debug eap through debug hw-module subslot periodic

debug eap through debug hw-module subslot periodic

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**| *method*] [**authenticator**| **peer**] {**all**| **errors**| **events**| **packets**| **sm**}

no debug eap [**all**| *method*] [**authenticator**| **peer**] {**all**| **errors**| **events**| **packets**| **sm**}

Syntax Description

all | *method*	(Optional) Specifies the method to which the debug command refers. • The **all** keyword turns on debugging for all EAP methods, including the EAP framework. • The *method* argument turns on debugging for specific methods. • This keyword or argument is dynamically linked into the parse chain and is present only if the method itself is present. • If this keyword or argument is omitted, the **debug** command is applied to the EAP framework.
authenticator	(Optional) Limits the scope of the output to only authenticator contexts.
peer	(Optional) Limits the scope of the output to only peer contexts.
all	Debugging is turned on for all debug types.
errors	Displays information about EAP packet errors.
events	Displays information about EAP events.
packets	Turns on packet debugging for the specified method or methods.
sm	Turns on state machine debugging for the specified method or methods.

Command Modes

Privileged EXEC #

Command History

Release	Modification
12.3(8)T	This command was introduced.
12.4(6)T	The *method* argument and **authenticator** and **peer** keywords were added.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXI	This command was integrated into Cisco IOS Release 12.2(33)SXI.

Examples

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

```
Router# debug eap all
*Nov  7 13:05:58.512: EAP-EVENT: Received get canned status from lower layer (0x00000000)
*Nov  7 13:05:59.460: EAP-EVENT: Received context create from lower layer (0x00000009)
*Nov  7 13:05:59.460:     eap_authen : initial state eap_auth_initialize has enter
*Nov  7 13:05:59.460: EAP-EVENT: Started 'Authenticator Start' timer (1s) for EAP sesion
handle 0xD6000008
*Nov  7 13:05:59.460: EAP-EVENT: Allocated new EAP context (handle = 0xD6000008)
*Nov  7 13:05:59.464: EAP-EVENT: Started EAP tick timer
*Nov  7 13:06:00.488: EAP-EVENT: 'Authenticator Start' timer expired for EAP sesion handle
 0xD6000008
*Nov  7 13:06:00.488:     eap_authen : during state eap_auth_initialize, got event
21(eapStartTmo)
*Nov  7 13:06:00.488: @@@ eap_authen : eap_auth_initialize -> eap_auth_select_action
*Nov  7 13:06:00.488:     eap_authen : during state eap_auth_select_action, got event
17(eapDecisionPropose)
*Nov  7 13:06:00.488: @@@ eap_authen : eap_auth_select_action -> eap_auth_propose_method
```

Related Commands

Command	Description
debug eou	Displays information about EAPoUDP.

debug ecfmpal

To enable debugging of the data path of the Ethernet Connectivity Fault Management (CFM) function, use the **debug ecfmpal** command in the privileged EXEC mode. To disable the debugging function, use the **no** form of this command.

debug ecfmpal {**all | api | common | ecfmpal | epl | isr**}

no debug ecfmpal {**all | api | common | ecfmpal | epl | isr**}

Syntax Description

all	Specifies all the platform Ethernet CFM events.
api	Specifies the platform-specific application program interface (API) Ethernet CFM events.
common	Specifies the common Ethernet CFM events.
ecfmpal	Specifies the general Ethernet CFM platform events.
epl	Specifies the end-point list Ethernet CFM events.
isr	Specifies the platform-interrupt service request events.

Command Default

Debugging is disabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
15.4(1)T	This command was introduced into Cisco IOS Release 15.4(1)T.

Usage Guidelines

Use this command to troubleshoot Ethernet CFM functions on the following routers:

- Cisco 3900 Series, Cisco 2900 Series, and Cisco 1900 Series Integrated Services Routers Generation 2
- Cisco 890 Series Integrated Services Routers

Examples

The following is a sample output of the **debug ecfmpal all** command:

```
Device# debug ecfmpal all

ECFMPAL EPL events debugging is on
ECFMPAL Ingress ISR events debugging is on
```

```
ECFMPAL events debugging is on
ECFMPAL API events debugging is on
ECFMPAL common events debugging is on
Router#
*Nov 15 05:10:25.909: ECFMPD-API: Port MEP Gi0/2.1 get 0 02DB7F44
```

debug eigrp address-family neighbor

To display debugging information about Enhanced Interior Gateway Routing Protocol (EIGRP) address family neighbors, use the **debug eigrp address-family neighbor** command in privileged EXEC mode. To disable debugging of EIGRP service-family neighbors, use the **no** form of this command.

debug eigrp address-family [**ipv4**| **ipv6**] **neighbor** [*ip-address*]

no debug eigrp address-family [**ipv4**| **ipv6**] **neighbor** [*ip-address*]

Syntax Description

ipv4	(Optional) Enables debugging for neighbors formed using the IPv4 protocol family.
ipv6	(Optional) Enables debugging for neighbors formed using the IPv6 protocol family.
ip-address	(Optional) IPv4 or IPv6 address of the neighbor. Specifying an address enables debugging for the service family at this address.

Command Default

Debugging of EIGRP service-family neighbors is disabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
15.0(1)M	This command was introduced.
12.2(33)SRE	This command was integrated into Cisco IOS Release 12.2(33)SRE.
12.2(33)XNE	This command was integrated into Cisco IOS Release 12.2(33)XNE.
Cisco IOS XE Release 2.5	This command was integrated into Cisco IOS XE Release 2.5.

Usage Guidelines

Consult Cisco technical support before using this command.

Caution Use of **debug** commands can have severe performance penalties and should be used with extreme caution. For this reason, Cisco recommends that you contact Cisco technical support before enabling a **debug** command.

Examples

The following example shows how to enable debugging of an EIGRP address-family neighbor at 10.0.0.0:

```
Router# debug eigrp address-family ipv4 neighbor 10.0.0.0
Neighbor target enabled on AS 3 for 10.0.0.0
*Mar 17 15:50:53.244: EIGRP: 10.0.0.0/24 - do advertise out Serial1/2
*Mar 17 15:50:53.244: EIGRP: Int 10.0.0.0/24 metric 20512000 -20000000 512000
*Mar 17 15:50:53.244: EIGRP: 10.0.0.0/24 - do advertise out Serial1/2
*Mar 17 15:50:53.244: EIGRP: Int 10.0.0.0/24 metric 28160 - 256002560
*Mar 17 15:50:53.244: EIGRP: 10.0.0.0/24 - do advertise out Serial1/2
*Mar 17 15:50:53.244: EIGRP: 10.0.0.0/24 - do advertise out Serial1/2
*Mar 17 15:50:53.244: EIGRP: Int 10.0.0.0/24 metric 28160 - 25600256
*Mar 17 15:50:53.668: EIGRP: Processing incoming UPDATE packet
*Mar 17 15:50:54.544: EIGRP: 10.0.0.0/24 - do advertise out Serial1/1
```

Related Commands

Command	Description
debug eigrp address-family notifications	Displays debugging information about EIGRP event notifications.

debug eigrp address-family notifications

To display debugging information about Enhanced Interior Gateway Routing Protocol (EIGRP) address family event notifications, use the **debug eigrp address-family notifications** command in privileged EXEC mode. To disable EIGRP event notification debugging, use the **no** form of this command.

debug eigrp address-family {**ipv4** [*autonomous-system-number*| **vrf** [*vrf-name*]| *ip-address*]| **ipv6** [*autonomous-system-number*| *ip-address*]} **notifications**

no debug eigrp address-family {**ipv4** [*autonomous-system-number*| **vrf** [*vrf-name*]| *ip-address*]| **ipv6** [*autonomous-system-number*| *ip-address*]} **notifications**

Syntax Description

ipv4	Enables debugging for neighbors formed using the IPv4 protocol family.
ipv6	Enables debugging for neighbors formed using the IPv6 protocol family.
autonomous-system- number	(Optional) Autonomous system number of the EIGRP routing process. If no autonomous system number is specified, debugging information is displayed for all autonomous systems.
vrf	(Optional) Enables debugging for the specified VRF.
vrf-name	(Optional) Name of the VRF address family to which the command is applied.
ip-address	(Optional) IPv4 or IPv6 address of neighbor. Specifying an address enables debugging for all entries with this address.

Command Default

EIGRP event notification debugging is disabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
15.0(1)M	This command was introduced.
12.2(33)SRE	This command was integrated into Cisco IOS Release 12.2(33)SRE.
12.2(33)XNE	This command was integrated into Cisco IOS Release 12.2(33)XNE.

Release	Modification
Cisco IOS XE Release 2.5	This command was integrated into Cisco IOS XE Release 2.5.

Usage Guidelines

Consult Cisco technical support before using this command.

Caution Use of **debug** commands can have severe performance penalties and should be used with extreme caution. For this reason, Cisco recommends that you contact Cisco technical support before enabling a **debug** command.

Examples

The following example shows how to enable EIGRP event notification debugging:

```
Router# debug eigrp address-family ipv4 notifications
*Mar 17 15:58:07.144: IP-EIGRP: Callback: reload_iptable
*Mar 17 15:58:08.148: IP-EIGRP: iptable_redistribute into eigrp AS 1
*Mar 17 15:58:12.144: IP-EIGRP: Callback: redist frm static AS 0 10.0.0.0/24
*Mar 17 15:58:12.144: into: eigrp AS 1 event: 1
*Mar 17 15:58:12.144: IP-EIGRP: Callback: redist frm static AS 0 172.16.0.0/24
*Mar 17 15:58:12.144: into: eigrp AS 1 event: 1
```

Related Commands

Command	Description
debug eigrp address-family neighbor	Displays debugging information about EIGRP service family neighbors.

debug eigrp frr

To display debugging information about Enhanced Interior Gateway Routing Protocol (EIGRP) Fast Reroute (FRR) events, use the **debug eigrp frr** command in privileged EXEC mode. To disable debugging, use the **no** form of this command.

debug eigrp frr

no debug eigrp frr

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
15.2(4)S	This command was introduced.
Cisco IOS XE Release 3.7S	This command was integrated into Cisco IOS XE Release 3.7S.

Examples

The following is sample output from the **debug eigrp frr** command. The fields in the display are self-explanatory.

```
Device# debug eigrp frr

*May 25 15:34:48.421:  EIGRP-FRR: 10.6.6.6 can not be a LFA
*May 25 15:34:48.421:  EIGRP-FRR: 192.168.1.0/24 is having 1 available paths and 1 successors
*May 25 15:34:48.422:  EIGRP-FRR: 192.168.3.0/24 is having 1 available paths and 1 successors
*May 25 15:34:48.422:  EIGRP-FRR: 10.3.3.3 can not be a LFA
*May 25 15:34:48.422:  EIGRP-FRR: 10.6.6.6 can not be a LFA
*May 25 15:34:48.422:  EIGRP-FRR: 192.168.1.0/24 is having 1 available paths and 1 successors
*May 25 15:34:48.422:  EIGRP-FRR: 192.168.2.0/24 is having 1 available paths and 1 successors
*May 25 15:34:48.422:  EIGRP-FRR: 192.168.3.0/24 is having 2 available paths and 2 successors
*May 25 15:34:48.435:  EIGRP-FRR: 10.5.5.0/24 is having 1 available paths and 1 successors
*May 25 15:34:48.435:  EIGRP-FRR: 10.1.1.2 can not be a LFA
*May 25 15:34:48.435:  EIGRP-FRR: 10.3.3.3 can not be a LFA
*May 25 15:34:48.435:  EIGRP-FRR: 10.6.6.6 can not be a LFA
*May 25 15:34:48.435:  EIGRP-FRR: 192.168.1.0/24 is having 1 available paths and 1 successors
*May 25 15:34:48.435:  EIGRP-FRR: 192.168.2.0/24 is having 2 available paths and 1 successors
*May 25 15:34:48.435:  EIGRP-FRR: Number of LFA are 1
*May 25 15:34:48.435:  EIGRP-FRR: Going to update repair path for 192.168.2.0/24 with
10.2.3.3
*May 25 15:34:48.435:  EIGRP-IPv4-FRR: Repair path for 192.168.2.0/24 with  nexthop 10.2.3.3
  has been updated
*May 25 15:34:48.436:  EIGRP-FRR: 192.168.3.0/24 is having 3 available paths and 1 successors
*May 25 15:34:48.436:  EIGRP-FRR: Number of LFA are 2
*May 25 15:34:48.436:  EIGRP-FRR: SRLG Disjoint
*May 25 15:34:48.436:  EIGRP-FRR: Going to update repair path for 192.168.3.0/24 with
10.3.3.1
*May 25 15:34:48.436:  EIGRP-IPv4-FRR: Repair path for 192.168.3.0/24 with  nexthop 10.3.3.1
  has been updated
*May 25 15:34:48.446:  EIGRP-FRR: 10.10.10.0/24 is having 1 available paths and 1 successors
*May 25 15:34:48.446:  EIGRP-FRR: 10.4.4.4 can not be a LFA
*May 25 15:34:48.446:  EIGRP-FRR: 10.1.1.2 can not be a LFA
*May 25 15:34:48.446:  EIGRP-FRR: 10.3.3.3 can not be a LFA
```

```
*May 25 15:34:48.446:  EIGRP-FRR: 10.6.6.6 can not be a LFA
*May 25 15:34:48.446:  EIGRP-FRR: 192.168.1.0/24 is having 1 available paths and 1 successors
*May 25 15:34:48.447:  EIGRP-FRR: 192.168.3.0/24 is having 4 available paths and 1 successors
*May 25 15:34:48.447:  EIGRP-FRR: Number of LFA are 3
*May 25 15:34:48.447:  EIGRP-FRR: SRLG Disjoint
*May 25 15:34:48.447:  EIGRP-FRR: Going to update repair path for 192.168.3.0/24 with
10.4.3.1
*May 25 15:34:48.447:  EIGRP-IPv4-FRR: Repair path for 192.168.3.0/24 with nexthop 10.4.3.4
 has been updated
RTR-1#
*May 25 15:34:48.447:  EIGRP-IPv4-FRR: Repair path for 192.168.3.0/24 with nexthop 10.2.3.3
 has been deleted
```

debug eigrp fsm

To display debugging information about Enhanced Interior Gateway Routing Protocol (EIGRP) feasible successor metrics (FSMs), 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.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(7)T	This command was introduced.
12.4(6)T	Support for IPv6 was added.
12.2(33)SRB	This command was integrated into Cisco IOS Release 12.2(33)SRB.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.
Cisco IOS XE Release 2.1	This command was introduced on Cisco ASR 1000 Series Routers.

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

siatimer	(Optional) Stuck-in-active (SIA) timer messages.
static	(Optional) Static routes.

Command Default

Debugging for EIGRP neighbors is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(7)T	This command was introduced.
12.4(6)T	Support for IPv6 was added.
12.2(33)SRB	This command was integrated into Cisco IOS Release 12.2(33)SRB.
12.2(37)SE	This command was integrated into Cisco IOS Release 12.2(37)SE.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.
Cisco IOS XE Release 2.1	This command was introduced on Cisco ASR 1000 Series Routers.

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

Command	Description
neighbor	Defines a neighboring router with which to exchange routing information.
show ip eigrp neighbors	Displays EIGRP neighbors.
show ipv6 eigrp neighbors	Displays IPv6 EIGRP neighbors.

debug eigrp notifications

To debug notifications sent from the L2L3 API interface, use the **debug eigrp notifications**command in privileged EXEC mode. To turn off debugging, use the **no** form of this command.

debug eigrp notifications {**rib**| **interface**}

Syntax Description

rib	Captures notifications from the routing information base (RIB)
interface	Captures notifications from the interface.

Command Default

Debugging of EIGRP notifications for the L2L3 API interface is not enabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.4(15)XF	This command was introduced.
12.4(15)T	This command was integrated into Cisco IOS Release 12.4(15)T.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.
12.2(33)SRE	This command was integrated into Cisco IOS Release 12.2(33)SRE.

Usage Guidelines

Consult Cisco technical support before using this command.

Caution Use of **debug** commands can have severe performance penalties and should be used with extreme caution. For this reason, Cisco recommends that you contact Cisco technical support before enabling a **debug** command.

Examples

The following example displays information about the L2L3 API Interface:

```
Router# debug eigrp notifications rib
```

Related Commands

Command	Description
eigrp interface	Sets a threshold value to minimize hysteresis in a router-to-radio configuration.
interface vmi	Creates a virtual multipoint interface (VMI) that can be configured and applied dynamically.

debug eigrp nsf

To display nonstop forwarding (NSF) events in the console of an NSF-aware or NSF-capable 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.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(15)T	This command was introduced.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.
15.0(1)M	This command was integrated into Cisco IOS Release 15.0(1)M.
Cisco IOS XE Release 3.6S	This command was modified. Support for IPv6 and IPv6 VPN Routing and Forwarding (VRF) was added.
15.2(2)S	This command was modified. Support for IPv6 and IPv6 VRF was added.
15.2(1)E	This command was integrated into Cisco IOS Release 15.2(1)E.

Usage Guidelines

The output from the **debug eigrp nsf** command displays NSF-specific events. The **debug eigrp nsf** command can be issued on either an NSF-capable or an NSF-aware router.

Examples

The following example shows how to enable the Enhanced Interior Gateway Routing Protocol (EIGRP) NSF debugging and display information about neighbor devices:

```
Device# debug eigrp nsf

EIGRP NSF debugging is on
Device# show ip eigrp neighbors detail

EIGRP-IPv4 Neighbors for AS(100)
H   Address                 Interface       Hold Uptime   SRTT   RTO  Q  Seq
                                            (sec)         (ms)       Cnt Num
```

```
0   10.1.2.1                Et1/0             11 00:00:25   10   200  0  5
   Version 5.1/3.0, Retrans: 2, Retries: 0, Prefixes: 1
   Topology-ids from peer - 0
!
*Sep 23 18:57:19.423: %DUAL-5-NBRCHANGE: EIGRP-IPv4 100: Neighbor 10.1.2.1 (Ethernet1/0)
is resync: peer graceful-restart
*Sep 23 18:57:19.423: EIGRP: NSF: AS100, NSF or GR initiated by 10.1.2.1, flags 0x4:(RS)
*Sep 23 18:57:36.028: EIGRP: NSF: AS100, Receive EOT from 10.1.2.1, Flags 0x8:(EOT)
*Sep 23 18:57:36.028: EIGRP: NSF: route hold timer set to flush stale routes
*Sep 23 18:57:36.038: EIGRP: NSF: AS100. route hold timer expiry
*Sep 23 18:57:36.038: EIGRP: NSF: EIGRP-IPv4: Search for stale routes from 10.1.2.1
!
Device# show ip eigrp neighbors detail

EIGRP-IPv4 Neighbors for AS(100)
H   Address                 Interface       Hold Uptime   SRTT   RTO  Q  Seq
                                            (sec)         (ms)       Cnt Num
0   10.1.2.1                 Et1/0             11 00:02:31   12   200  0  6
   Time since Restart 00:01:34
   Version 5.1/3.0, Retrans: 2, Retries: 0, Prefixes: 1
   Topology-ids from peer - 0
```

The following sample output is displayed when a router is unable to handle an event with NSF-Awareness:

```
*Jan 23 18:59:56.040: EIGRP: NSF: AS100: Checking if Graceful Restart is possible with
neighbor 1.1.2.1, peer_down reason 'peer restarted'
*Jan 23 18:59:56.040: EIGRP: NSF: Not possible: 'peer_down was called with a HARD resync
flag'
*Jan 23 18:59:56.040: %DUAL-5-NBRCHANGE: EIGRP-IPv6 100: Neighbor 10.1.2.1 (Ethernet1/0)
is down: peer restarted
*Jan 23 19:00:00.170: %DUAL-5-NBRCHANGE: EIGRP-IPv6 100: Neighbor 10.1.2.1 (Ethernet1/0)
is up: new adjacency
*Jan 23 19:00:00.170: EIGRP: NSF: Enqueuing NULL update to 10.1.2.1, flags 0x1:(INIT)
```

debug eigrp packets

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

debug eigrp packets [**SIAquery**| **SIAreply**| **ack**| **hello**| **ipxsap**| **probe**| **query**| **reply**| **request**| **retry**| **stub**| **terse**| **update**| **verbose**] [**detail**]

no debug eigrp packets

Syntax Description

SIAquery	(Optional) Enables debugging of stuck-in-active (SIA) query messages.
SIAreply	(Optional) Enables debugging of SIA reply messages.
ack	(Optional) Enables debugging of EIGRP acknowledgment packets.
hello	(Optional) Enables debugging of EIGRP hello packets.
ipxsap	(Optional) Enables debugging of EIGRP Internetwork Packet Exchange (IPX) Service Advertisement Protocol (SAP) packets.
probe	(Optional) Enables debugging of EIGRP probe packets.
query	(Optional) Enables debugging of EIGRP query packets.
reply	(Optional) Enables debugging of EIGRP reply packets.
request	(Optional) Enables debugging of EIGRP request packets.
retry	(Optional) Enables debugging of EIGRP retry packets.
stub	(Optional) Enables debugging of EIGRP stub packets.
terse	(Optional) Enables debugging of all EIGRP packets except hello packets.
update	(Optional) Enables debugging of EIGRP update packets.
verbose	(Optional) Enables debugging of all EIGRP packets.
detail	(Optional) Enables detailed debugging of packets, depending on any of the keywords specified in the command.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.0(7)T	This command was introduced.
12.4(6)T	This command was modified. Support for IPv6 was added.
12.2(33)SRB	This command was integrated into Cisco IOS Release 12.2(33)SRB.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.
Cisco IOS XE Release 2.1	This command was implemented on Cisco ASR 1000 Series Aggregation Services Routers.
15.2(4)S	This command was modified. The **detail** keyword was added.
Cisco IOS XE Release 3.7S	This command was modified. The **detail** keyword was added.

