
<b>debug ephone statistics</b>	Displays call statistics for IP phones.

# debug ephone raw

To provide raw low-level protocol debugging display for all Skinny Client Control Protocol (SCCP) messages, use the **debug ephone raw** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug ephone raw** [*mac-address mac-address*]

**no debug ephone raw** [*mac-address mac-address*]

## Syntax Description

<b>mac-address</b>	(Optional) Defines the MAC address of the Cisco IP phone.
<i>mac-address</i>	(Optional) Specifies the MAC address of the Cisco IP phone.

## Defaults

No default behavior or values

## Command Modes

Privileged EXEC

## Command History

Release	Modification
12.1(5)YD	This command was introduced on the following platforms: Cisco 2600 series and Cisco 3600 series multiservice routers, and Cisco IAD2420 series Integrated Access Devices (IADs).
12.2(2)XT	This command was implemented on the Cisco 1750 and Cisco 1751 multiservice routers.
12.2(8)T	This command was implemented on the Cisco 3725 and Cisco 3745 routers.
12.2(8)T1	This command was implemented on the Cisco 2600-XM and Cisco 2691 routers.
12.2(11)T	This command was implemented on the Cisco 1760 routers.

## Usage Guidelines

The **debug ephone raw** command provides raw low-level protocol debug display for all SCCP messages. The debug display provides byte level display of Skinny TCP socket messages.

If the **mac-address** keyword is not used, the **debug ephone raw** command debugs all Cisco IP phones that are registered to the router. You can remove debugging for the Cisco IP phones that you do not want to debug by using the **mac-address** keyword with the **no** form of this command.

You can enable or disable debugging on any number of Cisco IP phones. To see the Cisco IP phones that have debugging enabled, enter the **show ephone** command and look at the debug field in the output. When debugging is enabled for a Cisco IP phone, the debug output is displayed for the directory numbers associated with the Cisco IP phone.

**Examples**

The following is sample output of raw protocol debugging for the Cisco IP phone with MAC address 0030.94c3.E1A8:

```
Router# debug ephone raw mac-address 0030.94c3.E1A8

EPHONE raw protocol debugging is enabled for phone 0030.94C3.E1A8

1d05h: skinny socket received 4 bytes on socket [1]
0 0 0 0
1d05h:
1d05h: SkinnyMessageID = 0
1d05h: skinny send 4 bytes
4 0 0 0 0 0 0 0 0 1 0 0
1d05h: socket [1] sent 12 bytes OK (incl hdr) for ephone-(1)

1d06h: skinny socket received 4 bytes on socket [1]
0 0 0 0
1d06h:
1d06h: SkinnyMessageID = 0
1d06h: skinny send 4 bytes
4 0 0 0 0 0 0 0 0 1 0 0
1d06h: socket [1] sent 12 bytes OK (incl hdr) for ephone-(1)
```

**Related Commands**

Command	Description
<b>debug ephone alarm</b>	Sets SkinnyStation alarm messages debugging for the Cisco IP phone.
<b>debug ephone detail</b>	Sets detail debugging for the Cisco IP phone.
<b>debug ephone error</b>	Sets error debugging for the Cisco IP phone.
<b>debug ephone keepalive</b>	Sets keepalive debugging for the Cisco IP phone.
<b>debug ephone loopback</b>	Sets MWI debugging for the Cisco IP phone.
<b>debug ephone pak</b>	Provides voice packet level debugging and prints the contents of one voice packet in every 1024 voice packets.
<b>debug ephone register</b>	Sets registration debugging for the Cisco IP phone.
<b>debug ephone state</b>	Sets state debugging for the Cisco IP phone.
<b>debug ephone statistics</b>	Sets statistics debugging for the Cisco IP phone.
<b>show debugging</b>	Displays information about the types of debugging that are enabled for your router.

# debug ephone register

To set registration debugging for the Cisco IP phone, use the **debug ephone register** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug ephone register** [**mac-address** *mac-address*]

**no debug ephone register** [**mac-address** *mac-address*]

## Syntax Description

<b>mac-address</b>	(Optional) Defines the MAC address of the Cisco IP phone.
<i>mac-address</i>	(Optional) Specifies the MAC address of the Cisco IP phone.

## Defaults

No default behavior or values

## Command Modes

Privileged EXEC

## Command History

Release	Modification
12.1(5)YD	This command was introduced on the following platforms: Cisco 2600 series and Cisco 3600 series multiservice routers, and Cisco IAD2420 series Integrated Access Devices (IADs).
12.2(2)XT	This command was implemented on the Cisco 1750 and Cisco 1751 multiservice routers.
12.2(8)T	This command was integrated into Cisco IOS Release 12.2(8)T and implemented on the Cisco 3725 and Cisco 3745 routers.
12.2(8)T1	This command was implemented on the Cisco 2600-XM and Cisco 2691 routers.
12.2(11)T	This command was implemented on the Cisco 1760 routers.

## Usage Guidelines

The **debug ephone register** command sets registration debugging for the Cisco IP phones.

If the **mac-address** keyword is not used, the **debug ephone register** command debugs all Cisco IP phones that are registered to the router. You can remove debugging for the Cisco IP phones that you do not want to debug by using the **mac-address** keyword with the **no** form of this command.

You can enable or disable debugging on any number of Cisco IP phones. To see the Cisco IP phones that have debugging enabled, enter the **show ephone** command and look at the debug field in the output.

When debugging is enabled for a Cisco IP phone, the debug output is displayed for the directory numbers associated with the Cisco IP phone.

**Examples**

The following is sample output of registration debugging for the Cisco IP phone with MAC address 0030.94c3.8724:

```
Router# debug ephone register mac-address 0030.94c3.8724

Ephone registration debugging is enabled

1d06h: New Skinny socket accepted [1] (2 active)
1d06h: sin_family 2, sin_port 50778, in_addr 10.1.0.21
1d06h: skinny_add_socket 1 10.1.0.21 50778
1d06h: ephone-(1)[1] StationRegisterMessage (2/3/12) from 10.1.0.21
1d06h: ephone-(1)[1] Register StationIdentifier DeviceName SEP003094C3E1A8
1d06h: ephone-(1)[1] StationIdentifier Instance 1 deviceType 7
1d06h: ephone-1[-1]:stationIpAddr 10.1.0.21
1d06h: ephone-1[-1]:maxStreams 0
1d06h: ephone-(1) Allow any Skinny Server IP address 10.1.0.6
.
.
.
1d06h: ephone-1[1]:RegisterAck sent to ephone 1: keepalive period 30
.
```

**Related Commands**

Command	Description
<b>debug ephone alarm</b>	Sets SkinnyStation alarm messages debugging for the Cisco IP phone.
<b>debug ephone detail</b>	Sets detail debugging for the Cisco IP phone.
<b>debug ephone error</b>	Sets error debugging for the Cisco IP phone.
<b>debug ephone keepalive</b>	Sets keepalive debugging for the Cisco IP phone.
<b>debug ephone loopback</b>	Sets MWI debugging for the Cisco IP phone.
<b>debug ephone pak</b>	Provides voice packet level debugging and prints the contents of one voice packet in every 1024 voice packets.
<b>debug ephone raw</b>	Provides raw low-level protocol debugging display for all SCCP messages.
<b>debug ephone state</b>	Sets state debugging for the Cisco IP phone.
<b>debug ephone statistics</b>	Sets statistics debugging for the Cisco IP phone.
<b>show debugging</b>	Displays information about the types of debugging that are enabled for your router.

# debug ephone state

To set state debugging for the Cisco IP phone, use the **debug ephone state** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug ephone state** [**mac-address** *mac-address*]

**no debug ephone state** [**mac-address** *mac-address*]

## Syntax Description

<b>mac-address</b>	(Optional) Defines the MAC address of the Cisco IP phone.
<i>mac-address</i>	(Optional) Specifies the MAC address of the Cisco IP phone.