Usage Guidelines

Use the **debug eigrp packets** command to analyze messages traveling between local and remote hosts.

Note Although this command accepts a number of keywords, we do not recommend their use.

Examples

The following sample output from the **debug eigrp packets** command displays the transmission and receipt of 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. Wherever applicable, the network-layer address of the neighboring router is also included.

```
Device# debug eigrp packets

*May 24 15:33:10.255: EIGRP: Sending HELLO on Ethernet0/1
*May 24 15:33:10.255:   AS 109, Flags 0x0, Seq 0, Ack 0
*May 24 15:33:10.255: EIGRP: Sending HELLO on Ethernet0/1
*May 24 15:33:10.255:   AS 109, Flags 0x0, Seq 0, Ack 0
*May 24 15:33:10.255: EIGRP: Sending HELLO on Ethernet0/1
*May 24 15:33:10.255:   AS 109, Flags 0x0, Seq 0, Ack 0
*May 24 15:33:10.255: EIGRP: Received UPDATE on Ethernet0/1 from 192.168.78.24,
*May 24 15:33:10.255:   AS 109, Flags 0x1, Seq 1, Ack 0
*May 24 15:33:10.255: EIGRP: Sending HELLO/ACK on Ethernet0/1 to 192.168.78.24,
*May 24 15:33:10.255:   AS 109, Flags 0x0, Seq 0, Ack 1
*May 24 15:33:10.255: EIGRP: Sending HELLO/ACK on Ethernet0/1 to 192.168.78.24,
*May 24 15:33:10.255:   AS 109, Flags 0x0, Seq 0, Ack 1
*May 24 15:33:10.255: EIGRP: Received UPDATE on Ethernet0/1 from 192.168.78.24,
*May 24 15:33:10.255    AS 109, Flags 0x0, Seq 2, Ack 0
```

The following sample output from the **debug eigrp packets hello detail** command displays debugging details of EIGRP hello packets:

```
Device# debug eigrp packets hello detail
```

```
*May 25 15:33:10.255: EIGRP: Sending HELLO on Et1/0 - paklen 20
*May 25 15:33:10.255:   AS 1, Flags 0x0:(NULL), Seq 0/0 interfaceQ 0/0 iidbQ un/rely 0/0
*May 25 15:33:10.255:   {type = 1, length = 12}
*May 25 15:33:10.255:   {vector = {
*May 25 15:33:10.255:             {01000100 0000000F}
*May 25 15:33:10.255:   }
*May 25 15:33:10.255:   {type = 4, length = 8}
*May 25 15:33:10.255:   {vector = {
*May 25 15:33:10.255:             {0B000200}
*May 25 15:33:10.255:   }
*May 25 15:33:10.695: EIGRP: Sending HELLO on Et0/2 - paklen 20
*May 25 15:33:10.695:   AS 1, Flags 0x0:(NULL), Seq 0/0 interfaceQ 0/0 iidbQ un/rely 0/0
*May 25 15:33:10.695:   {type = 1, length = 12}
*May 25 15:33:10.695:   {vector = {
*May 25 15:33:10.695:             {01000100 0000000F}
*May 25 15:33:10.695:   }
*May 25 15:33:10.695:   {type = 4, length = 8}
*May 25 15:33:10.695:   {vector = {
*May 25 15:33:10.695:             {0B000200}
*May 25 15:33:10.695:   }
```

The table below describes the significant fields shown in the displays.

Table 1: debug eigrp packets Field Descriptions

Field	Description
EIGRP:	EIGRP packet information.
AS 109	Autonomous system number.
Flags 0x0	A flag of 1 means that the sending device is informing the receiving device that this is the first packet that is being sent to the receiver. A flag of 2 is a multicast that should be conditionally received by devices that have been previously set with the conditionally receive (CR) bit. This bit gets set when the sender of the multicast has previously sent a sequence packet explicitly telling the packet to set the CR bit. A flag of 0X0 means no flags are set on the packet.
HELLO	Hello packets are neighbor discovery packets. They are used to determine whether neighbors are still alive. As long as neighbors receive the hello packets that a device is sending, the neighbors validate the device and any routing information sent by the device.

debug eigrp service-family

To troubleshoot an Enhanced Interior Gateway Routing Protocol (EIGRP) service-family external client, client, neighbor, notification, topology, or a VRF instance, use the **debug eigrp service-family**command in privileged EXEC mode.

{**debug eigrp service-family** [**external-client** {**client** *client-label*| **messages** [*client-label*]| **protocol** [*client-label*]}]| {**ipv4| ipv6**} [[**vrf** *vrf-name*| *autonomous-system-number*| *service-instance-number*]| **client** *client-label*| **neighbor** *neighbor-ip-address*| **notifications topology** *service-instance-number*]}

Syntax Description

external-client	(Optional) Displays information for a Cisco SAF External Client.
client	Displays information for managing clients and TCP connections.
messages	(Optional) Reliability metric. The range is 0 to 255, entered in increments of 2.5 where 255 is 100-percent reliable.
protocol	(Optional) Displays information on an external-client protocol.
client-label	(Optional) Displays a **client**, **message**, or **protocol** debug for the specified Cisco SAF External Client.
ipv4	Specifies the IP Version 4 address family for this debug.
ipv6	Specifies the IP Version 6 address family for this debug.
vrf	(Optional) Specifies all virtual routing forwarding (VRF) instance tables or a specific VRF table for an IP address.
vrf-name	(Optional) Specifies a VRF table for an IP address.
autonomous-system-number	The Autonomous system number.
service-instance- number	(Optional) Service-instance number between 1 and 65535. Service instance numbers display as: service:subservice:instance.instance.instance.instance.
client	(Optional) Displays EIGRP client information.
client-label	(Optional) A specific client.

neighbors	(Optional) Displays EIGRP neighbor debugging information.
neighbor-ip- address	(Optional) The IP address of the neighbor.
notifications	(Optional) Displays EIGRP notification debugging information.
topology	(Optional) Specifies a service topology.
service-instance- number	(Optional) Service-instance number between 1 and 65535. Topology service instance numbers display as: service:subservice:instance.instance.instance.instance.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
15.0(1)M	This command was introduced.
12.2(33)SRE	This command was integrated into Cisco IOS Release 12.2(33)SRE.
12.2(33)XNE	This command was integrated into Cisco IOS Release 12.2(33)XNE.
Cisco IOS XE Release 2.5	This command was integrated into Cisco IOS XE Release 2.5.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.
15.2(1)S	This command was deprecated in Cisco IOS Release 15.2(1)S and replaced by the debug service-routing xmcp command.
Cisco IOS XE Release 3.5S	This command was deprecated in Cisco IOS XE Release 3.5S and replaced by the debug service-routing xmcp command.
15.2(2)T	This command was deprecated in Cisco IOS Release 15.2(2)T and replaced by the debug service-routing xmcp command.

Usage Guidelines

Use the **debug eigrp service-family external-client client**command to display information to help manage clients and TCP connections. Use the **debug eigrp service-family external-client messages**command to display message content and decoded messages. Use the **debug eigrp service-family external-client protocol**command to display encode and decode information to help manage the interaction with the Cisco SAF internal API.

Note Using the **debug eigrp service-family ipv6** commands requires an IPv6-enabled SAF client, which currently does not exist.

Examples

```
The following is sample output of a Cisco SAF External-Client debugging message:
Router# debug eigrp service-family external-client messages

*Jun 11 14:25:10.051: 2 found c1 c1
*Jun 11 14:25:10.051: SAF-EC: 100 byte message from c1
*Jun 11 14:25:10.051: 0001 0050 7F5A 9BC7 D285 A1D8 3C54 552F 37AE 655B 0014 0005 2253 4146
 2200
*Jun 11 14:25:10.051: 0000 0006 0005 756E 616D 6500 0000 1005 0002 6331 0000 1003 0004 0001
 0000
*Jun 11 14:25:10.051: 1001 0002 6331 0000 1004 0004 0000 0005 0008 0014 45F4 57A9 42CF 0556
 4077
*Jun 11 14:25:10.051: 7AA3 B94A 703F 1BA3 ACA7
*Jun 11 14:25:10.051:
*Jun 11 14:25:10.051:
Class: Success Response Method: Register
*Jun 11 14:25:10.051: Packet Length: 52 Not including 20 byte Saf Header
*Jun 11 14:25:10.051: Magic Cookie: 7F5A9BC7 Transaction ID: D285A1D83C54552F37AE65
Router#5B
*Jun 11 14:25:10.051: Realm: 014: Length: 5: "SAF"
*Jun 11 14:25:10.051: Keep Alive: 1006: Length: 4: 360000
*Jun 11 14:25:10.051: Client Handle: 1002: Length: 4: 2
*Jun 11 14:25:10.051: Message Integrity: 008: Length: 20:
86839D4C64E36476D743AAF26112D28C32E3DF99
*Jun 11 14:25:10.051: 0101 0034 7F5A 9BC7 D285 A1D8 3C54 552F 37AE 655B 0014 0005 2253 4146
 2200
*Jun 11 14:25:10.051: 0000 1006 0004 0005 7E40 1002 0004 0000 0002 0008 0014 8683 9D4C 64E3
 6476
*Jun 11 14:25:10.051: D743 AAF2 6112 D28C 32E3 DF99
*Jun 11 14:25:10.055:
*Jun 11 14:25:10.055: SAF-EC: kicked timer 360000
The following is sample output of a Cisco SAF External-Client debugging protocol message:
Router# debug eigrp service-family external-client protocol

*Jun 11 14:27:11.467: SAF-EC: attribute found, type: 1005
*Jun 11 14:27:11.467: No error
*Jun 11 14:27:11.467:
Class: Request Method: Register
*Jun 11 14:27:11.467: Packet Length: 80 bytes Not including 20 byte Saf Header
*Jun 11 14:27:11.467: Magic Cookie: 7F5A9BC7 Transaction ID: 8F1F3F36EE43784D0DFABEA6
*Jun 11 14:27:11.467: Realm: 014: Length: 5: "SAF"
*Jun 11 14:27:11.467: Username: 006: Length: 5: uname
*Jun 11 14:27:11.467: Client Label: 1005: Length: 2: c1
*Jun 11 14:27:11.467: Protocol Version: 1003: Length: 4: 10000
*Jun 11 14:27:11.467: Client Name: 1001: Length: 2: c1
*Jun 11 14:27:11.467: Page Size: 1004: Length: 4: 5
Router#
*Jun 11 14:27:11.467: Message Integrity: 008: Length: 20:
AB3D7C39E4E0673B1539750D6E21A79ACFCE51F8
*Jun 11 14:27:11.467: SAF-EC: request start.
*Jun 11 14:27:11.467: SAF-EC: client successfully registered. client_handle 3
Router#
```

Related Commands

Command	Description
exit-service-family	Exits service-family configuration mode.
router eigrp	Configures the EIGRP process.

Command	Description
service-family	Specifies service-family configuration mode.

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

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

Command Default

Debugging for EIGRP transmittal messages is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.1	This command was introduced.

Release	Modification
12.4(6)T	Support for IPv6 was added.
12.2(33)SRB	This command was integrated into Cisco IOS Release 12.2(33)SRB.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.
Cisco IOS XE Release 2.1	This command was introduced on Cisco ASR 1000 Series Routers.

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 elb-pal-pd

To display debugging information related to the Ethernet Data Plane Loopback feature, use the **debug elb-pal-pd** command in the privileged EXEC mode. To disable the debugging function, use the **no** form of this command.

debug elb-pal-pd {**all | error | event**}

no debug elb-pal-pd {**all | error | event**}

Syntax Description

all	Displays all the debugging information pertaining to the Ethernet data plane loopback configuration.
error	Displays debugging information pertaining to the Ethernet data plane loopback configuration errors.
event	Displays debugging information pertaining to the Ethernet data plane loopback configuration changes.

Command Default

Debugging is disabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
15.4(1)T	This command was introduced into Cisco IOS Release 15.4(1)T.

Usage Guidelines

Use this command to display the errors that occur when the Ethernet Data Plane Loopback feature is configured on the following routers:

- Cisco 3900 Series, Cisco 2900 Series, and Cisco 1900 Series Integrated Services Routers Generation 2
- Cisco 890 Series Integrated Services Routers

Examples

The following is a sample output of the **debug elb-pal-pd event** command:

```
Device# debug elb-pal-pd event

elb pal pd event debugging is on
```

The following is a sample output of the **ethernet loopback start local interface** command:

```
Device# ethernet loopback start local interface GigabitEthernet0/2.301 external timeout
none
```

```
This is an intrusive loopback and the packets matched with the service will not  be able
to pass through.
Continue? (yes/[no]): yes
Router#
*Nov 27 20:34:37.200: elb service info in PAL: session id 1, interface GigabitEthernet0/2.301,
 si_handle 0x0,
dir Facility, session_type 3,src mac 0000.0000.0000, dest mac 0000.0000.0000, dot1q_bl ,
second_dot1q_bl , cos 8,
oui 0 etype 0, filter_option_flags 0
*Nov 27 20:34:37.200: elb_pd_sb_init
*Nov 27 20:34:37.200: cn_xfr_ge_elb_pd_loopback: on_off 1, hwidb 0x3CFFC364,
elb_pd_intf_lpbk_count 1
*Nov 27 20:34:37.200: elb_pal_pd_loopback_wrapper: elb_pal_pd_loopback action succeeded.
*Nov 27 20:34:37.200: elb_pal_pd_start_lb: filter_out_vlan 0x0000, filter_in_vlan 0x0000,
intf_out_vlan 0xFFFF,
intf_in_vlan 0xFFFF
*Nov 27 20:34:37.200: elb_pal_pd_start_lb: option_flag 0x00000000, filter_dir 1,
loopback_on_off 1, session_type 3
*Nov 27 20:34:37.200: elb_pal_pd_start_lb: elb_pal_pd_num_intf_lpbk 1
*Nov 27 20:34:37.200: %E_DLB-6-DATAPLANE_LOOPBACK_START: Ethernet Dataplane Loopback Start
 on interface
GigabitEthernet0/2.301 with session id 1
Router#
```

Related Commands

Command	Description
ethernet loopback start local interface	Starts Ethernet external loopback on the subinterface.

debug emm

To debug Embedded Menu Manager (EMM) Menu Definition Files (MDFs), use the **debug emm**command in user EXEC or privileged EXEC mode. To disable debugging, use the **no** form of this command.

debug emm

no debug emm

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled.

Command Modes

User EXEC (#) Privileged EXEC (#)

Command History

Release	Modification
12.4(20)T	This command was introduced.

Usage Guidelines

Do not run this command on the same vty as the EMM menu.

Examples

The following is sample output from the **debug emm**command. The output is described in the EMM XML Schema Definition (XSD), which is available for download at this website:

http://forums.cisco.com/eforum/servlet/EEM?page=main

```
Router# debug emm
EMM debugging is on
*Jun 10 15:45:42.043: Looking for MenuTitle, parent = Menu
*Jun 10 15:45:42.063: Looking for GlobalTCL, parent = Menu
*Jun 10 15:45:42.083: Looking for MenuTitle, parent = Menu
```

Related Commands

Command	Description
emm	Loads and launches preconfigured MDFs or launches loaded preconfigured EMM menus.
emm clear	Changes the terminal clear-screen escape sequence.
show mdf	Displays loaded preconfigured MDFs.

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

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

Command Default

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.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXI	This command was integrated into Cisco IOS Release 12.2(33)SXI.

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_j]'
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

Command	Description
debug eap	Displays information about EAP messages.
debug ip admission eapoudp	Displays information about EAPoUDP network admission control events.

debug epc

To enable debugging of embedded packet capture (EPC) infrastructure, use the **debug epc** command in privileged EXEC mode. To disable debugging of packet capture infrastructure, use the **no** form of this command.

debug epc {**capture-point**| **provision**}

no debug epc {**capture-point**| **provision**}

Syntax Description

capture-point	Specifies debugging of the capture point configuration.
provisioning	Specifies debugging of capture provisioning.

Command Default

Debug messages are not logged.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS XE Release 3.7S	This command was introduced.