## Defaults

No default behavior or values

## Command Modes

Privileged EXEC

## Command History

Release	Modification
12.1(5)YD	This command was introduced on the following platforms: Cisco 2600 series and Cisco 3600 series multiservice routers, and Cisco IAD2420 series Integrated Access Devices (IADs).
12.2(2)XT	This command was implemented on the Cisco 1750 and Cisco 1751 multiservice routers.
12.2(8)T	This command was integrated into Cisco IOS Release 12.2(8)T and implemented on the Cisco 3725 and Cisco 3745 routers.
12.2(8)T1	This command was implemented on the Cisco 2600-XM and Cisco 2691 routers.
12.2(11)T	This command was implemented on Cisco 1760 routers.

## Usage Guidelines

The **debug ephone state** command sets state debugging for the Cisco IP phones.

If the **mac-address** keyword is not used, the **debug ephone state** command debugs all Cisco IP phones that are registered to the router. You can remove debugging for the Cisco IP phones that you do not want to debug by using the **mac-address** keyword with the **no** form of this command.

You can enable or disable debugging on any number of Cisco IP phones. To see the Cisco IP phones that have debugging enabled, enter the **show ephone** command and look at the debug field in the output.

When debugging is enabled for a Cisco IP phone, the debug output is displayed for the directory numbers associated with the Cisco IP phone.

**Examples**

The following is sample output of state debugging for the Cisco IP phone with MAC address 0030.94c3.E1A8:

```
Router# debug ephone state mac-address 0030.94c3.E1A8

EPHONE state debugging is enabled for phone 0030.94C3.E1A8

1d06h: ephone-1[1]:OFFHOOK
1d06h: ephone-1[1]:SIEZE on activeline 0
1d06h: ephone-1[1]:SetCallState line 1 DN 1 TsOffHook
1d06h: ephone-1[1]:Skinny-to-Skinny call DN 1 to DN 2 instance 1
1d06h: ephone-1[1]:SetCallState line 1 DN 1 TsRingOut
1d06h: ephone-1[1]:Call Info DN 1 line 1 ref 158 called 5002 calling 5001
1d06h: ephone-1[1]: Jane calling
1d06h: ephone-1[1]: Jill
1d06h: ephone-1[1]:SetCallState line 3 DN 2 TsRingIn
1d06h: ephone-1[1]:Call Info DN 2 line 3 ref 159 called 5002 calling 5001
1d06h: ephone-1[1]: Jane calling
1d06h: ephone-1[1]: Jill
1d06h: ephone-1[1]:SetCallState line 3 DN 2 TsCallRemoteMultiline
1d06h: ephone-1[1]:SetCallState line 1 DN 1 TsConnected
1d06h: ephone-1[1]:OpenReceive DN 1 codec 4:G711Ulaw64k duration 10 ms bytes 80
1d06h: ephone-1[1]:OpenReceiveChannelAck 1.2.172.21 port=24010
1d06h: ephone-1[1]:StartMedia 1.2.172.22 port=24612
1d06h: DN 1 codec 4:G711Ulaw64k duration 10 ms bytes 80
1d06h: ephone-1[1]:CloseReceive
1d06h: ephone-1[1]:StopMedia
1d06h: ephone-1[1]:SetCallState line 3 DN 2 TsOnHook
1d06h: ephone-1[1]:SetCallState line 1 DN 1 TsOnHook
1d06h: ephone-1[1]:SpeakerPhoneOnHook
1d06h: ephone-1[1]:ONHOOK
1d06h: ephone-1[1]:SpeakerPhoneOnHook
1d06h: SkinnyReportDnState DN 1 ONHOOK
```

**Related Commands**

Command	Description
<b>debug ephone alarm</b>	Sets SkinnyStation alarm messages debugging for the Cisco IP phone.
<b>debug ephone detail</b>	Sets detail debugging for the Cisco IP phone.
<b>debug ephone error</b>	Sets error debugging for the Cisco IP phone.
<b>debug ephone keepalive</b>	Sets keepalive debugging for the Cisco IP phone.
<b>debug ephone loopback</b>	Sets MWI debugging for the Cisco IP phone.
<b>debug ephone pak</b>	Provides voice packet level debugging and prints the contents of one voice packet in every 1024 voice packets.
<b>debug ephone raw</b>	Provides raw low-level protocol debugging display for all SCCP messages.
<b>debug ephone register</b>	Sets registration debugging for the Cisco IP phone.
<b>debug ephone</b>	Sets statistics debugging for the Cisco IP phone.
<b>show debugging</b>	Displays information about the types of debugging that are enabled for your router.

# debug ephone statistics

To set call statistics debugging for the Cisco IP phone, use the **debug ephone statistics** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug ephone statistics** [**mac-address** *mac-address*]

**no debug ephone statistics** [**mac-address** *mac-address*]

## Syntax Description

<b>mac-address</b>	(Optional) Defines the MAC address of the Cisco IP phone.
<i>mac-address</i>	(Optional) Specifies the MAC address of the Cisco IP phone.

## Defaults

No default behavior or values

## Command Modes

Privileged EXEC

## Command History

Release	Modification
12.1(5)YD	This command was introduced on the following platforms: Cisco 2600 series and Cisco 3600 series multiservice routers, and Cisco IAD2420 series Integrated Access Devices (IADs).
12.2(2)XT	This command was implemented on the Cisco 1750 and Cisco 1751 multiservice routers.
12.2(8)T	This command was integrated into Cisco IOS Release 12.2(8)T and implemented on the Cisco 3725 and Cisco 3745 routers.
12.2(8)T1	This command was implemented on the Cisco 2600-XM and Cisco 2691 routers.
12.2(11)T	This command was implemented on the Cisco 1760 routers.

## Usage Guidelines

The **debug ephone statistics** command provides a debug monitor display of the periodic messages from the Cisco IP phone to the router. These include transmit-and-receive packet counts and an estimate of drop packets. The call statistics can also be displayed for live calls using the **show ephone** command.

If the **mac-address** keyword is not used, the **debug ephone statistics** command debugs all Cisco IP phones that are registered to the router. You can remove debugging for the Cisco IP phones that you do not want to debug by using the **mac-address** keyword with the **no** form of this command.

You can enable or disable debugging on any number of Cisco IP phones. To see the Cisco IP phones that have debugging enabled, enter the **show ephone** command and look at the debug field in the output. When debugging is enabled for a Cisco IP phone, the debug output is displayed for the directory numbers associated with the Cisco IP phone.

**Examples**

The following is sample output of statistics debugging for the Cisco IP phone with MAC address 0030.94C3.E1A8:

```
Router# debug ephone statistics mac-address 0030.94C3.E1A8

EPHONE statistics debugging is enabled for phone 0030.94C3.E1A8

1d06h: Clear Call Stats for DN 1 call ref 162
1d06h: Clear Call Stats for DN 1 call ref 162
1d06h: Clear Call Stats for DN 1 call ref 162
1d06h: Clear Call Stats for DN 2 call ref 163
1d06h: ephone-1[1]:GetCallStats line 1 ref 162 DN 1: 5001
1d06h: ephone-1[1]:Call Stats for line 1 DN 1 5001 ref 162
1d06h: ephone-1[1]:TX Pkts 0 bytes 0 RX Pkts 0 bytes 0
1d06h: ephone-1[1]:Pkts lost 4504384 jitter 0 latency 0
1d06h: ephone-1[1]:Src 0.0.0.0 0 Dst 0.0.0.0 0 bytes 80 vad 0 G711Ulaw64k
1d06h: ephone-1[1]:GetCallStats line 1 ref 162 DN 1: 5001
1d06h: STATS: DN 1 Packets Sent 0
1d06h: STATS: DN 2 Packets Sent 0
1d06h: ephone-1[1]:Call Stats found DN -1 from Call Ref 162
1d06h: ephone-1[1]:Call Stats for line 0 DN -1 5001 ref 162
1d06h: ephone-1[1]:TX Pkts 275 bytes 25300 RX Pkts 275 bytes 25300
1d06h: ephone-1[1]:Pkts lost 0 jitter 0 latency 0
```