Examples

The following example shows how to enable debugging of the Embedded Packet Capture (EPC) capture point:

```
Device# debug epc capture-point

EPC capture point operations debugging is on
Device# monitor capture mycap start

*Jul  4 14:17:15.463: EPC CP:  Starting the capture cap1
*Jul  4 14:17:15.463: EPC CP:  (brief=3, detailed=4, dump=5) = 0
*Jul  4 14:17:15.463: EPC CP:  final check before activation
*Jul  4 14:17:15.463: EPC CP:  setting up c3pl infra
*Jul  4 14:17:15.463: EPC CP:  Setup c3pl acl-class-policy
*Jul  4 14:17:15.463: EPC CP:  Creating a class
*Jul  4 14:17:15.464: EPC CP:  Creating a class : Successful
*Jul  4 14:17:15.464: EPC CP:  class-map Created
*Jul  4 14:17:15.464: EPC CP: creating  policy-name epc_policy_cap1
*Jul  4 14:17:15.464: EPC CP:  Creating Policy epc_policy_cap1 of type 49 and client type
21
*Jul  4 14:17:15.464: EPC CP:  Storing a Policy
*Jul  4 14:17:15.464: EPC CP:  calling ppm_store_policy with epc_policy
*Jul  4 14:17:15.464: EPC CP:  Creating Policy : Successful
*Jul  4 14:17:15.464: EPC CP:  policy-map created
*Jul  4 14:17:15.464: EPC CP:  creating filter for ANY
*Jul  4 14:17:15.464: EPC CP:  Adding acl to class : Successful
*Jul  4 14:17:15.464: EPC CP:  Setup c3pl class to policy
*Jul  4 14:17:15.464: EPC CP:  Attaching Class to Policy
*Jul  4 14:17:15.464: EPC CP:  Attaching epc_class_cap1 to epc_policy_cap1
```

```
*Jul  4 14:17:15.464: EPC CP:  Attaching Class to Policy : Successful
*Jul  4 14:17:15.464: EPC CP:  setting up c3pl qos
*Jul  4 14:17:15.464: EPC CP:  DBG> Set packet rate limit to 1000
*Jul  4 14:17:15.464: EPC CP: creating action for policy_map epc_policy_cap1 class_map
epc_class_cap1
*Jul  4 14:17:15.464: EPC CP:  DBG> Set packet rate limit to 1000
*Jul  4 14:17:15.464: EPC CP:  Activating Interface GigabitEthernet1/0/1  direction both
*Jul  4 14:17:15.464: EPC CP: Id attached 0
*Jul  4 14:17:15.464: EPC CP:  inserting into active lists
*Jul  4 14:17:15.464: EPC CP: Id attached 0
*Jul  4 14:17:15.465: EPC CP:  inserting into active lists
*Jul  4 14:17:15.465: EPC CP:  Activating Vlan
*Jul  4 14:17:15.465: EPC CP:  Deleting all temp interfaces
*Jul  4 14:17:15.465: %BUFCAP-6-ENABLE: Capture Point cap1 enabled.
*Jul  4 14:17:15.465: EPC CP:  Active Capture 1
Device# monitor capture mycap stop

*Jul  4 14:17:31.963: EPC CP:  Stopping the capture cap1
*Jul  4 14:17:31.963: EPC CP: Warning: unable to unbind capture cap1
*Jul  4 14:17:31.963: EPC CP:  Deactivating policy-map
*Jul  4 14:17:31.963: EPC CP:  Policy epc_policy_cap1
*Jul  4 14:17:31.964: EPC CP:  Deactivating policy-map Successful
*Jul  4 14:17:31.964: EPC CP:  removing povision feature
*Jul  4 14:17:31.964: EPC CP:  Found action for policy-map epc_policy_cap1 class-map
epc_class_cap1
*Jul  4 14:17:31.964: EPC CP:  cleanning up c3pl infra
*Jul  4 14:17:31.964: EPC CP:  Removing Class epc_class_cap1 from Policy
*Jul  4 14:17:31.964: EPC CP:  Removing Class from epc_policy_cap1
*Jul  4 14:17:31.964: EPC CP:  Successfully removed
*Jul  4 14:17:31.964: EPC CP:  Removing acl mac from class
*Jul  4 14:17:31.964: EPC CP:  Removing acl from class : Successful
*Jul  4 14:17:31.964: EPC CP:  Removing all policies
*Jul  4 14:17:31.964: EPC CP:  Removing Policy  epc_policy_cap1
*Jul  4 14:17:31.964: EPC CP:  Removing Policy : Successful
*Jul  4 14:17:31.964: EPC CP:  Removing class epc_class_cap1
*Jul  4 14:17:31.965: EPC CP:  Removing class : Successful
*Jul  4 14:17:31.965: %BUFCAP-6-DISABLE: Capture Point cap1 disabled.
*Jul  4 14:17:31.965: EPC CP:  Active Capture 0
```

The following example shows how to enable debugging of EPC provisioning:

```
Device# debug epc provision

EPC provisioning debugging is on
Device# monitor capture mycap start

*Jul  4 14:17:54.991: EPC PROV: No action found for policy-map epc_policy_cap1 class-map
epc_class_cap1
*Jul  4 14:17:54.991: EPC PROV:
*Jul  4 14:17:54.991: Attempting to install service policy epc_policy_cap1
*Jul  4 14:17:54.992: EPC PROV: Attached service policy to epc idb subblock
*Jul  4 14:17:54.992: EPC PROV: Successful. Create feature object
*Jul  4 14:17:54.992: EPC PROV:
*Jul  4 14:17:54.992: Attempting to install service policy epc_policy_cap1
*Jul  4 14:17:54.992: EPC PROV: Successful. Create feature object
*Jul  4 14:17:54.992: %BUFCAP-6-ENABLE: Capture Point cap1 enabled.
Device# monitor capture mycap stop

*Jul  4 14:18:02.503: EPC PROV: Successful. Remove feature object
*Jul  4 14:18:02.504: EPC PROV: Successful. Remove feature object
*Jul  4 14:18:02.504: EPC PROV: Destroyed epc idb subblock
*Jul  4 14:18:02.504: EPC PROV: Found action for policy-map epc_policy_cap1 class-map
epc_class_cap1
*Jul  4 14:18:02.504: EPC PROV: Deleting EPC action
*Jul  4 14:18:02.504: EPC PROV: Successful. CLASS_REMOVE, policy-map epc_policy_cap1, class
 epc_class_cap1
*Jul  4 14:18:02.504: %BUFCAP-6-DISABLE: Capture Point cap1 disabled.
```

Related Commands

Command	Description
monitor capture start	Starts the capture of packet data at a traffic trace point into a buffer.
monitor capture stop	Stops the capture of packet data at a traffic trace point.

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

mac-address	(Optional) Defines the MAC address of the Cisco IP phone.
mac-address	(Optional) Specifies the MAC address of the Cisco IP phone.

Command Default

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

debug ephone blf

To display debugging information for Busy Lamp Field (BLF) presence features, use the **debug ephone blf** command in privileged EXEC mode. To disable debugging, use the **no** form of this command.

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

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

Syntax Description

mac-address *mac-address*	(Optional) Specifies the MAC address of a specific IP phone.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.4(11)XJ	This command was introduced.
12.4(15)T	This command was integrated into Cisco IOS Release 12.4(15)T.

Usage Guidelines

Use this command for troubleshooting BLF speed-dial and BLF call-list features for phones in a presence service.

Examples

The following is sample output from the **debug ephone blf** command.

```
Router# debug ephone blf
EPHONE BLF debugging is enabled
*Sep  4 07:18:26.307: skinny_asnl_callback: subID 16 type 4
*Sep  4 07:18:26.307: ASNL_RESP_NOTIFY_INDICATION
*Sep  4 07:18:26.307: ephone-1[1]:ASNL notify indication message, feature index 4, subID
[16]
*Sep  4 07:18:26.307: ephone-1[1]:line status 6, subID [16]
*Sep  4 07:18:26.307: ephone-1[1]:StationFeatureStatV2Message sent, status 2
*Sep  4 07:18:26.307: skinny_asnl_callback: subID 23 type 4
*Sep  4 07:18:26.307: ASNL_RESP_NOTIFY_INDICATION
*Sep  4 07:18:26.307: ephone-2[2]:ASNL notify indication message, feature index 2, subID
[23]
*Sep  4 07:18:26.311: ephone-2[2]:line status 6, subID [23]
*Sep  4 07:18:26.311: ephone-2[2]:StationFeatureStatV2Message sent, status 2
*Sep  4 07:18:28.951: skinny_asnl_callback: subID 16 type 4
*Sep  4 07:18:28.951: ASNL_RESP_NOTIFY_INDICATION
*Sep  4 07:18:28.951: ephone-1[1]:ASNL notify indication message, feature index 4, subID
[16]
*Sep  4 07:18:28.951: ephone-1[1]:line status 1, subID [16]
*Sep  4 07:18:28.951: ephone-1[1]:StationFeatureStatV2Message sent, status 1
*Sep  4 07:18:28.951: skinny_asnl_callback: subID 23 type 4
*Sep  4 07:18:28.951: ASNL_RESP_NOTIFY_INDICATION
*Sep  4 07:18:28.951: ephone-2[2]:ASNL notify indication message, feature index 2, subID
[23]
```

```
*Sep  4 07:18:28.951: ephone-2[2]:line status 1, subID [23]
*Sep  4 07:18:28.951: ephone-2[2]:StationFeatureStatV2Message sent, status 1
```

Related Commands

Command	Description
blf-speed-dial	Enables BLF monitoring for a speed-dial number on a phone registered to Cisco Unified CME.
presence call-list	Enables BLF monitoring for call lists and directories on phones registered to a Cisco Unified CME router.
show presence global	Displays configuration information about the presence service.
show presence subscription	Displays information about active presence subscriptions.

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
debug ephone state	Displays call state information.
show debugging	Displays information about the types of debugging that are enabled for your router.
show ephone	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

mac-address	(Optional) Defines the MAC address of the Cisco IP phone.
mac-address	(Optional) Specifies the MAC address of the Cisco IP phone.

Command Default

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

Command	Description
show debugging	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

mac-address	(Optional) Defines the MAC address of the Cisco IP phone.
mac-address	(Optional) Specifies the MAC address of the Cisco IP phone.

Command Default

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

debug ephone extension-assigner

To display status messages produced by the extension assigner application, use the **debug ephone extension-assigner** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ephone extension-assigner

no debug ephone extension-assigner

Syntax Description

This command has no arguments or keywords.

Command Default

Debug ephone extension-assigner is disabled.

Command Modes

Privileged EXEC

Command History

Cisco IOS Release	Cisco Product	Modification
12.4(4)XC4	Cisco Unified CME 4.0(3)	This command was introduced.
12.4(11)XJ	Cisco Unified CME 4.1	This command was introduced.
12.4(15)T	Cisco Unified CME 4.1	This command was integrated into Cisco IOS Release 12.4(15)T.

Usage Guidelines

This command displays status messages produced by the extension assigner application, including messages related to the functions performed by the following Tcl commands:

- phone query--Verifies whether the ephone tag has been assigned a MAC address.
- phone assign--Binds the MAC address from the caller's phone to a preexisting ephone template.
- phone unassign--Removes the MAC address from the ephone tag.

Before using this command, you must load the Tcl script for the extension assigner application.

Examples

The following is sample output of extension assigner debugging as the extension assigner application queries phones for their status and issues commands to assign or unassign extension numbers.

```
*Jun 9 19:08:10.627: ephone_query: inCallID=47, tag=4, ephone_tag=4
*Jun 9 19:08:10.627: extAssigner_IsEphoneMacPreset: ephone_tag = 4, ipKeyswitch.max_ephones
 = 96
*Jun 9 19:08:10.627: extAssigner_IsEphoneMacPreset: ephone_ptr->mac_addr_str = 000B46BDE075,
 MAC_EXT_RESERVED_VALUE = 02EAEAEA0000
*Jun 9 19:08:10.627: SkinnyGetActivePhoneIndexFromCallid: callID = 47
*Jun 9 19:08:10.627: SkinnyGetActivePhoneIndexFromCallid: vdbPtr->physical_interface_type
(26); CV_VOICE_EFXS (26)
```

```
*Jun 9 19:08:10.627: SkinnyGetActivePhoneIndexFromCallid: vdbPtr->type (6); CC_IF_TELEPHONY
 (6)
*Jun 9 19:08:10.627: SkinnyGetActivePhoneIndexFromCallid: htsp->sig_type (26); CV_VOICE_EFXS
 (26)
*Jun 9 19:08:10.627: SkinnyGetActivePhoneIndexFromCallid: dn = 4, chan = 1
*Jun 9 19:08:10.627: ephone_query: EXTASSIGNER_RC_SLOT_ASSIGNED_TO_CALLING_PHONE
*Jun 9 19:08:22.763: ephone_unassign: inCallID=47, tag=4, ephone_tag=4
*Jun 9 19:08:22.763: extAssigner_IsEphoneMacPreset: ephone_tag = 4, ipKeyswitch.max_ephones
 = 96
*Jun 9 19:08:22.763: extAssigner_IsEphoneMacPreset: ephone_ptr->mac_addr_str = 000B46BDE075,
 MAC_EXT_RESERVED_VALUE = 02EAEAEA000
*Jun 9 19:08:22.763: is_ephone_auto_assigned: button-1 dn_tag=4
*Jun 9 19:08:22.763: is_ephone_auto_assigned: NO
*Jun 9 19:08:22.763: SkinnyGetActivePhoneIndexFromCallid: callID = 47
*Jun 9 19:08:22.763: SkinnyGetActivePhoneIndexFromCallid: vdbPtr->physical_interface_type
(26); CV_VOICE_EFXS (26)
*Jun 9 19:08:22.767: SkinnyGetActivePhoneIndexFromCallid: vdbPtr->type (6); CC_IF_TELEPHONY
 (6)
*Jun 9 19:08:22.767: SkinnyGetActivePhoneIndexFromCallid: htsp->sig_type (26); CV_VOICE_EFXS
 (26)
*Jun 9 19:08:22.767: SkinnyGetActivePhoneIndexFromCallid: dn = 4, chan = 1
*Jun 9 19:08:29.795: ephone-4[8]:fStationOnHookMessage: Extension Assigner request restart,
 cmd=2, new mac=02EAEAEA0004, ephone_tag=4
*Jun 9 19:08:30.063: %IPPHONE-6-UNREGISTER_NORMAL: ephone-4:SEP000B46BDE075 IP:5.5.0.1
Socket:8 DeviceType:Phone has unregistered normally.
*Jun 9 19:08:30.063: ephone-4[8][SEP000B46BDE075]:extAssigner_assign: new mac=02EAEAEA0004,
 ephone-tag=4
*Jun 9 19:08:30.063: extAssigner_simple_assign: mac=02EAEAEA0004, tag=4
*Jun 9 19:08:30.063: ephone_updateCNF: update cnf_file ephone_tag=4
*Jun 9 19:08:30.063: extAssigner_assign: restart again (mac=02EAEAEA0004) ephone_tag=4
*Jun 9 19:08:30.131: %IPPHONE-6-REG_ALARM: 23: Name=SEP000B46BDE075 Load=8.0(2.0)
Last=Reset-Restart
*Jun 9 19:08:30.135: %IPPHONE-6-REGISTER_NEW: ephone-7:SEP000B46BDE075 IP:5.5.0.1 Socket:10
 DeviceType:Phone has registered.
*Jun 9 19:08:30.503: %IPPHONE-6-UNREGISTER_NORMAL: ephone-7:SEP000B46BDE075 IP:5.5.0.1
Socket:10 DeviceType:Phone has unregistered normally.
*Jun 9 19:08:43.127: %IPPHONE-6-REG_ALARM: 22: Name=SEP000B46BDE075 Load=8.0(2.0)
Last=Reset-Reset
*Jun 9 19:08:43.131: %IPPHONE-6-REGISTER: ephone-7:SEP000B46BDE075 IP:5.5.0.1 Socket:13
DeviceType:Phone has registered.
```

Related Commands

Command	Description
debug ephone state	Sets state debugging for Cisco IP phones.
debug voip application script	Displays status messages produced by voice over IP application scripts.

debug ephone lpcor

To display debugging information for calls using the logical partitioning class of restriction (LPCOR) feature, use the **debug ephone lpcor** command in privileged EXEC mode. To disable debugging, use the **no** form of this command.

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

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

Syntax Description

mac-address *mac-address*	(Optional) Specifies the MAC address of a specific IP phone.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
15.0(1)XA	This command was introduced.
15.1(1)T	This command was integrated into Cisco IOS Release 15.1(1)T.

Usage Guidelines

Use this command for troubleshooting LPCOR calls to phones in a Cisco Unified CME system.

If the **mac-address** keyword is not used, this command debugs all phones that are registered to the Cisco Unified CME router. You can disable debugging for specific phones by using the **mac-address** keyword with the **no** form of this command.

Examples

The following is sample output from the **debug ephone lpcor** command for a call between ephone-1 and ephone-2 that was blocked by LPCOR policy validation:

```
Router# debug ephone lpcor
*Jun 24 11:23:45.599: ephone-1[0/3][SEP003094C25F38]:ephone_get_lpcor_index: dir 0
*Jun 24 11:23:46.603: ephone-2[1/2][SEP0021A02DB62A]:ephone_get_lpcor_index: dir 1
```

Related Commands

Command	Description
debug voip application lpcor	Enables debugging of the LPCOR application system.
debug voip lpcor	Displays debugging information for the LPCOR feature.

Command	Description
lpcor incoming	Associates an incoming call with a LPCOR resource-group policy.
lpcor outgoing	Associates an outgoing call with a LPCOR resource-group policy.
show ephone	Displays information about phones registered to Cisco Unified CME.
show voice lpcor policy	Displays the LPCOR policy for the specified resource group.

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

mac-address	(Optional) Defines the MAC address of the Cisco IP phone.
mac-address	(Optional) Specifies the MAC address of the Cisco IP phone.

Command Default

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
debug ephone alarm	Sets SkinnyStation alarm messages debugging for the Cisco IP phone.
debug ephone detail	Sets detail debugging for the Cisco IP phone.
debug ephone error	Sets error debugging for the Cisco IP phone.
debug ephone loopback	Sets MWI debugging for the Cisco IP phone.
debug ephone pak	Provides voice packet level debugging and prints the contents of one voice packet in every 1024 voice packets.
debug ephone raw	Provides raw low-level protocol debugging display for all SCCP messages.
debug ephone register	Sets registration debugging for the Cisco IP phone.
debug ephone state	Sets state debugging for the Cisco IP phone.
debug ephone statistics	Sets statistics debugging for the Cisco IP phone.
show debugging	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 **l oopback-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:TONE OFF
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
```

The below table describes the significant fields shown in the display.

Table 2: debug ephone loopback Field Descriptions

Field	Description
Called Number	Original called number as presented to the incoming side of the loopback-dn.

Field	Description
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: • Ephone-dn tag and channel of the far-end call terminator (in this example, ephone-dn 14 is extension 1514). • other--Ephone-dn tag of the outgoing side of the loopback. • lb--Ephone-dn tag of the incoming side of the loopback. • far--Ephone-dn tag and channel of the far-end call originator, or -1 for a nonlocal number. • 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.
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.

Field	Description
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: • 16--Normal disconnect. • 17--User busy. • 19--No answer. • 28--Invalid number.

Related Commands

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

debug ephone message

To enable message tracing between ephones, use the **debug ephone message**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ephone message [detail]

no debug ephone message

Syntax Description

detail	(Optional) Displays signaling connection control protocol (SCCP) messages sent and received between ephones in the Cisco Unified CallManager Express (Cisco Unified CME) system.

Command Modes

Privileged EXEC

Command History

Cisco IOS Release	Modification
12.4(4)XC	This command was introduced.
12.4(9)T	This command was integrated into Cisco IOS Release 12.4(9)T.

Usage Guidelines

The **debug ephone message** command enables message tracing between ephones.

The debug ephone command debugs all ephones associated with a Cisco Unified CME router.

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

Examples

The following is sample output for the **debug ephone message** command for ephones:

```
Router# debug ephone message
EPHONE skinny message debugging is enabled
*Jul 17 12:12:54.883: Received message from phone 7, SkinnyMessageID = StationKe
epAliveMessageID
*Jul 17 12:12:54.883: Sending  message to   phone 7, SkinnyMessageID = StationKe
epAliveAckMessageID
```

The following command disables ephone message debugging:

```
Router# no debug ephone message
EPHONE skinny message debugging is disabled
```

Related Commands

Command	Description
debug ephone alarm	Sets SkinnyStation alarm messages debugging for the ephone.
debug ephone detail	Sets detail debugging for the ephone.
debug ephone error	Sets error debugging for the ephone.
debug ephone mwi	Sets MWI debugging for the ephone.
debug ephone pak	Provides voice packet level debugging and displays the contents of one voice packet in every 1024 voice packets.
debug ephone raw	Provides raw low-level protocol debugging display for all SCCP messages.
debug ephone register	Sets registration debugging for the ephone.
debug ephone state	Sets state debugging for the ephone.
debug ephone statistics	Sets statistics debugging for the ephone.
debug ephone video	Sets video debugging for the ephone.
show debugging	Displays information about the types of debugging that are enabled for your router.
show ephone	Displays information about ephones.

debug ephone mlpp

To display debugging information for Multilevel Precedence and Preemption (MLPP) calls to phones in a Cisco Unified CME system, use the **debug ephone mlpp**command in privileged EXEC mode. To disable debugging, use the **no** form of this command.

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

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

Syntax Description

mac-address *mac-address*	(Optional) Specifies the MAC address of a specific IP phone.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.4(22)YB	This command was introduced.
12.4(24)T	This command was integrated into Cisco IOS Release 12.4(24)T .

Usage Guidelines

Use this command to troubleshoot calls that use the MLPP service.

Examples

The following is sample output from the **debug ephone mlpp** command. This example shows output for the following call scenario:

- Ephone 1 is connected to ephone 3 (nonMLPP call).
- Ephone 4 makes an MLPP call to ephone 3. Preemption tone is played to both ephone 1 and 3.
- Ephone 3 is disconnected after the preemption tone timeout and precedence ringing.
- Ephone 3 answers the MLPP call and is connected to ephone 4.