**Related Commands**

Command	Description
<b>debug ephone alarm</b>	Sets SkinnyStation alarm messages debugging for the Cisco IP phone.
<b>debug ephone detail</b>	Sets detail debugging for the Cisco IP phone.
<b>debug ephone error</b>	Sets error debugging for the Cisco IP phone.
<b>debug ephone keepalive</b>	Sets keepalive debugging for the Cisco IP phone.
<b>debug ephone loopback</b>	Sets MWI debugging for the Cisco IP phone.
<b>debug ephone pak</b>	Provides voice packet level debugging and prints the contents of one voice packet in every 1024 voice packets.
<b>debug ephone raw</b>	Provides raw low-level protocol debugging display for all SCCP messages.
<b>debug ephone register</b>	Sets registration debugging for the Cisco IP phone.
<b>debug ephone state</b>	Sets state debugging for the Cisco IP phone.
<b>show debugging</b>	Displays information about the types of debugging that are enabled for your router.

# debug ephone vm-integration

To display pattern manipulation information used for integration with voice-mail applications, use the **debug ephone vm-integration** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

**debug ephone vm-integration** [**mac-address** *mac-address*]

**no debug ephone vm-integration** [**mac-address** *mac-address*]

## Syntax Description

**mac-address** *mac-address* (Optional) Specifies the MAC address of a Cisco IP phone for debugging.

## Command Modes

Privileged EXEC

## Command History

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

## Usage Guidelines

This command displays the voice-mail integration patterns that were created using the **pattern** commands in vm-integration configuration mode. The patterns are used to forward calls to a voice-mail number that is set with the **voicemail** command.

If you do not specify the **mac-address** keyword, the **debug ephone vm-integration** command debugs all Cisco IP phones that are registered to the router. To remove debugging for Cisco IP phones, enter the **no** form of this command with the **mac-address** keyword.

## Examples

The following sample output shows information for the vm-integration tokens that have been defined:

```
Router# debug ephone vm-integration

*Jul 23 15:38:03.294:ephone-3[3]:StimulusMessage 15 (1) From ephone 2
*Jul 23 15:38:03.294:ephone-3[3]:Voicemail access number pattern check
*Jul 23 15:38:03.294:SkinnyGetCallState for DN 3 chan 1 IDLE
*Jul 23 15:38:03.294:called DN -1 chan 1, calling DN -1 chan 1 phone -1 s2s:0
*Jul 23 15:38:03.294:dn number for dn 3 is 19003
*Jul 23 15:38:03.294:Updated number for token 1 is 19003
*Jul 23 15:38:03.294:CDN number for dn 3 is
*Jul 23 15:38:03.294:Updated number for token 2 is
*Jul 23 15:38:03.294:Updated number for token 0 is
*Jul 23 15:38:03.294:Update is 219003*
*Jul 23 15:38:03.294:New Voicemail number is 19101219003*
```

Table 73 describes the significant fields shown in the display.

**Table 73** *debug ephone vm-integration Field Descriptions*

Field	Description
token 0	First token that was defined in the pattern.
token 1	Second token that was defined in the pattern.
token 2	Third token that was defined in the pattern.

#### Related Commands

Command	Description
<b>pattern direct</b>	Configures the DTMF digit pattern forwarding necessary to activate the voice-mail system when a user presses the Messages button on a phone.
<b>pattern ext-to-ext busy</b>	Configures the DTMF digit pattern forwarding necessary to activate the voice-mail system once an internal extension reaches a busy extension and the call is forwarded to voice mail.
<b>pattern ext-to-ext no-answer</b>	Configures the DTMF digit pattern forwarding necessary to activate the voice-mail system once an internal extension fails to connect to an extension and the call is forwarded to voice mail.
<b>pattern trunk-to-ext busy</b>	Configures the DTMF digit pattern forwarding necessary to activate the voice-mail system once an external trunk call reaches a busy extension and the call is forwarded to voice mail.
<b>pattern trunk-to-ext no-answer</b>	Configures the DTMF digit pattern forwarding necessary to activate the voice-mail system when an external trunk call reaches an unanswered extension and the call is forwarded to voice mail.
<b>vm-integration</b>	Enters voice-mail integration configuration mode and enables voice-mail integration with DTMF and analog voice-mail systems.
<b>voicemail</b>	Defines the telephone number that is speed-dialed when the Messages button on a Cisco IP phone is pressed.

# debug errors

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

**debug errors**

**no debug errors**

---

**Syntax Description** This command has no arguments or keywords.

---

**Command Modes** Privileged EXEC

---

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

```
Router# debug errors  
  
(2/0): Encapsulation error, link=7, host=836CA86D.  
(4/0): VCD#7 failed to echo OAM. 4 tries
```

The first line of output indicates that a packet was routed to the interface, but no static map was set up to route that packet to the proper virtual circuit.

The second line of output shows that an OAM F5 (virtual circuit) cell error occurred.

# debug eswilp

To enable debugging of Ethernet switch network module features, use the **debug eswilp** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

```
debug eswilp { dot1x | filtermgr | fltdrv | igmp | port-driver | power-supply | span | switch-pm }
```

```
no debug eswilp { dot1x | filtermgr | fltdrv | igmp | port-driver | power-supply | span | switch-pm }
```

Syntax Description	Parameter	Description
	<b>dot1x</b>	Displays Ethernet Switch with Inline Power (ESWILP) 802.1x debugging messages.
	<b>filtermgr</b>	Displays ESWILP filter manager debugging messages.
	<b>fltdrv</b>	Displays ESWILP filter driver debugging messages.
	<b>igmp</b>	Displays ESWILP Internet Group Management Protocol (IGMP) debugging messages.
	<b>port-driver</b>	Displays ESWILP port driver debugging messages.
	<b>power-supply</b>	Displays ESWILP power supply information debugging messages.
	<b>span</b>	Displays ESWILP Switched Port Analyzer (SPAN) debugging messages.
	<b>switch-pm</b>	Displays ESWILP switch port manager debugging messages.

**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. The <b>dot1x</b> , <b>filtermgr</b> , and <b>fltdrv</b> keywords were added.
	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.

**Usage Guidelines** The **undebug eswilp** command is the same as the **no debug eswilp** command.

**Examples** The following example shows debugging messages for the IGMP snooping services on the Ethernet switch network module being displayed:

```
Router# debug eswilp igmp
```

---

**Related Commands**

<b>Command</b>	<b>Description</b>
<b>show debugging</b>	Displays information about the types of debugging that are enabled.

# debug event manager

To turn on the debugging output of Embedded Event Manager (EEM) processes, use the **debug event manager** command in privileged EXEC mode. To turn off debugging output, use the **no** form of this command or the **undebug** command.

```
debug event manager {action cli | action cns | action mail | all | api calls | api errors | detector
all | detector application | detector cli | detector counter | detector interface | detector
ioswdsysmon | detector none | detector snmp | detector syslog | detector timer | metricdir |
policydir | server events | server scheduling | tcl cli_library | tcl commands | tcl
smtp_library }
```

```
no debug event manager {action cli | action cns | action mail | all | api calls | api errors | detector
all | detector application | detector cli | detector counter | detector interface | detector
ioswdsysmon | detector none | detector snmp | detector syslog | detector timer | metricdir |
policydir | server events | server scheduling | tcl cli_library | tcl commands | tcl
smtp_library }
```

## Syntax Description

<b>action cli</b>	Displays debugging messages about command-line interface (CLI) event messages.
<b>action cns</b>	Displays debugging messages about CNS event messages.
<b>action mail</b>	Displays debugging messages about e-mail event messages.
<b>all</b>	Displays all debugging messages.
<b>api calls</b>	Displays debugging messages about application programming interface (API) calls.
<b>api errors</b>	Displays debugging messages about API errors.
<b>detector all</b>	Displays all event detector debugging messages.
<b>detector application</b>	Displays debugging messages about the application-specific event detector.
<b>detector cli</b>	Displays debugging messages about the CLI event detector.
<b>detector counter</b>	Displays debugging messages about the counter event detector.
<b>detector interface</b>	Displays debugging messages about the interface counter event detector.
<b>detector ioswdsysmon</b>	Displays debugging messages about the watchdog event detector.
<b>detector none</b>	Displays debugging messages about the none event detector.
<b>detector snmp</b>	Displays debugging messages about the Simple Network Management Protocol (SNMP) event detector.
<b>detector syslog</b>	Displays debugging messages about the syslog event detector.
<b>detector timer</b>	Displays debugging messages about the timer event detector.
<b>metricdir</b>	Displays debugging messages about the EEM metrics event detector.
<b>policydir</b>	Displays debugging messages about the EEM policy director.
<b>server events</b>	Displays debugging messages about the EEM server events.
<b>server scheduling</b>	Displays all debugging messages about the EEM server scheduling events.
<b>tcl cli_library</b>	Displays all debugging messages about the Tool Command Language (Tcl) command-line interface (CLI) library.