```
Router# debug ephone mlpp

Sep  5 14:23:00.499: ephone-4[3/3][SEP001AE2BC3EE7]:indication=1
Sep  5 14:23:00.499: ephone-4[3/3][SEP001AE2BC3EE7]:max precedence=0
Sep  5 14:23:02.299: ephone-4[3/3][SEP001AE2BC3EE7]:mlpp_ephone_display_update callID=294
Sep  5 14:23:02.299: ephone-4[3/3][SEP001AE2BC3EE7]:indication=1
Sep  5 14:23:02.299: ephone-4[3/3][SEP001AE2BC3EE7]:mlpp precedence=4, domain=0
Sep  5 14:23:02.303: ephone-3[2/1][SEP001B54BA0D64]:preemption=1
Sep  5 14:23:02.303: ephone-3[2/1][SEP001B54BA0D64]:preemption=1
Sep  5 14:23:02.303: mlpp_ephone_find_call: preempt_htsp=1774234732,
prempt_htsp->mlpp_preemptor_cid=294
Sep  5 14:23:02.303: //294/A6B5C03A8141/VOIP-MLPP/voice_mlpp_get_preemptInfo:
   mlpp_ephone_find_call is successful
```

```
Sep  5 14:23:02.303: ephone-4[3/3][SEP001AE2BC3EE7]:indication=1
Sep  5 14:23:02.303: ephone-4[3/3][SEP001AE2BC3EE7]:mlpp precedence=4, domain=0
Sep  5 14:23:02.303: ephone-4[3/3][SEP001AE2BC3EE7]:indication=1
Sep  5 14:23:02.303: ephone-4[3/3][SEP001AE2BC3EE7]:mlpp precedence=4, domain=0
Sep  5 14:23:02.303: ephone-6[5/6][SEP0018187F49FD]:indication=1
Sep  5 14:23:02.303: ephone-6[5/6][SEP0018187F49FD]:mlpp precedence=4, domain=0
Sep  5 14:23:02.303: ephone-4[3/3][SEP001AE2BC3EE7]:indication=1
Sep  5 14:23:02.307: ephone-1[0/2][SEP0014A9818797]:indication=1
Sep  5 14:23:02.307: ephone-3[2/1][SEP001B54BA0D64]:indication=1
Sep  5 14:23:02.307: ephone-1[0/2][SEP0014A9818797]:indication=1DtPreemptionTone
Sep  5 14:23:02.307: ephone-3[2/1][SEP001B54BA0D64]:indication=1DtPreemptionTone
Sep  5 14:23:07.307: ephone-3[2/1][SEP001B54BA0D64]:indication=1
Sep  5 14:23:07.307: ephone-1[0/2][SEP0014A9818797]:indication=1
Sep  5 14:23:07.319: ephone-3[2/1][SEP001B54BA0D64]:indication=1
Sep  5 14:23:07.319: ephone-3[2/1][SEP001B54BA0D64]:indication=1
Sep  5 14:23:07.319: ephone-3[2/1][SEP001B54BA0D64]:mlpp precedence=4, domain=0
Sep  5 14:23:07.319: ephone-3[2/1][SEP001B54BA0D64]:indication=1
Sep  5 14:23:07.319: ephone-3[2/1][SEP001B54BA0D64]: MLPP Precedence Ring 6 instead
Sep  5 14:23:10.623: ephone-3[2/1][SEP001B54BA0D64]:indication=1
Sep  5 14:23:10.623: ephone-3[2/1][SEP001B54BA0D64]:indication=1
Sep  5 14:23:10.623: ephone-3[2/1][SEP001B54BA0D64]:mlpp precedence=4, domain=0
Sep  5 14:23:10.623: ephone-3[2/1][SEP001B54BA0D64]:indication=1
Sep  5 14:23:10.623: ephone-3[2/1][SEP001B54BA0D64]:mlpp precedence=4, domain=0
Sep  5 14:23:10.623: ephone-4[3/3][SEP001AE2BC3EE7]:indication=1
Sep  5 14:23:10.623: ephone-4[3/3][SEP001AE2BC3EE7]:mlpp precedence=4, domain=0
Sep  5 14:23:10.623: ephone-6[5/6][SEP0018187F49FD]:indication=1
Sep  5 14:23:10.623: ephone-6[5/6][SEP0018187F49FD]:mlpp precedence=4, domain=0
```

Related Commands

Command	Description
debug voice mlpp	Displays debugging information for MLPP service.
mlpp indication	Enables MLPP indication on an SCCP phone or analog FXS port.
mlpp max-precedence	Sets the maximum precedence (priority) level that a phone user can specify when making an MLPP call.
mlpp preemption	Enables preemption capability on an SCCP phone or analog FXS port.

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

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 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
```

The below table describes the significant fields shown in the display.

Table 3: 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
moh (telephony-service)	Generates an audio stream from a file for MOH in a Cisco CME system.
multicast moh	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.

Command Default

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

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

mac-address	(Optional) Defines the MAC address of the Cisco IP phone.
mac-address	(Optional) Specifies the MAC address of the Cisco IP phone.

Command Default

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_frf11:0 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_frf11:0 vlen:0 delta_t:160 vofr1:0 vofr2:0
```

Related Commands

Command	Description
debug ephone alarm	Sets SkinnyStation alarm messages debugging for the Cisco IP phone.
debug ephone detail	Sets detail debugging for the Cisco IP phone.
debug ephone error	Sets error debugging for the Cisco IP phone.
debug ephone keepalive	Sets keepalive debugging for the Cisco IP phone.
debug ephone loopback	Sets MWI debugging for the Cisco IP phone.

Command	Description
debug ephone raw	Provides raw low-level protocol debugging display for all SCCP messages.
debug ephone register	Sets registration debugging for the Cisco IP phone.
debug ephone state	Sets state debugging for the Cisco IP phone.
debug ephone statistics	Sets statistics debugging for the Cisco IP phone.
show debugging	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)
```

The following table describes the significant fields shown in the display.

Table 4: 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. **Note** This value is meaningful only if the call uses a G.711 codec.

Related Commands

Command	Description
debug ephone alarm	Displays alarm messages for IP phones.

Command	Description
debug ephone statistics	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

mac-address	(Optional) Defines the MAC address of the Cisco IP phone.
mac-address	(Optional) Specifies the MAC address of the Cisco IP phone.

Command Default

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
debug ephone alarm	Sets SkinnyStation alarm messages debugging for the Cisco IP phone.
debug ephone detail	Sets detail debugging for the Cisco IP phone.
debug ephone error	Sets error debugging for the Cisco IP phone.
debug ephone keepalive	Sets keepalive debugging for the Cisco IP phone.
debug ephone loopback	Sets MWI debugging for the Cisco IP phone.
debug ephone pak	Provides voice packet level debugging and prints the contents of one voice packet in every 1024 voice packets.
debug ephone register	Sets registration debugging for the Cisco IP phone.
debug ephone state	Sets state debugging for the Cisco IP phone.
debug ephone statistics	Sets statistics debugging for the Cisco IP phone.
show debugging	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

mac-address	(Optional) Defines the MAC address of the Cisco IP phone.
mac-address	(Optional) Specifies the MAC address of the Cisco IP phone.

Command Default

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

debug ephone sccp-state

To set debugging for the SCCP call state, use the **debug ephone sccp-state** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

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

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

Syntax Description

mac-address *mac-address*	(Optional) Specifies the MAC address of a phone.

Command Default

Debugging is not enabled for SCCP state.

Command Modes

Privileged EXEC

Command History

Cisco IOS Release	Modification
12.4(4)XC	This command was introduced.
12.4(9)T	This command was integrated into Cisco IOS Release 12.4(9)T.

Usage Guidelines

This command is used with Cisco Unified CallManager Express (Cisco Unified CME).

This command outputs only the debug messages that correspond to SCCP messages sent to IP phones to indicate the SCCP phone call state, such as RingIn, OffHook, Connected, and OnHook. These debug messages are also included in the output for the **debug ephone detail** command among other information.

Examples

The following example sets SCCP state debugging for one Cisco Unified CME phone with the MAC address of 678B.AEF9.DAB5.

```
Router# debug ephone sccp-state mac-address 678B.AEF9.DAB5
EPHONE SCCP state message debugging is enabled
  for ephones 000B.BEF9.DFB5
*Mar  8 06:38:45.863: %ISDN-6-CONNECT: Interface Serial2/0/0:22 is now connected to 4085254871
 unknown
*Mar  8 06:38:50.487: ephone-2[13]:SetCallState line 4 DN 60(60) chan 1 ref 100 TsRingIn
*Mar  8 06:38:52.399: ephone-2[13]:SetCallState line 4 DN 60(-1) chan 1 ref 100 TsOffHook
*Mar  8 06:38:52.399: ephone-2[13]:SetCallState line 4 DN 60(-1) chan 1 ref 100 TsConnected

*Mar  8 06:38:58.415: %ISDN-6-CONNECT: Interface Serial2/0/0:22 is now connected to 4085254871
 unknown
*Mar  8 06:38:59.963: ephone-2[13]:SetCallState line 4 DN 60(-1) chan 1 ref 100 TsOnHook
*Mar  8 06:38:59.975: %ISDN-6-DISCONNECT: Interface Serial2/0/0:22 disconnected from
4085254871 , call lasted 7 seconds
```

Related Commands

Command	Description
debug ephone detail	Sets detail debugging for one or all Cisco Unified IP phones.

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

mac-address	(Optional) Defines the MAC address of the Cisco IP phone.
mac-address	(Optional) Specifies the MAC address of the Cisco IP phone.

Command Default

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

Command	Description
show debugging	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

mac-address	(Optional) Defines the MAC address of the Cisco IP phone.
mac-address	(Optional) Specifies the MAC address of the Cisco IP phone.

Command Default

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

debug ephone video

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

debug ephone video

no debug ephone video

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging is disabled for ephone video.

Command Modes

Privileged EXEC

Command History

Cisco IOS Release	Modification
12.4(4)XC	This command was introduced.
12.4(9)T	This command was integrated into Cisco IOS Release 12.4(9)T.

Usage Guidelines

The **debug ephone video** command sets ephone video traces, which provide information about different video states for the call, including video capabilities selection, start, and stop.

The debug ephone command debugs all ephones that are registered to the Cisco Unified CallManager Express (Cisco Unified CME) system.

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

Examples

The following is sample output for the **debug ephone video** command for ephones:

```
Router# debug ephone video
*Mar 13 16:10:02.703: SkinnyVideoCodecMatch_Caps2Caps: match capability: tx_idxcap = 4,
tx_idxpref = 3,
*Mar 13 16:10:02.703:                rx_idxcap = 0, rx_idxpref = 0, videoBitRate = 7040
tx_mpi = 1
*Mar 13 16:10:04.711: ephone-19[1][SEPFFFA00000019]:checkToOpenMultiMedia: dn=19, chan=1
*Mar 13 16:10:04.711: ephone-19[1]:skinnyDP[19].s2s = 0
*Mar 13 16:10:04.711: ephone-19[1]:s2s is not set - hence not video capable
*Mar 13 16:10:04.719: ephone-19[1][SEPFFFA00000019]:SkinnyStartMultiMediaTransmission: chan
 1 dn 19
*Mar 13 16:10:04.723: ephone-19[1]:Accept OLC and open multimedia channel
*Mar 13 16:10:04.723: ephone-19[1][SEPFFFA00000019]:SkinnyOpenMultiMediaReceiveChannel: dn
 19 chan 1
*Mar 13 16:10:04.967: ephone-19[1][SEPFFFA00000019]:fStationOpenReceiveChannelAckMessage:
MEDIA_DN 19 MEDIA_CHAN 1
*Mar 13 16:10:04.967: ephone-19[1]:fStationOpenMultiMediaReceiveChannelAckMessage:
```

```
*Mar 13 16:10:04.967: ephone-19[1]:Other_dn == -1
sk3745-2#
*Mar 13 16:10:14.787: ephone-19[1]:SkinnyStopMedia: Stop Multimedia
*Mar 13 16:10:14.787: ephone-19[1][SEPFFFA00000019]:SkinnyCloseMultiMediaReceiveChannel:
passThruPartyID = 0, callReference = 23
*Mar 13 16:10:14.787: ephone-19[1]:SkinnyStopMultiMediaTransmission: line 1 chan 1 dn 19
```

Related Commands

Command	Description
debug ephone alarm	Sets SkinnyStation alarm messages debugging for the ephone.
debug ephone detail	Sets detail debugging for the ephone.
debug ephone error	Sets error debugging for the ephone.
debug ephone message	Sets message debugging for the ephone.
debug ephone mwi	Sets MWI debugging for the ephone.
debug ephone pak	Provides voice packet level debugging and displays the contents of one voice packet in every 1024 voice packets.
debug ephone raw	Provides raw low-level protocol debugging display for all SCCP messages.
debug ephone register	Sets registration debugging for the ephone.
debug ephone state	Sets state debugging for the ephone.
debug ephone statistics	Sets statistics debugging for the ephone.
show debugging	Displays information about the types of debugging that are enabled for your router.
show ephone	Displays information about registered ephones.

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*
```

The below table describes the significant fields shown in the display.

Table 5: 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
pattern direct	Configures the DTMF digit pattern forwarding necessary to activate the voice-mail system when a user presses the Messages button on a phone.
pattern ext-to-ext busy	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.
pattern ext-to-ext no-answer	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.
pattern trunk-to-ext busy	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.
pattern trunk-to-ext no-answer	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.
vm-integration	Enters voice-mail integration configuration mode and enables voice-mail integration with DTMF and analog voice-mail systems.
voicemail	Defines the telephone number that is speed-dialed when the Messages button on a Cisco IP phone is pressed.

debug ephone whisper-intercom

To display debugging messages for the Whisper Intercom feature, use the **debug ephone whisper-intercom** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ephone whisper-intercom

no debug ephone whisper-intercom

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging for Whisper Intercom is disabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.4(22)YB	This command was introduced.
12.4(24)T	This command was integrated into Cisco IOS Release 12.4(24

Usage Guidelines

This command displays debugging information about the Whisper Intercom feature configured on a directory number of a SCCP phone.

Examples

The following example displays output from the **debug ephone whisper-intercom** command:

```
Router# debug ephone whisper-intercom

ephone-1[0] Mac:1111.C1C1.0001 TCP socket:[8] activeLine:0 whisperLine:2 REGISTERED in SCCP
 ver 12/12 max_streams=3
mediaActive:0 whisper_mediaActive:0 startMedia:1 offhook:1 ringing:0 reset:0 reset_sent:0
paging 0 debug:0 caps:5
IP:10.6.2.185 9237 7970  keepalive 16 max_line 8
button 1: dn 1  number 2001 CH1   IDLE         CH2   IDLE
button 2: dn 161 number 6001  auto dial 6002 CH1   WHISPER
Preferred Codec: g711ulaw
Active Call on DN 161 chan 1 :6001 0.0.0.0 0 to 10.6.2.185 9280 via 10.6.2.185
G711Ulaw64k  160 bytes no vad
Tx Pkts 0 bytes 0 Rx Pkts 0 bytes 0 Lost 0
Jitter 0 Latency 0 callingDn -1 calledDn 162


ephone-2[1] Mac:1111.C1C1.0002 TCP socket:[7] activeLine:0 whisperLine:2 REGISTERED in SCCP
 ver 12/12 max_streams=3
mediaActive:0 whisper_mediaActive:1 startMedia:0 offhook:1 ringing:0 reset:0 reset_sent:0
paging 0 debug:0 caps:5
IP:10.6.2.185 9240 7970  keepalive 16 max_line 8
button 1: dn 2  number 2002 CH1   IDLE         CH2   IDLE
button 2: dn 162 number 6002  auto dial 6001 CH1   WHISPER
Preferred Codec: g711ulaw
```

```
Active Call on DN 162 chan 1 :6002 10.6.2.185 9280 to 10.6.2.254 2000 via 10.6.2.185
G711Ulaw64k  160 bytes no vad
Tx Pkts 0 bytes 0 Rx Pkts 0 bytes 0 Lost 0
Jitter 0 Latency 0 callingDn 161 calledDn -1
```

Related Commands

Command	Description
show ephone-dn whisper	Displays information about whisper intercom ephone-dns that have been created in Cisco Unified CME.
whisper-intercom	Enables the Whisper Intercom feature on a directory number.

debug epmpal

To enable debugging of the Y.1731 Ethernet performance monitoring functions, use the **debug epmpal** command in the privileged EXEC mode. To disable the debugging function, use the **no** form of this command.

debug epmpal {**all | api | rx | tx**}

no debug epmpal {**all | api | rx | tx**}

Syntax Description

all	Enables debugging of all the performance-monitoring events.
api	Enables debugging of the application program interface (API) events.
rx	Enables debugging of the receive events.
tx	Enables debugging of the transmit events.

Command Default

Debugging is disabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
15.4(1)T	This command was introduced into Cisco IOS Release 15.4(1)T.

Usage Guidelines

Use this command to troubleshoot the Y.1731 Ethernet performance-monitoring functions on the following routers:

- Cisco 3900 Series, Cisco 2900 Series, and Cisco 1900 Series Integrated Services Routers Generation 2
- Cisco 890 Series Integrated Services Routers

Examples

The following is a sample output of the **debug epmpal rx** command:

```
Device# debug epmpal rx

EPM platform pkt receive events debugging is on
```

The following is a sample output of the **clear log** command:

```
Device# clear log

Clear logging buffer [confirm]
```

The following is a sample output of the **show log** command:

```
Device# show log

Syslog logging: enabled (0 messages dropped, 7 messages rate-limited, 0 flushes, 0
overruns, xml disabled, filtering disabled)
No Active Message Discriminator.
No Inactive Message Discriminator.
Console logging: disabled
Monitor logging: level debugging, 0 messages logged, xml disabled,
filtering disabled
Buffer logging: level debugging, 100 messages logged, xml disabled,
filtering disabled
Exception Logging: size (8192 bytes)
Count and timestamp logging messages: disabled
Persistent logging: disabled
No active filter modules.
Trap logging: level informational, 90 message lines logged
Logging Source-Interface: VRF Name:
Log Buffer (1000000 bytes):
*Jul 15 16:56:35.383: Rcvd message of type 47, setting l2cos to 5, vlan_id to 1200.
*Jul 15 16:56:35.383: Timestamped incoming DMR packet with 3582896195:383921796.
*Jul 15 16:56:36.383: Rcvd message of type 47, setting l2cos to 5, vlan_id to 1200.
*Jul 15 16:56:36.383: Timestamped incoming DMR packet with 3582896196:383799732.
*Jul 15 16:56:37.379: Rcvd message of type 47, setting l2cos to 5, vlan_id to 1200.
*Jul 15 16:56:37.379: Timestamped incoming DMR packet with 3582896197:383677668.
```

Related Commands

Command	Description
clear log	Clears messages from the logging buffer.
show log	Displays the contents of the standard syslog buffer.

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

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

Command Default

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 **dot1x**, **filtermgr**, and **fltdrv** keywords were added.

Release	Modification
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.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

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

Command	Description
show debugging	Displays information about the types of debugging that are enabled.

debug ethernet cfm all

To enable all Ethernet connectivity fault management (CFM) debug messages, use the **debug ethernet cfm all** command in privileged EXEC mode. To disable all Ethernet CFM debug messages, use the **no** form of this command.

Cisco pre-Standard CFM Draft 8 (CFM D8)

ethernet cfm all [**domain** *domain-name*| **level** *level-id*] [**evc** *evc-name*| **vlan** *vlan-id*]

no debug ethernet cfm all [**domain** *domain-name*| **level** *level-id*] [**evc** *evc-name*| **vlan** *vlan-id*]

CFM IEEE 802.1ag Standard (CFM IEEE)

debug ethernet cfm all [**domain** *domain-name*] [**port**| **vlan** *vlan-id*]

no debug ethernet cfm all [**domain** *domain-name*] [**port**| **vlan** *vlan-id*]

Syntax Description

domain	(Optional) Indicates that a domain is specified.
domain-name	(Optional) String of a maximum of 154 characters.
level	(Optional) Indicates that a maintenance level is specified.
level-id	(Optional) Integer in the range of 0 to 7 that specifies the maintenance level.
evc	(Optional) Identifies the Ethernet virtual connection (EVC). An EVC is an association of two or more user network interfaces (UNIs).
evc-name	(Optional) String that identifies the EVC name.
port	(Optional) Indicates a DOWN service direction with no VLAN association (untagged).
vlan	(Optional) Indicates that a VLAN is specified.
vlan-id	(Optional) Integer in the range of 1 to 4094 that identifies the affected VLAN.

Command Default

All debug commands are enabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(33)SRA	This command was introduced.
12.4(11)T	This command was integrated into Cisco IOS Release 12.4(11)T.
12.2(33)SRD	The **evc** keyword and *evc-name* argument were introduced on the Cisco 7600 Series Route Switch Processor 720 (RSP 720) and the Cisco 7600 Series Supervisor Engine 720.
12.2(33)SXI2	This command was integrated into Cisco IOS Release 12.2(33)SXI2. • The **level** and **evc** keywords and the *level-id* and *evc-name* arguments are not supported in Cisco IOS Release 12.2(33)SXI2.
12.2(33)SRE	This command was integrated into Cisco IOS Release 12.2(33)SRE.
15.1(1)T	This command was integrated into Cisco IOS Release 15.1(1)T.
12.2(50)SY	This command was integrated into Cisco IOS Release 12.2(50)SY.
Cisco IOS XE Release 3.5S	This command was integrated into Cisco IOS XE Release 3.5S.
15.3(1)S	This command was integrated into Cisco IOS Release 15.3(1)S.

Usage Guidelines

In CFM IEEE, if a domain name has more than 43 characters, a warning message is displayed notifying that the maintenance domain ID (MDID) will be truncated to 43 characters in continuity check messages (CCMs) if “id <fmt> <MDID>” is not configured.

This command allows you to conditionally enable debug messages. The messages depend on the version of CFM you are running. When CFM IEEE is running, you are prompted to respond “yes” or “no.” The messages relate to the following:

- Maintenance domain
- Maintenance level
- Maintenance domain plus VLAN or EVC
- Maintenance level plus VLAN or EVC

Additionally, you can filter debug messages by the following:

- Maintenance domain
- Maintenance level
- VLAN or EVC
- Combination of maintenance domain and VLAN or EVC
- Combination of maintenance level and VLAN or EVC

The output from the **debug ethernet cfm all** command is a log of activity that shows all Ethernet CFM-related debug messages. Use this command to troubleshoot Ethernet CFM in your network.

Examples

The following example shows output of the **debug ethernet cfm all** command:

```
Router# debug ethernet cfm all domain Domain_L5 vlan 9
This may impact network performance. Continue? (yes/[no]): yes
Ethernet CFM level 5 domain Domain_L5 vlan 9 packet debugging is on
Ethernet CFM level 5 domain Domain_L5 vlan 9 event debugging is on
Router#
Jun 17 21:41:49.839: CFM-PKT: Received a CC packet with MPID 401, level 5, vlan 9 from
interface Ethernet0/0.9
Jun 17 21:41:49.839: CFM-EVT: Found remote mep for domain Domain_L5, level 5 vlan 9, mpid
401 mac aabb.cc03.bb99
Jun 17 21:41:49.839: CFM-EVT: Updated rmep in MIP CCDB, domain Domain_L5 level 5,
                                                        vlan 9 mac aabb.cc03.bb99 intf
Ethernet0/0.9
Router#
Jun 17 21:41:56.007: CFM-PKT: Sending Up direction MEP 220 CC message, level 5, vlan 9
Router#
Jun 17 21:42:00.539: CFM-PKT: Received a CC packet with MPID 401, level 5, vlan 9 from
interface Ethernet0/0.9
Jun 17 21:42:00.539: CFM-EVT: Found remote mep for domain Domain_L5, level 5 vlan 9, mpid
401 mac aabb.cc03.bb99
Jun 17 21:42:00.539: CFM-EVT: Updated rmep in MIP CCDB, domain Domain_L5 level 5,
        vlan 9 mac aabb.cc03.bb99 intf Ethernet0/0.9
```

Related Commands

Command	Description
debug ethernet cfm diagnostic	Enables low-level diagnostic debugging of Ethernet CFM general or packet-related events.
debug ethernet cfm error	Enables debugging of Ethernet CFM errors.
debug ethernet cfm events	Enables debugging of Ethernet CFM events.
debug ethernet cfm packets	Enables debugging of Ethernet CFM message packets.

debug ethernet cfm diagnostic

To enable low-level diagnostic debugging of Ethernet connectivity fault management (CFM) general events or packet-related events, use the **debug ethernet cfm diagnostic** command in privileged EXEC mode. To disable the debugging, use the **no** form of this command.

Cisco pre-Standard CFM Draft 1 (CFM D1)

debug ethernet cfm diagnostic [**events**| **packets**]

no debug ethernet cfm diagnostic [**events**| **packets**]

CFM IEEE 802.1ag Standard (CFM IEEE)

debug ethernet cfm diagnostic [**ais**| **events**| **lck**| **mip-autocreate**| **packets** [**ais**| **cc**| **lb**| **lck**| **lt**]]

no debug ethernet cfm diagnostic [**ais**| **events**| **lck**| **mip-autocreate**| **packets** [**ais**| **cc**| **lb**| **lck**| **lt**]]

Syntax Description

ais	(Optional) Triggers debugging of Alarm Indication Signal (AIS) activities.
events	(Optional) Triggers debugging of events.
lck	(Optional) Triggers debugging of lck events. • When used with the **packets** keyword, triggers debugging of lck packets.
mip-autocreate	(Optional) Triggers debugging of MIP autocreate activities.
packets	(Optional) Triggers debugging of packets.
ais	(Optional) Triggers debugging of alarm indication signal (AIS) packets.
cc	(Optional) Triggers debugging of continuity check (cc) packets.
lb	(Optional) Triggers debugging of loopback (lb) packets.
lt	(Optional) Triggers debugging of linktrace (lt) packets.

Command Default

Diagnostic debugging for both events and packets is enabled.

Command Modes Privileged EXEC (#)

Command History

Release	Modification
12.2(33)SRA	This command was introduced.
12.4(11)T	This command was integrated into Cisco IOS Release 12.4(11)T.
12.2(33)SXI2	This command was integrated into Cisco IOS Release 12.2(33)SXI2.
12.2(33)SRE	This command was integrated into Cisco IOS Release 12.2(33)SRE.
15.1(1)T	This command was integrated into Cisco IOS Release 15.1(1)T.
Cisco IOS XE Release 3.5S	This command was integrated into Cisco IOS XE Release 3.5S.
15.3(1)S	This command was integrated into Cisco IOS Release 15.3(1)S.

Usage Guidelines The output from this command is a log of activity. Use this command to troubleshoot Ethernet CFM in your network.

Examples The following example shows output of the **debug ethernet cfm diagnostic** command with no options specified:

```
Router# debug ethernet cfm diagnostic
Ethernet CFM diagnostic events debugging is on
Ethernet CFM diagnostic packets debugging is on
Ethernet CFM diagnostic mip_autocreate debugging is on
Ethernet CFM diagnostic ais debugging is on
Ethernet CFM diagnostic lck debugging is on
Ethernet CFM diagnostic packet cc debugging is on
Ethernet CFM diagnostic packet lb debugging is on
Ethernet CFM diagnostic packet lt debugging is on
Ethernet CFM diagnostic packet filter debugging is on
Ethernet CFM diagnostic packet ais debugging is on
Ethernet CFM diagnostic packet lck debugging is on
Router#
Router#
*Jun 17 21:48:56.803: CFM-PKT: Sending Up direction MEP 401 CC message, level 5, vlan 9
*Jun 17 21:48:56.803: CFMPAL-PKT: pak (CC) sent to interface Ethernet0/0.1 (linktype=1AG)
Router#
*Jun 17 21:49:00.535: CFMPAL-PKT: Received a CFM packet (CC) from port Ethernet0/0.11
(linktype=1AG)
*Jun 17 21:49:00.535: cc_filter, service mcl = 7 for vlan = 11
*Jun 17 21:49:00.535: ecfm_pal_cc_filter:computed levels - fl = -1, fm = -1, hm = -1, level
 = 7
*Jun 17 21:49:00.535: ecfm_pal_cc_filter:hi_ofm=-1, lo_ofm=-1, hi_ifm=-1, lo_ifm=-1, mcl=7,
 mip_level=-1
*Jun 17 21:49:00.535: pak_level EQUAL MCL
*Jun 17 21:49:00.535: L > fl, Punt and Forward
*Jun 17 21:49:00.535: CFMPAL-EVT: packet not sent out on Ethernet0/1.11, hmep = 7
*Jun 17 21:49:00.535: CFM-PKT: Received a CC packet from interface Ethernet0/0.11
*Jun 17 21:49:00.535: CFM-PKT: cfm packet dump - 105 bytes, interface Ethernet0/0.11, vlan
 11
```

```
*Jun 17 21:49:00.535: CFM-PKT: ethernet CFM (1AG) message dump,
          dest: 0180.c200.0037
          src: aabb.cc03.b999
          Version: 0
          Maintenance Level: 7
          MsgType: CC(1)
          Flags: 0x5
          First TLV Offset: 70
*Jun 17 21:49:00.535: 01 80 C2 00 00 37 AA BB CC 03 B9 99 89 02 E0 01 05 46 26 FB AC E5 00
 65 04
*Jun 17 21:49:00.535: 09 44 6F 6D 61 69 6E 5F 4C 37 02 0B 63 75 73 74 5F 37 30 30 5F 6C 37
 00 00
*Jun 17 21:49:00.535: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 00 00
*Jun 17 21:49:00.535: 00 00 00 00 00 00 00 00 00 2C 20 56 65 02 00 01 02 04 00 01 01 1F 00
 05 00
*Jun 17 21:49:00.535: 00 0C 01 02 00
*Jun 17 21:49:00.535: CFM-PKT: Received a CC packet with MPID 101, level 7, vlan 11 from
interface Ethernet0/0.11
*Jun 17 21:49:00.535: CFMPAL-I-PKT: pak (CC) sent to interface Ethernet0/2.1 (linktype=1AG)
Router#
*Jun 17 21:49:02.675: CFM-PKT: Sending Up direction MEP 301 CC message, level 7, vlan 11
*Jun 17 21:49:02.675: CFMPAL-PKT: pak (CC) sent to interface Ethernet0/0.11 (linktype=1AG)
*Jun 17 21:49:02.675: CFMPAL-PKT: pak (CC) sent to interface Ethernet0/2.1 (linktype=1AG)
*Jun 17 21:49:02.943: CFMPAL-PKT: Received a CFM packet (CC) from port Ethernet0/0.1
(linktype=1AG)
*Jun 17 21:49:02.947: cc_filter, service mcl = 5 for vlan = 9
*Jun 17 21:49:02.947: ecfm_pal_cc_filter:computed levels - fl = -1, fm = -1, hm = -1, level
 = 5
*Jun 17 21:49:02.947: ecfm_pal_cc_filter:hi_ofm=-1, lo_ofm=-1, hi_ifm=-1, lo_ifm=-1, mcl=5,
 mip_level=-1
*Jun 17 21:49:02.947: pak_level EQUAL MCL
*Jun 17 21:49:02.947: L > fl, Punt and Forward
*Jun 17 21:49:02.947: CFMPAL-EVT: packet not sent out on Ethernet0/1.1, hmep = 5
*Jun 17 21:49:02.947: CFM-PKT: Received a CC packet from interface Ethernet0/0.1
*Jun 17 21:49:02.947: CFM-PKT: cfm packet dump - 105 bytes, interface Ethernet0/0.1, vlan
9
*Jun 17 21:49:02.947: CFM-PKT: ethernet CFM (1AG) message dump,
          dest: 0180.c200.0035
          src: aabb.cc03.b999
          Version: 0
          Maintenance Level: 5
          MsgType: CC(1)
          Flags: 0x5
          First TLV Offset: 70
*Jun 17 21:49:02.947: 01 80 C2 00 00 35 AA BB CC 03 B9 99 89 02 A0 01 05 46 06 5B 0C 0E 00
 DC 04
Router#
*Jun 17 21:49:02.947: 09 44 6F 6D 61 69 6E 5F 4C 35 02 0B 63 75 73 74 5F 35 30 30 5F 6C 35
 00 00
*Jun 17 21:49:02.947: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 00 00
*Jun 17 21:49:02.947: 00 00 00 00 00 00 00 00 00 2C 20 56 65 02 00 01 02 04 00 01 01 1F 00
 05 00
*Jun 17 21:49:02.947: 00 0C 01 02 00
*Jun 17 21:49:02.947: CFM-PKT: Received a CC packet with MPID 220, level 5, vlan 9 from
interface Ethernet0/0.1
```

Related Commands

Command	Description
debug ethernet cfm all	Enables all Ethernet CFM debug messages.
debug ethernet cfm error	Enables debugging of Ethernet CFM errors.
debug ethernet cfm events	Enables debugging of Ethernet CFM events.
debug ethernet cfm packets	Enables debugging of Ethernet CFM message packets.

debug ethernet cfm error

To enable debugging of Ethernet connectivity fault management (CFM) errors, use the **debug ethernet cfm error** command in privileged EXEC mode. To disable the debugging, use the **no** form of this command.

debug ethernet cfm error

no debug ethernet cfm error

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging is disabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(33)SXI2	This command was introduced. This command replaces the **debug ethernet cfm errors** command.
12.2(33)SRE	This command was integrated into Cisco IOS Release 12.2(33)SRE.
15.1(1)T	This command was integrated into Cisco IOS Release 15.1(1)T.

Usage Guidelines

There is no filtering capability for this command. The output from this command is a log of activity.

Use this command to troubleshoot Ethernet CFM in your network.

Examples

The following example shows output of the **debug ethernet cfm error** command:

```
Device# debug ethernet cfm error

Ethernet CFM error debugging is on
Device#
Device#
Device#! turn off cc messages on the other side for vlan 9
Device#
*Jun 17 21:53:44.955: CFM-ERR: Lifetime timer fired for level 5, vlan 9 mac aabb.cc03.b999,
 mpid 220
Device#
Device# show ethernet cfm error

-------------------------------------------------------------------------------
MPID Domain Id                                Mac Address     Type   Id  Lvl
     MAName                                   Reason                 Age
-------------------------------------------------------------------------------
220  Domain_L5                                aabb.cc03.b999  Vlan   9    5
     cust_500_l5                              Lifetime Timer Expired 17s
```

Related Commands

Command	Description
debug ethernet cfm all	Enables all Ethernet CFM debug messages.
debug ethernet cfm diagnostic	Enables low-level diagnostic debugging of Ethernet CFM general events or packet-related events.
debug ethernet cfm events	Enables debugging of Ethernet CFM events.
debug ethernet cfm packets	Enables debugging of Ethernet CFM message packets.

debug ethernet cfm errors

Note Effective with Cisco IOS Release 12.2(33)SXI2, the **debug ethernet cfm errors** command is replaced by the **debug ethernet cfm error** command. See the **debug ethernet cfm error** command for more information.

To enable debugging of Ethernet connectivity fault management (CFM) errors, use the **debug ethernet cfm errors**command in privileged EXEC mode. To disable the debugging, use the **no** form of this command.

debug ethernet cfm errors

no debug ethernet cfm errors

Syntax Description

This command has no keywords or arguments.

Command Default

Debugging is disabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(33)SRA	This command was introduced.
12.4(11)T	This command was integrated into Cisco IOS Release 12.4(11)T.
12.2(33)SXI2	This command was replaced by the **debug ethernet cfm error** command.
Cisco IOS XE Release 3.5S	This command was integrated into Cisco IOS XE Release 3.5S.

Usage Guidelines

The output from this command is a log of activity. Use this command to troubleshoot Ethernet CFM in your network.

Examples

The following example shows output of the **debug ethernet cfm errors** command:

```
Router# debug ethernet cfm errors
10:46:26: CFM-ERR: MPID matched with a local MEP!, level 5, svlan 2, mpid 50
10:46:26: CFM-ERR: Received duplicate mpid 50 due to configuration error for level 5, svlan
 2
10:48:55: CFM-ERR: Lifetime timer fired for level 4, svlan 2 mac aabb.cc00.0501, mpid 43
10:48:56: CFM-ERR: Lifetime timer fired for level 6, svlan 2 mac aabb.cc00.0300, mpid 21
10:48:57: CFM-ERR: Lifetime timer fired for level 5, svlan 5 mac aabb.cc00.0602, mpid 60
```

Related Commands

Command	Description
debug ethernet cfm all	Enables all Ethernet CFM debug messages.
debug ethernet cfm diagnostic	Enables low-level diagnostic debugging of Ethernet CFM general events or packet-related events.
debug ethernet cfm events	Enables debugging of Ethernet CFM events.
debug ethernet cfm packets	Enables debugging of Ethernet CFM message packets.

debug ethernet cfm ha

To enable debugging of Ethernet connectivity fault management (CFM) high availability (HA) features, use the **debug ethernet cfm ha** command in privileged EXEC mode. To disable debugging, use the **no** form of this command.

debug ethernet cfm ha

no debug ethernet cfm ha

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging is disabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(33)SRD	This command was introduced.
12.2(33)SXI2	This command was integrated into Cisco IOS Release 12.2(33)SXI2.
12.2(33)SRE	This command was integrated into Cisco IOS Release 12.2(33)SRE.

Examples

The following example shows sample output of the **debug ethernet cfm ha** command:

```
Device# debug ethernet cfm ha

00:18:12: CFM-HA: RF progression Callbk CID 207, Seq 169, Event RF_PROG_STANDBY_FILESYS,
Op 0, State ACTIVE, Peer STANDBY COLD-FILESYS
00:04:30: %SYS-SPSTBY-6-BOOTTIME: Time taken to reboot after reload =  391 seconds
00:18:22: CFM-HA: RF progression Callbk CID 207, Seq 169, Event RF_PROG_STANDBY_BULK, Op
0, State ACTIVE, Peer STANDBY COLD-BULK
00:18:22: CFM-HA: All Remote Mep Action 1
00:18:22: CFM-HA: MIP CCDB Bulk Sync Invoked
00:18:22: CFM-HA: Get buffer size 316 msg 4
00:18:22: CFM-HA: Get buffer size 316 msg 4
00:18:22: CFM-HA: Get buffer size 8 msg 1
00:18:22: CFM-HA: Event to Sync Buffer: Add MIP CCDB : vlan 100, level 5, mpid 100, version
 2, lifetime 210000, addr 0014.69b6.200e, id_fmt 4 ma_fmt 2
00:18:22: CFM-HA: Get buffer size 316 msg 4
00:18:22: CFM-HA: Event to Sync Buffer: Add MIP CCDB : vlan 200, level 6, mpid 1998, version
 2, lifetime 35000, addr 0014.f15c.a403, id_fmt 4 ma_fmt 2
00:18:22: CFM-HA: Get buffer size 316 msg 4
00:18:22: CFM-HA: Sending 2 records in Bulk
00:18:22: CFM-HA: All Remote Mep Action 1
00:18:22: CFM-HA:MEP CCDB Bulk Sync Invoked
00:18:22: CFM-HA: Get buffer size 284 msg 2
00:18:22: CFM-HA: Get buffer size 284 msg 2
00:18:22: CFM-HA: Get buffer size 8 msg 1
00:18:22: CFM-HA: Event to Sync Buffer: Add MEP CCDB: vlan 100, level 5, mpid 100, port_state
 2, archive FALSE , intf_state 1, ccheck_rmep_ok FALSE, addr 0014.69b6.200e, name
```