<b>tcl commands</b>	Displays all debugging messages about the Tcl commands.
<b>tcl smtp_library</b>	Displays all debugging messages about the Tcl Simple Mail Transfer Protocol (SMTP) library.

**Command Modes** Privileged EXEC

Release	Modification
12.0(26)S	This command was introduced.
12.3(4)T	This command was integrated into Cisco IOS Release 12.3(4)T.
12.3(2)XE	This command was integrated into Cisco IOS Release 12.3(2)XE.
12.2(25)S	This command was integrated into Cisco IOS Release 12.2(25)S.
12.3(14)T	The <b>action cli</b> , <b>action mail</b> , <b>detector all</b> , <b>detector cli</b> , <b>detector none</b> , and <b>metricdir</b> keywords were added.

**Usage Guidelines** Use the **debug event manager** command to troubleshoot EEM command operations.



**Caution**

Use any debugging command with caution because the volume of generated output can slow or stop the router operations. We recommend that this command be used only under the supervision of a Cisco engineer.

**Examples** The following example turns on debugging messages about EEM server events and then configures an applet to write a message—Test message—to syslog. The debug output that follows displays the various EEM operations that occur as the applet is processed.

```
Router# debug event manager server events

Debug Embedded Event Manager server events debugging is on
Router# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)# event manager applet timer-test
Router(config-applet)# event timer countdown time 20
Router(config-applet)# action label1 syslog msg "Test message"
Router(config-applet)# end
Router#

03:46:55: fh_server: fh_io_msg: received msg 6 from client jobid 11
03:46:55: fh_server: fh_io_msg: handling event register with esid = 23
03:46:55: fh_msg_send_to_fd: receive a reply msg, minor: 5
03:46:55: fh_server: fh_io_msg: received msg 26 from client jobid 11
03:46:55: fh_msg_send_to_fd: receive a reply msg, minor: 5
03:46:55: %SYS-5-CONFIG_I: Configured from console by console
03:47:15: fd_pulse_hdlr: received a pulse from /dev/fm/fd_timer
03:47:15: fh_msg_send_to_fd: receive a reply msg, minor: 5
03:47:15: fd_pulse_hdlr: received FH_MSG_EVENT_PUBLISH
03:47:15: fh_schedule_callback: fh_schedule_callback: cc=632C0B68 prev_epc=0; epc=63A41670
03:47:15: fh_io_msg: received FH_MSG_API_INIT; jobid=13, processid=82, client=3, job
name=EEM Callback Thread
03:47:15: fh_server: fh_io_msg: received msg 10 from client jobid 13
```

```

03:47:15: %HA_EM-6-LOG: timer-test: Test message
03:47:15: fh_server: fh_io_msg: received msg 62 from client jobid 13
03:47:15: fh_schedule_callback: fh_schedule_callback: cc=632C0B68 prev_epc=63A41670; epc=0
03:47:15: fh_server: fh_io_msg: received msg 1 from client jobid 13
03:47:15: fh_io_msg: received FH_MSG_API_CLOSE client=3

```

Table 74 describes some of the significant fields shown in the display.

**Table 74** *debug event manager Field Descriptions*

Field	Description
Debug Embedded Event Manager server events debugging	Indicates the type of debugging output and whether the debugging is on or off.
fh_server	Indicates a server event message.
fh_io_msg	Indicates that a message has been sent to, or received from, a client process.
fh_msg_send_to_fd	Indicates that a message has been sent to the event detector.
fd_pulse_hndlr	Indicates that a message has been received by the event detector pulse handler.