```
00:18:22: CFM-HA: Get buffer size 284 msg 2
00:18:22: CFM-HA: Event to Sync Buffer: Add MEP CCDB: vlan 200, level 6, mpid 1998, port_state
 2, archive FALSE , intf_state 1, ccheck_rmep_ok FALSE, addr 0014.f15c.a403, name
00:18:22: CFM-HA: Get buffer size 284 msg 2
00:18:22: CFM-HA: Sending 2 records in Bulk
```

Related Commands

Command	Description
debug ethernet cfm all	Enables all Ethernet CFM debug messages.
debug ethernet cfm diagnostic	Enables low-level diagnostic debugging of Ethernet CFM general or packet-related events.
debug ethernet cfm error	Enables debugging of Ethernet CFM errors.
debug ethernet cfm events	Enables debugging of Ethernet CFM events.
debug ethernet cfm packets	Enables debugging of Ethernet CFM message packets.

debug ethernet cfm packets

To enable debugging of Ethernet connectivity fault management (CFM) message packets, use the **debug ethernet cfm packets** command in privileged EXEC mode. To disable the debugging, use the **no** form of this command.

debug ethernet cfm packets [**domain** *domain-name*| **level** *level-id*] [**evc** *evc-name*| **vlan** *vlan-id*]

no debug ethernet cfm packets [**domain** *domain-name*| **level** *level-id*] [**evc** *evc-name*| **vlan** *vlan-id*]

debug ethernet cfm packets [**domain** *domain-name*] [**port**| **vlan** *vlan-id*]

no debug ethernet cfm packets [**domain** *domain-name*] [**port**| **vlan** *vlan-id*]

Syntax Description

domain	(Optional) Indicates that a domain is specified.
domain-name	(Optional) String of a maximum of 154 characters.
level	(Optional) Indicates that a maintenance level is specified.
level-id	(Optional) Integer in the range of 0 to 7 that specifies the maintenance level.
evc	(Optional) Identifies the Ethernet virtual connection (EVC). An EVC is an association of two or more user network interfaces (UNIs).
evc-name	(Optional) String that identifies the EVC name.
port	(Optional) Indicates a DOWN service direction with no VLAN association (untagged).
vlan	(Optional) Indicates that a VLAN is specified.
vlan-id	(Optional) Integer in the range of 1 to 4094.

Command Default

Debugging is enabled for all domains and VLANs.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(33)SRA	This command was introduced.

Release	Modification
12.4(11)T	This command was integrated into Cisco IOS Release 12.4(11)T.
12.2(33)SRD	The **evc** keyword and *evc-name* argument were introduced on the Cisco 7600 Series Route Switch Processor 720 (RSP 720) and the Cisco 7600 Series Supervisor Engine 720.
12.2(33)SXI2	This command was integrated into Cisco IOS Release 12.2(33)SXI2. • The **level** and **evc** keywords and the *level-id* and *evc-name* arguments are not supported in Cisco IOS Release 12.2(33)SXI2.
12.2(33)SRE	This command was integrated into Cisco IOS Release 12.2(33)SRE.
15.1(1)T	This command was integrated into Cisco IOS Release 15.1(1)T.
12.2(50)SY	This command was integrated into Cisco IOS Release 12.2(50)SY.
Cisco IOS XE Release 3.5S	This command was integrated into Cisco IOS XE Release 3.5S.
15.3(1)S	This command was integrated into Cisco IOS Release 15.3(1)S

Usage Guidelines

In CFM IEEE, if a domain name has more than 43 characters, a warning message is displayed notifying that the maintenance domain ID (MDID) will be truncated to 43 characters in continuity check messages (CCMs) if “id <fmt> <MDID>” is not configured.

This command allows you to filter debug messages. The messages depend on the version of CFM you are running. When CFM IEEE is running, you are prompted to respond “yes” or “no.” The messages relate to the following:

- Maintenance domain
- Maintenance level
- VLAN or EVC
- Combination of maintenance domain and VLAN or EVC
- Combination of maintenance level and VLAN or EVC

The output from this command is a log of activity. Use this command to troubleshoot Ethernet CFM in your network.

Examples

The following is sample output of the **debug ethernet cfm packets** command:

```
Router# debug ethernet cfm packets
Ethernet CFM packet debugging is on for all
Router#
*Jun 17 22:03:38.059: CFM-PKT: Sending Up direction MEP 301 CC message, level 7, vlan 11
Router#
*Jun 17 22:03:42.879: CFM-PKT: Sending Up direction MEP 401 CC message, level 5, vlan 9
Router#
```

```
*Jun 17 22:03:46.431: CFM-PKT: Received a CC packet with MPID 220, level 5, vlan 9 from
interface Ethernet0/0.1
*Jun 17 22:03:46.703: CFM-PKT: Received a CC packet with MPID 101, level 7, vlan 11 from
interface Ethernet0/0.11
Router#
*Jun 17 22:03:48.783: CFM-PKT: Sending Up direction MEP 301 CC message, level 7, vlan 11
Router#
*Jun 17 22:03:53.571: CFM-PKT: Sending Up direction MEP 401 CC message, level 5, vlan 9
Router#
*Jun 17 22:03:57.083: CFM-PKT: Received a CC packet with MPID 220, level 5, vlan 9 from
interface Ethernet0/0.1
*Jun 17 22:03:57.355: CFM-PKT: Received a CC packet with MPID 101, level 7, vlan 11 from
interface Ethernet0/0.11
Router#
```

Related Commands

Command	Description
debug ethernet cfm all	Enables all Ethernet CFM debug messages.
debug ethernet cfm diagnostic	Enables low-level diagnostic debugging of Ethernet CFM general events or packet-related events.
debug ethernet cfm error	Enables debugging of Ethernet CFM errors.
debug ethernet cfm events	Enables debugging of Ethernet CFM events.

debug ethernet cfm pm

To enable debug messages for Ethernet connectivity fault management (CFM) performance monitoring, use the **debug ethernet cfm pm**command in privileged EXEC mode. To disable Ethernet CFM performance monitoring debug messages, use the **no** form of this command.

debug ethernet cfm pm {**diagnostic**| {**error**| **events**| **ipc**| **packets**} [**session** *session-id*]}

no debug ethernet cfm pm {**diagnostic**| {**error**| **events**| **ipc**| **packets**} [**session** *session-id*]}

Syntax Description

diagnostic	Specifies debugging for performance monitoring diagnostic information.
error	Specifies debugging for performance monitoring error information.
events	Specifies debugging for performance monitoring event information.
ipc	Specifies debugging for performance monitoring Internet protocol communications (IPC).
packets	Specifies debugging for performance monitoring packet information.
session	(Optional) Indicates a specific session.
session-id	(Optional) Integer that identifies a session. Range is 0 to 9999999.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
15.1(2)S	This command was introduced.
Cisco IOS XE Release 3.5S	This command was integrated into Cisco IOS XE Release 3.5S.

Usage Guidelines

Per session debugging is recommended to reduce the number of debugs and to manage console output.

Use the **debug ethernet cfm pm** command with the **diagnostic** keyword to perform debugging while a session is being created.

Examples

The following example shows how to initiate debug messages for CFM performance monitoring events in session 25:

```
Router# debug ethernet cfm pm events session 25
```

debug ethernet event microwave

Enables debugging for Ethernet microwave events, use the **debug ethernet event microwave** command in privileged EXEC mode. To disable debugging of these events, use the **no** form of this command.

debug ethernet event microwave [all] [errors]

no debug ethernet event microwave

Syntax Description

all	(Optional) Displays debugging output for all Ethernet microwave events.
errors	(Optional) Displays debugging output associated with bandwidth errors.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS XE Release 3.8S	This command was introduced.

Examples

The following is sample output from the **debug ethernet event microwave** command. In this example, debugging for Ethernet microwave events has been enabled:

```
Device# debug ethernet event microwave
Ethernet Microwave Event Error debugging is on
```

debug ethernet l2ctrl

To enable debugging messages for Ethernet Layer 2 Control (L2ctrl), use the **debug ethernet l2ctrl** command in privileged EXEC mode. To disable debugging messages for Ethernet L2CTRL, use the **no** form of this command.

debug ethernet l2ctrl {**all**| **errors**| **events**}

no debug ethernet l2ctrl {**all**| **errors**| **events**}

Syntax Description

all	Displays all Ethernet L2CTRL debugging messages.
errors	Displays Ethernet L2CTRL error information
events	Displays Ethernet L2CTRL event information.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(33)SRD	This command was introduced.

Examples

The following is sample output from the debug ethernet l2ctrl events command:

When bridge-domain of 30 is defined on a service instance 3 at gigabitethernet interface 1/0/0 the output is as follows:

```
Router# debug ethernet l2ctrl events
17:17:03.174: EI/L2CTRL/ADD/EV: Gi1/0/0 (if_num 10) efp 3 (0x4944B5E8) vlan 30 state Up
```

When bridge-domain of 30 is not configured on service instance 3 at gigabitethernet interface 1/0/0 the output is as follows:

```
Router# debug ethernet l2ctrl events
17:16:30.546: EI/L2CTRL/DELETE/EV: Gi1/0/0 efp 3 vlan 30
```

The following is sample output from debug ethernet l2ctrl errors command:

```
Router# debug ethernet l2ctrl errors
17:16:30.546: DELETE/ERR: no vport found for Gi1/0/0 efp 3
```

Related Commands

Command	Description
debug l2ctrl	Enables debugging for L2CTRL.

debug ethernet lmi

To enable debugging of Ethernet Local Management Interface (LMI) messages on all interfaces or on a specified interface, use the **debug ethernet lmi** command in privileged EXEC mode. To disable debugging, use the **no** form of this command.

debug ethernet lmi {**all**| **errors**| **events**| **ha**| **packets**} [**interface** *type number*]

no debug ethernet lmi {**all**| **errors**| **events**| **ha**| **packets**} [**interface** *type number*]

Syntax Description

all	When you use the **all** keyword, keep in mind the number of interfaces that support Ethernet LMI. Some messages may be lost if many interfaces are supported.
errors	Use of the **errors** keyword enables debugging of Ethernet LMI errors such as invalid messages; for example, unexpected information element (IE) and mandatory IE missing.
events	Use of the **events** keyword enables debugging of Ethernet LMI events such as status changes, timeouts, and messages received.
ha	Use of the **ha** keyword enables debugging of Ethernet LMI high availability messages.
packets	Use of the **packets** keyword enables debugging of decoded Ethernet LMI packets.
interface	(Optional) Specifies an interface to use to filter debug messages.
type	(Optional) String that identifies the type of interface. Valid options are the following: • **ethernet—**Ethernet IEEE 802.3 interface. • **fastethernet —**Fast Ethernet IEEE 802.3 interface. • **gigabitethernet —**Gigabit Ethernet IEEE 802.3z interface.
number	(Optional) Integer that identifies the interface.

Command Default

Debugging is disabled.

Command Modes

Privileged EXEC(#)

Command History

Release	Modification
12.4(9)T	This command was introduced.
12.2(33)SRB	Support for this command on the Cisco 7600 router was integrated into Cisco IOS Release 12.2(33)SRB.
12.2(33)SRD	This command was modified. The **ha** keyword was added.
15.3(1)S	This command was integrated into Cisco IOS Release 15.3(1)S.
Cisco IOS XE Release 3.8S	This command was integrated into Cisco IOS XE Release 3.8S.

Usage Guidelines

The output from this command is a log of activity. Use this command to troubleshoot Ethernet LMI in your network.

Examples

The following example output from the **debug ethernet lmi all** command shows event and packet messages:

```
Device# debug ethernet lmi all

Ethernet LMI errors debugging is on
Ethernet LMI ha debugging is on
Ethernet LMI packets debugging is on
Ethernet LMI events debugging is on
Ethernet LMI packets hex debugging is on
00:29:32: ELMI Et0/0 EVENT: ce_event: State 0x0, Event 0x4
00:29:32: ELMI Et0/0 EVENT: Old State=0x0, Event=0x4, New State=0x2
00:29:32: ELMI Et0/0 EVENT: Updated Stat Type: ETHER_LMI_ST_LMSG_SENT
00:29:32: ELMI Et0/0 PKT HEX: TX->:0x01750101010202B4B3030500000000400000000000
00:29:32: ELMI Et0/0 PACKET: Outgoing
                Protocol Version     : 0x1
                Message              : STATUS ENQ (0x75)
                Report Type          : Check
                Sequence Number      : Snd(0xB4), Rcv(0xB3)
                Data Instance        : Value(0x4)
00:29:32: ELMI Et0/0 PKT HEX: RX<-:0x017D0101010202B4B4030500000000400000000000
00:29:32: ELMI Et0/0 PACKET: Incoming
                Protocol Version     : 0x1
                Message              : STATUS (0x7D)
                Report Type          : Check
                Sequence Number      : Snd(0xB4), Rcv(0xB4)
                Data Instance        : Value(0x4)
00:29:32: ELMI Et0/0 EVENT: ce_event: State 0x2, Event 0x1
00:29:32: ELMI Et0/0 EVENT: Update seq: current send 0xB4 rcv 0xB3
00:29:32: ELMI Et0/0 EVENT: Updated Stat Type: ETHER_LMI_ST_LMSG_RCVD
00:29:32: ELMI Et0/0 EVENT: Old State=0x2, Event=0x1, New State=0x0
00:06:30: ELMI HA: cpf_status_callback status 2
00:07:37: ELMI HA: RF progression Callbk CID 202, Seq 142, Event RF_PROG_STANDBY_CONFIG,
Op 0, State ACTIVE, Peer STANDBY COLD-CONFIG
00:07:37: ELMI HA: ISSU: Force negotiation version to V1
00:07:51: ELMI HA: RF progression Callbk CID 202, Seq 142, Event RF_PROG_STANDBY_FILESYS,
Op 0, State ACTIVE, Peer STANDBY COLD-FILESYS
```

The following example output from the **debug ethernet lmi all** command shows detailed information about the user-network interfaces (UNIs) and Ethernet virtual connections (EVCs) for packet messages.

```
Device# debug ethernet lmi all

Ethernet LMI errors debugging is on
Ethernet LMI ha debugging is on
Ethernet LMI packets debugging is on
Ethernet LMI events debugging is on
Ethernet LMI packets hex debugging is on
Jun 16 18:59:49.372: ELMI Gi0/1 PKT HEX: RX<-:0x017D0101000202D30103050000000004
Jun 16 18:59:49.372: ELMI Gi0/1 PACKET: Incoming
                Protocol Version     : 0x1
                Message              : STATUS (0x7D)
                Report Type          : Full
                Sequence Number      : Snd(0xD3), Rcv(0x1)
                Data Instance        : Value(0x4)
                UNI                  : Bundle
                  UNI Id             : 'uni_sandiego'
                EVC Status           : Evc Ref(0x1), New, Active
                  EVC Parameters     : Point-to-Point
                  EVC Id             : 'EVC_P2P_110'
                  Remote UNI Sum     : Cfgd(1), Up(1)
                EVC Status           : Evc Ref(0x2), New, Active
                  EVC Parameters     : MultiPoint-to-MultiPoint
                  EVC Id             : 'EVC_MP2MP_101'
                  Remote UNI Sum     : Cfgd(2), Up(2)
                CEVLAN EVC Map       : Evc Ref(0x1), Seq(0x1)
                  EVC Map            : Num Vlans(1), 110
                CEVLAN EVC Map       : Evc Ref(0x2), Seq(0x1)
                  EVC Map            : Num Vlans(1), 101
                Remote UNI Status    : Evc Ref(0x1), Uni Ref(0x26), Up
                  UNI Id         o deb al    : 'cisco_newyork'
                Remote UNI Status    : Evc Ref(0x2), Uni Ref(0x1D), Up
                  UNI Id             : 'uni_newyork'
                Remote UNI Status    : Evc Ref(0x2), Uni Ref(0x96), Up
                  UNI Id             : 'miami-detroit'
Jun 16 18:59:49.372: ELMI Gi0/1 EVENT: ce_event: State 0x1, Event 0x0
Jun 16 18:59:49.372: ELMI Gi0/1 EVENT: Update seq: current send 0x1 rcv 0x0
Jun 16 18:59:49.372: ELMI Gi0/1 EVENT: Update uni:
Jun 16 18:59:49.372: ELMI Gi0/1 EVENT: Update evc_sts: ref_id: 0x1
Jun 16 18:59:49.372: ELMI Gi0/1 EVENT: Update evc_param: type 0x0
Jun 16 18:59:49.372: ELMI Gi0/1 EVENT: Update evc_id
Jun 16 18:59:49.372: ELMI Gi0/1 EVENT: Update remote_uni_sum cfgd 1 up 1
Jun 16 18:59:49.372: ELMI Gi0/1 EVENT: Update evc_sts: ref_id: 0x2
Jun 16 18:59:49.372: ELMI Gi0/1 EVENT: Update evc_param: type 0x1
Jun 16 18:59:49.372: ELMI Gi0/1 EVENT: Update evc_id
Jun 16 18:59:49.372: ELMI Gi0/1 EVENT: Update remote_uni_sum cfgd 2 up 2
Jun 16 18:59:49.372: ELMI Gi0/1 EVENT: Update cevlan_evc_map: ref_id: 0x1 seq#1
Jun 16 18:59:49.372: ELMI Gi0/1 EVENT: Update evc_map: num_vlans 1
Jun 16 18:59:49.372: ELMI Gi0/1 EVENT: Update cevlan_evc_map: ref_id: 0x2 seq# 1
Jun 16 18:59:49.372: ELMI Gi0/1 EVENT: Update evc_map: num_vlans 1
Jun 16 18:59:49.372: ELMI Gi0/1 EVENT: Update remote_uni_det: evc ref_id: 0x1 u6
Jun 16 18:59:49.372: ELMI Gi0/1 EVENT: Update remote_uni_det: evc ref_id: 0x2 uD
Jun 16 18:59:49.372: ELMI Gi0/1 EVENT: Update remote_uni_det: evc ref_id: 0x2 u6
Jun 16 18:59:49.372: ELMI Gi0/1 EVENT: upd_lmi_db: new uni_evc ref 0x1
Jun 16 18:59:49.372: ELMI Gi0/1 EVENT: upd_lmi_db: new uni_evc ref 0x2
Jun 16 18:59:49.372: %ETHER_LMI-6-MISMATCHED_VLAN_NOT_CONFIGURED: VLAN 101,110 1
Jun 16 18:59:49.372: %LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthn
Jun 16 18:59:49.376: ELMI Gi0/1 EVENT: Update di: current 0x0 rcvd 0x4
Jun 16 18:59:49.376: ELMI Gi0/1 EVENT: Old State=0x1, Event=0x0, New State=0x0
Jun 16 18:59:49.376: ELMI Gi0/1 EVENT: Updated Stat Type: ETHER_LMI_ST_LFULL_MSD
Jun 16 18:59:50.100: %SYS-5-CONFIG_I: Configured from console by console
Jun 16 18:59:59.376: ELMI Gi0/1 EVENT: ce_event: State 0x0, Event 0x4
Jun 16 18:59:59.376: ELMI Gi0/1 EVENT: Old State=0x0, Event=0x4, New State=0x2
Jun 16 18:59:59.376: ELMI Gi0/1 EVENT: Updated Stat Type: ETHER_LMI_ST_LMSG_SENT
```

The following example shows output of the **debug ethernet lmi all interface** command.

```
Device# debug ethernet lmi all interface ethernet 0/0

Ethernet LMI errors debugging is on for Ethernet0/0
```

```
Ethernet LMI ha debugging is on for Ethernet0/0
Ethernet LMI packets debugging is on for Ethernet0/0
Ethernet LMI events debugging is on for Ethernet0/0
Ethernet LMI packets hex debugging is on for Ethernet0/0
00:45:14: ELMI Et0/0 EVENT: ce_event: State 0x0, Event 0x4
00:45:14: ELMI Et0/0 EVENT: Old State=0x0, Event=0x4, New State=0x2
00:45:14: ELMI Et0/0 EVENT: Updated Stat Type: ETHER_LMI_ST_LMSG_SENT
00:45:14: ELMI Et0/0 PKT HEX: TX->:0x017501010102021312030500000000400000000000
00:45:14: ELMI Et0/0 PACKET: Outgoing
                Protocol Version     : 0x1
                Message              : STATUS ENQ (0x75)
                Report Type          : Check
                Sequence Number      : Snd(0x13), Rcv(0x12)
                Data Instance        : Value(0x4)
00:45:14: ELMI Et0/0 PKT HEX: RX<-:0x017D01010102021313030500000000400000000000
00:45:14: ELMI Et0/0 PACKET: Incoming
                Protocol Version     : 0x1
                Message              : STATUS (0x7D)
                Report Type          : Check
                Sequence Number      : Snd(0x13), Rcv(0x13)
                Data Instance        : Value(0x4)
00:45:14: ELMI Et0/0 EVENT: ce_event: State 0x2, Event 0x1
00:45:14: ELMI Et0/0 EVENT: Update seq: current send 0x13 rcv 0x12
00:45:14: ELMI Et0/0 EVENT: Updated Stat Type: ETHER_LMI_ST_LMSG_RCVD
00:45:14: ELMI Et0/0 EVENT: Old State=0x2, Event=0x1, New State=0x0
```

debug ethernet nid

To display detailed debugging information related to the Network Interface Device (NID) functionality, use the **debug ethernet nid** command in the privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ethernet nid {**configuration | packet egress | packet ingress**}

no debug ethernet nid {**configuration | packet egress | packet ingress**}

Syntax Description

configuration	Enables debugging of configuration-related issues.
packet egress	Enables debugging of packet-processing-related (VLAN tag push) issues on the egress side.
packet ingress	Enables debugging of packet-processing-related (VLAN tag pop) issues on the ingress side.

Command Default

Debugging is disabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
15.4(1)T	This command was introduced into Cisco IOS Release 15.4(1)T.

Usage Guidelines

Use the **debug ethernet nid** command to troubleshoot issues related to the NID functionality configured on a Layer 3 interface on the following routers:

- Cisco 3900 Series, Cisco 2900 Series, and Cisco 1900 Series Integrated Services Routers Generation 2
- Cisco 890 Series Integrated Services Routers

Examples

The following is a sample output of the **debug ethernet nid configuration** command:

```
Device# debug ethernet nid configuration

*Nov 20 01:23:40.643:  Port Tagging mode issued on  GigabitEthernet0/0 *Nov 20 01:23:40.651:
  Port Tagging mode
issued on  GigabitEthernet0/0 *Nov 20 01:23:40.651: Successfully destroyed NID HW Subblock
 *Nov 20 01:23:44.995:
Port Tagging mode issued on  GigabitEthernet0/0 *Nov 20 01:23:45.023:  Configuring vlan_type
 65, vlan_dot1q_id 1000
*Nov 20 01:23:45.023:  No encap configuration existing on the interface *Nov 20 01:23:45.023:
```

```
 Successfully created
NID HW SB on Gi0/0 *Nov 20 01:23:45.023: Successfully added encap to the subblock *Nov 20
01:23:45.023: nid vlan type
is 65, vlan id is 1000, is_cos_cfgd is 0, cos is 0 *Nov 20 01:23:45.039:  Cofiguring cos
value 5 *Nov 20 01:23:45.
039: Successfully modified cos value *Nov 20 01:23:45.039: nid vlan type is 65, vlan id is
 1000, is_cos_cfgd is 1,
cos is 5
```

debug ethernet oam

To enable all Ethernet operations, administration, and maintenance (OAM) debugging, use the **debug ethernet oam**command in privileged EXEC mode. To disable Ethernet OAM debuging, use the **no** form of this command.

debug ethernet oam {**all**| **config**| **ha**| **link-monitor**| **loopback**| **packet** {**decode**| **rx**| **tx**}| **sm**}

no debug ethernet oam {**all**| **config**| **ha**| **link-monitor**| **loopback**| **packet** {**decode**| **rx**| **tx**}| **sm**}

Syntax Description

all	Debugging for all Ethernet OAM flags is on.
config	Debugging for Ethernet OAM configurations is on.
ha	Debugging for Ethernet OAM high-availability events is on.
link-monitor	Debugging for Ethernet OAM link monitoring is on.
loopback	Debugging for Ethernet OAM loopback messages is on.
packet	Debugging for Ethernet OAM protocol data units (PDUs) is on.
decode	Decoding for ingress or egress OAMPDUs, or both, is on.
rx	Debugging for Ethernet ingress OAMPDUs is on.
tx	Debugging for Ethernet egress OAMPDUs is on.
sm	Debugging for the Ethernet OAM state machine is on.

Command Default

All Ethernet OAM debug commands are enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(33)SRA	This command was introduced.
12.4(15)T	This command was integrated into Cisco IOS Release 12.4(15)T.

Release	Modification
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.
Cisco IOS XE Release 3.5S	This command was integrated into Cisco IOS XE Release 3.5S.

Usage Guidelines

When you use the **all** keyword, keep in mind the number of interfaces supporting Ethernet OAM. If many interfaces are supported, some messages may be lost and system performance can degrade.

The **ha** keyword is not available in Cisco IOS Release 12.4(15)T.

The output from this command is a log of activity. Use this command to troubleshoot Ethernet OAM in your network.s

Examples

The following example shows output of the **debug ethernet oam all** command:

```
Router# debug ethernet oam all
*Aug 17 14:00:53.732:     ether_oam_port Gi2/9: during state INACTIVE,
got event 3(link_up)
*Aug 17 14:00:53.732: @@@ ether_oam_port Gi2/9: INACTIVE -> FAULT
*Aug 17 14:00:53.732:     ether_oam_port Gi2/9: idle during state FAULT
*Aug 17 14:00:53.732: @@@ ether_oam_port Gi2/9: FAULT -> FAULT2
*Aug 17 14:00:53.732:     ether_oam_port Gi2/9: during state FAULT2, got
event 6(mode_active)
*Aug 17 14:00:53.732: @@@ ether_oam_port Gi2/9: FAULT2 -> ACTIVE_SEND_LOCAL
*Aug 17 14:00:54.212: EOAM RX PAK(Gi2/9):
*Aug 17 14:00:54.212: 03 00 08 00 01 10 01 00 00 00 0D 05 DC 00 00 0C *Aug 17 14:00:54.212:
 00 00 00 01 00 00 00 00 00 00 00 00 00 00 00 00
*Aug 17 14:00:54.212: 00 00 00 00 00 00 00 00
*Aug 17 14:00:54.212:     ether_oam_port Gi2/9: during state
SEND_LOCAL_REMOTE, got event 8(local_satisfied)
1w5d: %ETHERNET_OAM-6-ENTER_SESSION: The client on interface Gi2/11 has entered the OAM
session.
*Aug 17 14:00:55.212: EOAM RX PAK(Gi2/9):
*Aug 17 14:00:55.212: 03 00 50 00 01 10 01 00 00 00 0D 05 DC 00 00 0C
*Aug 17 14:00:55.212: 00 00 00 SYMPRD w=104857600 lt=1 ht=0
elapsed_time=1032(ms) rx_sym=1000000000 err_sym=0 *Aug 17 14:00:55.740: EOAM LM(Gi2/9): FRM
 w=1 lt=1 ht=0 t_frm=0 err_frm=0
*Aug 17 14:00:55.740: EOAM LM(Gi2/9): FRMPRD w=10000000 lt=1 ht=0 t_frm=1 err_frm=0
*Aug 17 14:00:55.740: EOAM LM(Gi2/9): w=1 lt=10 ht=0 err_frm=0
*Aug 17 14:00:55.740: EOAM LM(Gi2/9): w=1 lt=10 ht=0 err_frm=0
*Aug 17 14:00:55.832: EOAM TX PAK(Gi2/9):
*Aug 17 14:00:55.832: 03 00 50 00 0 1 10 01 00 00 00 0D 05 DC 00 00 0C
*Aug 17 14:00:55.832: 00 00 00 01 02 10 01 00 00 00 0D 05 DC 00 00 0C *Aug 17 14:00:55.832:
 00 00 00 01 *Aug 17 14:00:55.832: EOAM TX PAK(Gi2/9): 00 00 00 0D 05 DC 00 00 0C
*Aug 17 14:00:56.212: 00 00 00 01 02 10 01 00 00 00 0D 05 DC 00 00 0C
*Aug 17 14:00:56.212: 00 00 00 01 00 00 00 00 00 00 00 00 00 00
*Aug 17 14:00:56.212: EOAM RX PAK(Gi2/9): infotlv w/ same revision *Aug 17 14:00:56.820:
EOAM LM(Gi2/9): SYMPRD w=104857600 lt=1 ht=0
elapsed_time=1000(ms) rx_sym=1000000000 err_sym=0
*Aug 17 14:00:56.820: EOAM LM(Gi2/9): FRM w=1 lt=1 ht=0 t_frm=0 err_frm=0
*Aug 17 14:00:56.820: EOAM LM(Gi2/9): 05 FRMPRD w=10000000 lt=1 ht=0 t_frm=3 err_frm=0
*Aug 17 14:00:57.820: EOAM LM(Gi2/9): w=1 lt=10 ht=0 err_frm=0
*Aug 17 14:00:57.820: EOAM LM(Gi2/9): w=1 lt=10 ht=0 err_frm=0
*Aug 17 14:00:57.856: EOAM TX PAK(Gi2/9):
*Aug 17 14:00:57.856: 03 00 50 00 01 10 01 00 00 00 0D 05 DC 00 00 0C
*Aug 17 14:00:57.856: 00 00 00 01 02 10 01 00 00 00 0D 17 14:00:58.212: 05 DC 00 00 0C
*Aug 17 14:00:57.856: 00 00 00 01
*Aug 17 14:00:57.856: EOAM TX PAK(Gi2/9): sent OAMPDU w/ op=0
*Aug 17 14:00:58.212: EOAM RX PAK(Gi2/9):
*Aug 17 14:00:58.212: EOAM RX PAK(Gi2/9): infotlv w/ same revision
*Aug 17 14:00:58.820: EOAM LM(Gi2/9): SYMPRD w=104857600 lt=1 ht=0
elapsed_time=1000(ms) rx_sym=1000000000 err_sym=0
```

```
*Aug 17 14:00:58.820: EOAM LM(Gi2/9): FRM w=1 lt=1 ht=0 t_frm=0 err_frm=0
*Aug 17 14:00:58.820: EOAM LM(Gi2/9): FRMPRD w=10000000 lt=1 ht=0 t_frm=4 err_frm=0
*Aug 17 14:00:58.820: EOAM LM(Gi2/9): w=1 lt=10 ht=0 err_frm=0
*Aug 17 14:00:58.820: EOAM LM(Gi2/9): w=1 lt=10 ht=0 err_frm=0
*Aug 17 14:00:58.856: EOAM TX PAK(Gi2/9):
*Aug 17 14:00:58.856: 03 00 50 00 01 10 01 00 00 00 0D 05 DC 00 00 0C
*Aug 17 14:00:58.856: 00 00 00 01 02 10 01 00 00 00 0D 05 DC 00 00 0C
*Aug 17 14:00:58.856: 00 sent OAMPDU w/ op=0w=1 lt=10 ht=0 err_frm=0
*Aug 17 14:00:59.856: EOAM TX PAK(Gi2/9):
*Aug 17 14:00:59.856: 03 00 50 00 01 10 01 00 00
*Aug 17 14:01:00.832: EOAM LM(Gi2/9): SYMPRD w=104857600 lt=1 ht=0
elapsed_time=1008(ms) rx_sym=1000000000 err_sym=0
*Aug 17 14:01:00.832: EOAM LM(Gi2/9): FRM w=1 lt=1 ht=0 t_frm=0 err_frm=0
*Aug 17 14:01:00.832: EOAM LM(Gi2/9): FRMPRD w=10000000 lt=1 ht=0 t_frm=6 err_frm=0
*Aug 17 14:01:00.832: EOAM LM(Gi2/9): w=1 lt=10 ht=0 err_frm=0
*Aug 17 14:01:00.832: EOA M LM(Gi2/9): w=1 lt=10 ht=0 err_frm=0
*Aug 17 14:01:00.856: EOAM TX PAK(Gi2/9):
*Aug 17 14:01:00.856: 03 00 50 00 01 10 01 00 00 00 0D 05 DC 00 00 0C
*Aug 17 14:01:00.856: 00 00
```

debug ethernet ring g8032 errors

To enable debugging of Ethernet Ring Protocol (ERP) errors, use the **debug ethernet ring g8032 errors** command in privileged EXEC mode.

debug ethernet ring g8032 errors [*ring-name* [**instance** *instance-id*]]

Syntax Description

ring-name	(Optional) Ethernet ring name.
instance *instance-id*	(Optional) Enter the **instance** keyword followed by the instance identifier.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS XE Release 3.6S	This command was introduced.
15.2(4)S	This command was integrated into Cisco IOS Release 15.2(4)S.

Usage Guidelines

This command can be used to debug the errors for all ERP instances in an ERP ring, for an ERP instance for a specified ERP ring, or for all ERP instances configured on the device.

Examples

The following example shows how to enable the **debug ethernet ring g8032 errors** command. Output is generated only when error conditions are encountered.

```
Device# debug ethernet ring g8032 errors
```

debug ethernet ring g8032 events

To enable debugging of Ethernet Ring Protocol (ERP) events, use the **debug ethernet ring g8032 events** command in privileged EXEC mode.

debug ethernet ring g8032 events [*ring-name* [**instance** *instance-id*]]

Syntax Description

ring-name	(Optional) Ethernet ring name.
instance *instance-id*	(Optional) Enter the **instance** keyword followed by the instance identifier.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS XE Release 3.6S	This command was introduced.
15.2(4)S	This command was integrated into Cisco IOS Release 15.2(4)S.

Usage Guidelines

This command can be used to debug the events for all ERP instances in an ERP ring, for an ERP instance for a specified ERP ring, or for all ERP instances configured on the device.

Examples

The following example shows how to enable the **debug ethernet ring g8032 events** command. Output is generated only when error conditions are encountered.

```
Device# debug ethernet ring g8032 events
```

debug ethernet ring g8032 fsm

To enable debugging of Finite State Machine (FSM) state changes for Ethernet Ring Protocol (ERP) instances, use the **debug ethernet ring g8032 fsm** command in privileged EXEC mode.

debug ethernet ring g8032 fsm [*ring-name* [**instance** *instance-id*]] [**detail**]

Syntax Description

ring-name	(Optional) Ethernet ring name.
instance *instance-id*	(Optional) Enter the **instance** keyword followed by the instance identifier.
detail	(Optional) Displays detailed information.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS XE Release 3.6S	This command was introduced.
15.2(4)S	This command was integrated into Cisco IOS Release 15.2(4)S.

Usage Guidelines

This command can be used to debug the FSM state changes for all ERP instances in an ERP ring, for an ERP instance for a specified ERP ring, or for all ERP instances configured on the device.

Examples

The following example shows how to enable the **debug ethernet ring g8032 fsm** command. Output is generated only when error conditions are encountered.

```
Device# debug ethernet ring g8032 fsm
```

debug ethernet ring g8032 packets

To enable debugging of Ethernet Ring Protocol (ERP) packets, use the **debug ethernet ring g8032 packets** command in privileged EXEC mode.

debug ethernet ring g8032 packets [*ring-name* [**instance** *instance-id*]] [**detail**]

Syntax Description

ring-name	(Optional) Ethernet ring name.
instance *instance-id*	(Optional) Enter the **instance** keyword followed by the instance identifier.
detail	(Optional) Displays detailed information.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS XE Release 3.6S	This command was introduced.
15.2(4)S	This command was integrated into Cisco IOS Release 15.2(4)S.

Usage Guidelines

This command can be used to debug the packets for all ERP instances in an ERP ring, for an ERP instance for a specified ERP ring, or for all ERP instances configured on the device.

Examples

The following example shows how to enable the **debug ethernet ring g8032 packets** command. Output is generated only when error conditions are encountered.