# debug events

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

**debug events**

**no debug events**

---

**Syntax Description** This command has no arguments or keywords.

---

**Command Modes** Privileged EXEC

---

**Usage Guidelines** This command displays events that occur on the interface processor and is useful for diagnosing problems in a network. It provides an overall picture of the stability of the network. In a stable network, the **debug events** command does not return any information. If the command generates numerous messages, the messages can indicate the possible source of problems.

When configuring or making changes to a router or interface for, enable the **debug events** command. Doing so alerts you to the progress of the changes or to any errors that might result. Also use this command periodically when you suspect network problems.

---

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

```
Router# debug events

RESET(4/0): PLIM type is 1, Rate is 100Mbps
aip_disable(4/0): state=1
config(4/0)
aip_love_note(4/0): asr=0x201
aip_enable(4/0)
aip_love_note(4/0): asr=0x4000
aip_enable(4/0): restarting VCs: 7
aip_setup_vc(4/0): vc:1 vpi:1 vci:1
aip_love_note(4/0): asr=0x200
aip_setup_vc(4/0): vc:2 vpi:2 vci:2
aip_love_note(4/0): asr=0x200
aip_setup_vc(4/0): vc:3 vpi:3 vci:3
aip_love_note(4/0): asr=0x200
aip_setup_vc(4/0): vc:4 vpi:4 vci:4
aip_love_note(4/0): asr=0x200
aip_setup_vc(4/0): vc:6 vpi:6 vci:6
aip_love_note(4/0): asr=0x200
aip_setup_vc(4/0): vc:7 vpi:7 vci:7
aip_love_note(4/0): asr=0x200
aip_setup_vc(4/0): vc:11 vpi:11 vci:11
aip_love_note(4/0): asr=0x200
```

Table 75 describes the significant fields shown in the display.

**Table 75** *debug events Field Descriptions*

Field	Description
PLIM type	Indicates the interface rate in Mbps. Possible values are: <ul style="list-style-type: none"> <li>• 1 = TAXI(4B5B) 100 Mbps</li> <li>• 2 = SONET 155 Mbps</li> <li>• 3 = E3 34 Mbps</li> </ul>
state	Indicates current state of the ATM Interface Processor (AIP). Possible values are: <ul style="list-style-type: none"> <li>• 1 = An ENABLE will be issued soon.</li> <li>• 0 = The AIP will remain shut down.</li> </ul>
asr	Defines a bitmask, which indicates actions or completions to commands. Valid bitmask values are: <ul style="list-style-type: none"> <li>• 0x0800 = AIP crashed, reload may be required.</li> <li>• 0x0400 = AIP detected a carrier state change.</li> <li>• 0x0n00 = Command completion status. Command completion status codes are: <ul style="list-style-type: none"> <li>- n = 8 Invalid physical layer interface module (PLIM) detected</li> <li>- n = 4 Command failed</li> <li>- n = 2 Command completed successfully</li> <li>- n = 1 CONFIG request failed</li> <li>- n = 0 Invalid value</li> </ul> </li> </ul>

The following line indicates that the AIP was reset. The PLIM detected was 1, so the maximum rate is set to 100 Mbps.

```
RESET(4/0): PLIM type is 1, Rate is 100Mbps
```

The following line indicates that the AIP was given a **shutdown** command, but the current configuration indicates that the AIP should be up:

```
aip_disable(4/0): state=1
```

The following line indicates that a configuration command has been completed by the AIP:

```
aip_love_note(4/0): asr=0x201
```

The following line indicates that the AIP was given a **no shutdown** command to take it out of the shutdown state:

```
aip_enable(4/0)
```

The following line indicates that the AIP detected a carrier state change. It does not indicate that the carrier is down or up, only that it has changed.

```
aip_love_note(4/0): asr=0x4000
```

The following line of output indicates that the AIP enable function is restarting all permanent virtual circuits (PVCs) automatically:

```
aip_enable(4/0): restarting VCs: 7
```

The following lines of output indicate that PVC 1 was set up and a successful completion code was returned:

```
aip_setup_vc(4/0): vc:1 vpi:1 vci:1  
aip_love_note(4/0): asr=0x200
```