```
Device# debug ethernet ring g8032 packets
```

debug ethernet service

To enable debugging of Ethernet customer service instances, use the **debug ethernet service** command in privileged EXEC mode. To disable debugging, use the **no** form of this command.

debug ethernet service {**all**| **api**| **error**| **evc** [*evc-id*]| **ha**| **instance** [**id** *id*| **interface** *type number* [**dynamic**| **mac**]| **qos**]| **interface** *type number*| **microblock**| **oam-mgr**}

no debug ethernet service {**all**| **api**| **error**| **evc**| **ha**| **instance**| **interface**| **microblock**| **oam-mgr**}

Syntax Description

all	Displays all Ethernet customer-service debug messages.
api	Displays debug messages about the interaction between the Ethernet infrastructure and its clients.
error	Displays Ethernet customer service error messages occurring in the Ethernet infrastructure subsystem.
evc	Displays Ethernet virtual circuit (EVC) debug messages.
evc-id	(Optional) String from 1 to 100 characters that identifies an EVC for debugging.
ha	Displays High Availability (HA) Ethernet service debug messages.
instance	Displays debug messages related to Ethernet customer service instances.
id	(Optional) Displays Ethernet service-instance debug messages for a specific Ethernet service instance ID and interface.
id	(Optional) Integer in the range from 1 to 4294967295 that is the service identifier.
interface	Displays debugging for Ethernet services on all interfaces or on a specified interface. (Optional) When used as an option with the **instance** keyword, service instance debug messages for the interface are displayed.
type number	Type and number of the physical interface.
dynamic	(Optional) Displays debug messages for the Ethernet Layer 2 (L2) context dynamic service instances.

mac	(Optional) Displays debug messages for MAC address activity.
qos	Displays debug messages for the Ethernet service quality of service (QoS).
microblock	Displays debug messages for the Ethernet service microblocks.
oam-mgr	Displays debug messages for the Ethernet operations, administration, and maintenance (OAM) manager component of the infrastructure.

Command Default

Ethernet service debugging is disabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(25)SEG	This command was introduced.
12.2(33)SRB	This command was implemented on the Cisco 7600 series routers.
12.2(33)SRD	The **ha** keyword was added.
15.1(2)S	This command was modified. The **dynamic** keyword was added.
Cisco IOS XE Release 3.8S	This command was integrated into Cisco IOS XE Release 3.8S.

Usage Guidelines

The **debug ethernet service** command is useful for troubleshooting. The **undebug ethernet service** command is the same as the **no debug ethernet service** command.

When you use the **evc** keyword without specifying an EVC ID, debugging is enabled for all EVCs on the system.

When you use the **instance** keyword without specifying options, debugging for all service instances is enabled. If a service instance ID and interface are specified, only debug messages for the associated service instance are displayed. If only an interface is specified, debug messages for all service instances on that interface only are displayed.

Examples

The following example shows output after issuing the **debug ethernet service all** command:

```
Device# debug ethernet service all

Ethernet service error debugging is on
```

```
Ethernet service api debugging is on
Ethernet service interface debugging is on
Ethernet service instance debugging is on
Ethernet service instance qos debugging is on
Ethernet service evc debugging is on
Ethernet service OAM Manager debugging is on
Ethernet service ha debugging is on
```

Related Commands

Command	Description
show debugging	Displays information about the types of debugging that are enabled.

debug ethernet service instance dynamic

To enable debugging of Ethernet Layer 2 (L2) context service instances, use the **debug ethernet service instance dynamic**command in privileged EXEC mode. To disable debugging, use the **no** form of this command.

debug ethernet service instance dynamic {**errors**| **events**| **ha** {**errors**| **events**}| **issu** {**errors**| **events**}}

no debug ethernet service

Syntax Description

errors	Displays Ethernet L2 context error messages occurring in the Ethernet infrastructure subsystem.
events	Enables debugging L2 context events.
ha	Enables debugging for High Availability (HA) Ethernet service errors or events.
issu	Enables debugging for In-Service Software Upgrade (ISSU) errors or events.

Command Default

Ethernet L2 context service instance debugging is disabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
15.1(2)S	This command was introduced.

Usage Guidelines

The **debug ethernet service instance dynamic**command is useful for troubleshooting. The **undebug ethernet service instance dynamic**command is the same as the **no debug ethernet service instance dynamic**command.

Examples

The following example shows how to enable Ethernet L2 context service instance debugging:

```
Router# debug ethernet service instance dynamic
```

Related Commands

Command	Description
show debugging	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**| **common**| **detector all**| **detector appl**| **detector cli**| **detector config**| **detector counter**| **detector env**| **detector gold**| **detector interface**| **detector ioswdsysmon**| **detector ipsla**| **detector nf**| **detector none**| **detector oir**| **detector resource**| **detector rf**| **detector routing**| **detector rpc**| **detector snmp**| **detector snmp-notification**| **detector syslog**| **detector test**| **detector timer**| **detector track**| **metricdir**| **policydir**| **server ISSU**| **server events**| **server scheduling**| **snap calls**| **snap errors**| **tcl cli_library**| **tcl commands**| **tcl smtp_library**| **xml parser**}

no debug event manager {**action cli**| **action cns**| **action mail**| **all**| **api calls**| **api errors**| **common**| **detector all**| **detector appl**| **detector cli**| **detector config**| **detector counter**| **detector env**| **detector gold**| **detector interface**| **detector ioswdsysmon**| **detector ipsla**| **detector nf**| **detector none**| **detector oir**| **detector resource**| **detector rf**| **detector routing**| **detector rpc**| **detector snmp**| **detector snmp-notification**| **detector syslog**| **detector test**| **detector timer**| **detector track**| **metricdir**| **policydir**| **server ISSU**| **server events**| **server scheduling**| **snap calls**| **snap errors**| **tcl cli_library**| **tcl commands**| **tcl smtp_library**| **xml parser**}

Syntax Description

action cli	Displays debugging messages about command-line interface (CLI) event messages.
action cns	Displays debugging messages about Cisco Networking Services (CNS) event messages.
action mail	Displays debugging messages about e-mail event messages.
all	Displays all debugging messages.
api calls	Displays debugging messages about EEM client application programming interface (API) calls.
api errors	Displays debugging messages about EEM client API errors.
common	Displays common library code debugging messages.
detector all	Displays all event detector debugging messages.
detector appl	Displays debugging messages about the application-specific event detector. **Note** In Cisco IOS Release 12.4(20)T and later releases, the **application** keyword was replaced with the **appl** keyword.

detector cli	Displays debugging messages about the CLI event detector.
detector config	Displays debugging messages about the config event detector.
detector counter	Displays debugging messages about the counter event detector.
detector env	Displays debugging messages about the environmental event detector.
detector gold	Displays debugging messages about the GOLD event detector.
detector interface	Displays debugging messages about the interface counter event detector.
detector ioswdsysmon	Displays debugging messages about the IOS watchdog event detector.
detector ipsla	Displays debugging messages about the IP SLA event detector.
detector nf	Displays debugging messages about the NetFlow event detector.
detector none	Displays debugging messages about the none event detector.
detector oir	Displays debugging messages about the OIR event detector.
detector resource	Displays debugging messages about the Embedded Resource Manager (ERM) event detector.
detector rf	Displays debugging messages about the redundancy-facility (RF) event detector.
detector routing	Displays debugging messages about the routing event detector.
detector rpc	Displays debugging messages about the remote procedure call (RPC) event detector.
detector snmp	Displays debugging messages about the Simple Network Management Protocol (SNMP) event detector.

detector snmp-notification	Displays debugging messages about the SNMP notification event detector.
detector syslog	Displays debugging messages about the syslog event detector.
detector test	Displays debugging messages about the test event detector.
detector timer	Displays debugging messages about the timer event detector.
detector track	Displays debugging messages about the Enhanced Object Tracking (EOT).
metricdir	Displays debugging messages about the EEM metric event detector.
policydir	Displays debugging messages about the EEM policy director.
server ISSU	Displays debugging messages about In-Service Software Upgrade (ISSU) server events.
server events	Displays debugging messages about the EEM server events.
server scheduling	Displays all debugging messages about the EEM server scheduling events.
snap calls	Displays debugging messages about EEM SNAP client application programming interface (API) calls.
snap errors	Displays debugging messages about EEM SNAP client API errors.
tcl cli_library	Displays all debugging messages about the Tool Command Language (Tcl) command-line interface (CLI) library.
tcl commands	Displays all debugging messages about the Tcl commands.
tcl smtp_library	Displays all debugging messages about the Tcl Simple Mail Transfer Protocol (SMTP) library.
xml parser	Displays debugging messages about the EEM XML parser.

Command Modes Privileged EXEC (#)

Command History

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	The **detector application**, **detector counter**, **detector interface**, **detector ioswdsysmon**,and **detector timer** keywords were added and this command was integrated into Cisco IOS Release 12.2(25)S.
12.3(14)T	The **action cli**, **action mail**, **detector all**, **detector cli**, **detector none**, **detector oir**, and **metricdir**keywords were added.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.4(2)T	The **detector resource**, **detector rf**, and **detector track**keywords were added.
12.2(18)SXF4	The **detector gold** keyword was added and this command was integrated into Cisco IOS Release 12.2(18)SXF4 to support Software Modularity images only.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(18)SXF5	This command was integrated into Cisco IOS Release 12.2(18)SXF5.
12.4(20)T	The **common**, **detector config**, **detector env**, **detector rf**, **detector snmp-notification**, **detector test**, **server ISSU**, and **xml parser** keywords were added and the **detector application** keyword was replaced with the **detector appl** keyword.
12.4(22)T	The **detector ipsla**, **detector nf**, and **detector routing** keywords were added.
15.1(1)SY	This command was integrated into Cisco IOS Release 15.1(1)SY.

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

Note 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
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_hndlr: 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_hndlr: 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
```

The below table describes the significant fields shown in the display.

Table 6: 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 an 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
```

The below table describes the significant fields shown in the display.

Table 7: debug events Field Descriptions

Field	Description
PLIM type	Indicates the interface rate in Mbps. Possible values are: • 1 = TAXI(4B5B) 100 Mbps • 2 = SONET 155 Mbps • 3 = E3 34 Mbps
state	Indicates current state of the ATM Interface Processor (AIP). Possible values are: • 1 = An ENABLE will be issued soon. • 0 = The AIP will remain shut down.
asr	Defines a bitmask, which indicates actions or completions to commands. Valid bitmask values are: • 0x0800 = AIP crashed, reload may be required. • 0x0400 = AIP detected a carrier state change. • 0x0n00 = Command completion status. Command completion status codes are: • n = 8 Invalid physical layer interface module (PLIM) detected • n = 4 Command failed • n = 2 Command completed successfully • n = 1 CONFIG request failed • n = 0 Invalid value

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
```

debug fax dmsp

To troubleshoot the fax Document Media Service Provider (DMSP), use the **debug fax dmsp** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug fax dmsp [**all**| **default**| **detail**| **error** [**call [informational]**| **software [informational]**]| **event**| **function**| **inout**]

no debug fax dmsp

Syntax Description

all	(Optional) Displays all fax DMSP debugging messages.
default	(Optional) Displays fax DMSP error and inout information. This option also runs if no keywords are added.
detail	(Optional) Displays fax DMSP background messages.
error	(Optional) Displays fax DMSP error messages.
call	(Optional) Displays call processing errors.
informational	(Optional) Displays minor errors and major errors. Without the **informational** keyword, only major errors are displayed.
software	(Optional) Displays software errors.
event	(Optional) Displays fax DMSP events.
function	(Optional) Displays fax DMSP functions.
inout	(Optional) Displays fax DMSP in/out functions.

Command Default

Debugging is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(8)T	This command replaces the **debug dmsp doc-to-fax** and **debug dmsp fax-to-doc** commands.

Examples

The following is sample output from the **debug fax dmsp all** command:

```
Router# debug fax dmsp all

2d07h: //70/67D6061D8012/DMSP/docmsp_call_setup_request:
   ramp data dir=ONRAMP, conf dir=DEST
2d07h: //70/67D6061D8012/DMSP/docmsp_caps_ind:
   cid(0x46), srcCallID(0x44)
2d07h: //70/67D6061D8012/DMSP/docmsp_bridge:
   conf id(0x33), srcCallID(0x46), dstCallID(0x44),
   ramp data dir=ONRAMP, conf dir=DEST, encode out=1
2d07h: //70/67D6061D8012/DMSP/docmsp_bridge:
   Bridge done
2d07h: //70/67D6061D8012/DMSP/docmsp_bridge:
   conf id(0x34), srcCallID(0x46), dstCallID(0x45),
   ramp data dir=ONRAMP, conf dir=SRC, encode out=1
2d07h: //70/67D6061D8012/DMSP/docmsp_bridge:
   Bridge done
2d07h: //70/67D6061D8012/DMSP/docmsp_xmit:
   srcCallID(0x44), dstCallID(0x46), direction=0
2d07h: //68/67D6061D8012/DMSP/docmsp_process_rcv_data:
   evID=0, proto_flag=3, srcCallID(0x44), dstCallID(0x46)
2d07h: //70/67D6061D8012/DMSP_ON/docmsp_tiff_writer_data_process:
   START_OF_CONNECTION
2d07h: //70/67D6061D8012/DMSP_ON/docmsp_tiff_writer_data_process:
   START_OF_FAX_PAGE
2d07h: //70/67D6061D8012/DMSP_ON/docmsp_tiff_writer_get_buffer_callback:
   tiff_segment=0x63A88ECC
2d07h: //70/67D6061D8012/DMSP_ON/docmsp_tiff_writer_get_buffer_callback:
   tiff_segment=0x63D58944
2d07h: //70/67D6061D8012/DMSP/docmsp_process_rcv_data:
   Done
```

The following table describes the significant fields shown in the display.

Table 8: debug fax dmsp Field Descriptions

Field	Description
//70/67D6061D8012/DMSP/ docmsp_call_setup_request:	The format of this message is //callid/GUID/DMSP/function name: • CallEntry ID is 70. This indicates a unique call leg. • GUID is 67D6061D8012. This identifies the call. • DMSP is the module name. • Thedocmsp_call_setup_request field shows that the DMSP is requesting a call setup.
ramp data dir	Indicates if the data direction is on-ramp or off-ramp.
conf dir	Indicates if the data is from the source or destination.
docmsp_bridge:	Indicates that the DMSP is setting up a bridge to the destination.

Field	Description
docmsp_xmit:	Indicates that the DMSP is transmitting.
docmsp_process_rcv_data:	Indicates that the DMSP is starting the process to receive data.
docmsp_tiff_writer_data_process:	Indicates the process that is being started.
docmsp_tiff_writer_get_buffer_ callback:	Indicates the segment for the DMSP TIFF writer get_buffer_callback parameter.

debug fax fmsp

To troubleshoot the Fax Media Service Provider (FMSP), use the **debug fax fmsp** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug fax fmsp [**all**| **default**| **detail**| **error** [**call [informational]**| **software [informational]**]| **event**| **function**| **inout**| **receive**| **send**]

no debug fax fmsp

Syntax Description

all	(Optional) Displays all fax FMSP debugging messages.
default	(Optional) Displays fax FMSP error and inout information. This option also runs if no keywords are added.
detail	(Optional) Displays fax FMSP background messages.
error	(Optional) Displays fax FMSP error messages.
call	(Optional) Displays call processing errors.
informational	(Optional) Displays minor errors and major errors. Without the **informational** keyword, only major errors are displayed.
software	(Optional) Displays software errors.
event	(Optional) Displays fax FMSP events.
function	(Optional) Displays fax FMSP functions.
inout	(Optional) Displays fax FMSP in/out functions.
receive	(Optional) Receives T.30 or T.38 debugs.
send	(Optional) Sends T.30 or T.38 debugs.

Command Default

Debugging is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(8)T	This command replaces the **debug fmsp receive** and **debug fmsp send** commands.

Examples

The following is sample output from the **debug fax fmsp all** command:

```
Router# debug fax fmsp all

2d08h: //76/90A52CB88014/FMSP/faxmsp_call_setup_request:
   session(0x63A8A474), vdbPtr(0x62CA45A8),data dir=ONRAMP, conf dir=DEST
2d08h: //76/90A52CB88014/FMSP/faxmsp_bridge:
   confID(0x38), srcCID(0x4C), dstCID(0x4B)
2d08h: //76/90A52CB88014/FMSP/faxmsp_bridge:
   ramp data dir=ONRAMP, conf dir=DEST
2d08h: //76/90A52CB88014/FMSP/faxmsp_bridge:
   Explicit caps ind. done; Wait for registry cap ind
2d08h: //76/90A52CB88014/FMSP/faxmsp_caps_ind:
   per_bridge_info(0x63D52FD8), cap_ind_state(0x6)
2d08h: //76/90A52CB88014/FMSP/faxmsp_caps_ind[1617]:
2d08h: //76/90A52CB88014/FMSP/faxmsp_caps_ack:
   direction=0, srcCID(0x4B), dstCID(0x0)
2d08h: //76/90A52CB88014/FMSP/faxmsp_codec_download_done:
   per_bridge_info(0x63D52FD8), application_data(0x63C54698), state(0x2), direction=2
2d08h: //76/90A52CB88014/FMSP/faxMsp_get_tx_buffer:
   event(0x402897C0), bufferBegin(0x63A86B5C), dataBegin(0x402897EC)
2d08h: //76/90A52CB88014/FMSP/faxMsp_get_tx_buffer:
   event(0x40289B50), bufferBegin(0x63C55794), dataBegin(0x40289B7C)
2d08h: //76/90A52CB88014/FMSP/faxMsp_get_tx_buffer:
   event(0x40851E58), bufferBegin(0x63D4EAE4), dataBegin(0x40851E84)
2d08h: //76/90A52CB88014/FMSP/faxmsp_xmit:
   srcCallID(0x4B), dstCallID(0x4C)
2d08h: //76/90A52CB88014/FMSP/faxmsp_xmit[1813]:
2d08h: //76/90A52CB88014/FMSP/faxmsp_process_rcv_data:
   state(0x1), evID=918065, evProtoFlag=2
2d08h: //76/90A52CB88014/FMSP/t38_rx_buffer:
   t38 rx msg dump (size=6):
   00 00 01 00 00 00
2d08h: //76/90A52CB88014/FMSP/faxmsp_process_rcv_data[1994]:
2d08h: //76/90A52CB88014/FMSP/fax2_phaseB_receive:
   CSI_PACKET(8881111)
   DIS_PACKET(speed=5, resolution=1, encoding=1)
2d08h: //76/90A52CB88014/FMSP/faxMsp_get_tx_buffer:
   event(0x40289690), bufferBegin(0x63A7E798), dataBegin(0x402896BC)
2d08h: //76/90A52CB88014/FMSP/faxMsp_get_tx_buffer:
   event(0x4028858C), bufferBegin(0x63B89AC0), dataBegin(0x402885B8)
2d08h: //76/90A52CB88014/FMSP/t38_tx_command:
   t38 tx msg dump (size=47):
   00 04 2A C0 04 80 00 16 FF C0 02 8C 8C 8C 8C 1C 1C 1C 04 04
   04 04 04 04 04 04 04 04 04 04 28 00 09 FF C8 01 00 77 1F
   01 01 19 80 40 00 00
2d08h: //76/90A52CB88014/FMSP/fax2_phaseB_receive:
   fax2_response_receive, PROCESSING
```

The below table describes the significant fields shown in the display.

Table 9: debug fax fmsp all Field Descriptions

Field	Description
//76/90A52CB88014/FMSP/ faxmsp_call_setup_request:	The format of this message is //callid/GUID/FMSP/function name • CallEntry ID is 76. This indicates a unique call leg. • GUID is 90A52CB88014. This identifies the call. • FMSP is the module name. • Thefaxmsp_call_setup_requestfield shows that the FMSP is requesting a call setup.
ramp data dir	Indicates if the data direction is on-ramp or off-ramp.
conf dir	Indicates if the data is from the source or destination.
faxmsp_bridge:	Indicates that the FMSP is setting up a bridge to the destination.
faxmsp_xmit:	Indicates that the FMSP is transmitting data.
faxmsp_process_rcv_data:	Indicates that the FMSP is beginning the process to receive data.
t38_rx_buffer:	Shows the contents of the T.38 transmit buffer.
t38_tx_command:	Shows the T.38 transmit command.

debug fax foip

To troubleshoot fax mail, use the **debug fax foip** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug fax foip [**all**| **default**| **detail**| **error** [**call [informational]**| **software [informational]**]| **event**| **function**| **inout**]

no debug fax foip

Syntax Description

all	(Optional) Displays all fax mail debugging messages.
default	(Optional) Displays fax mail error and inout information. This option also runs if no keywords are added.
detail	(Optional) Displays fax mail background messages.
error	(Optional) Displays fax mail error messages.
call	(Optional) Displays call processing errors.
informational	(Optional) Displays minor errors and major errors. Without the **informational** keyword, only major errors are displayed.
software	(Optional) Displays software errors.
event	(Optional) Displays fax mail events.
function	(Optional) Displays fax mail functions.
inout	(Optional) Displays fax mail in/out functions.

Command Default

Debugging is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(8)T	This command replaces the **debug foip off-ramp** and **debug foip on-ramp** commands.

Examples

The following is sample output from the **debug fax foip all** command:

```
Router# debug fax foip all

2d07h: //35/67E715B7800A/FOIP_ON/lapp_on_call_handoff:
   Authentication:
     Id: 0
     Method: IVR or unknown
     Status: SUCCESS
     Enabled: FALSE
     Template:
     List: fax
     MailtoAddress: Calling Oct3A=0x0
2d07h: //35/67E715B7800A/FOIP_ON/lapp_on_conference_vtsp_fmsp:
   Begin Conferencing VTSP and FMSP
2d07h: //35/67E715B7800A/FOIP_ON/lapp_on_conference_vtsp_fmsp[887]:
2d07h: //35/67E715B7800A/FOIP_ON/lapp_on_change_state:
   Old State=0, New State=1
2d07h: //35/67E715B7800A/FOIP_ON/lapp_on_call_handoff[2953]:
2d07h: //35/67E715B7800A/FOIP_ON/lapp_on_validate_context[930]:
2d07h: //35/67E715B7800A/FOIP_ON/lapp_on_conference_created:
   VTSP and FMSP Are Conferenced;
   Waiting for FMSP Call Detail Event
2d07h: //35/67E715B7800A/FOIP_ON/lapp_on_change_state:
   Old State=1, New State=2
2d07h: %ISDN-6-CONNECT: Interface Serial2:30 is now connected to unknown
2d07h: //35/67E715B7800A/FOIP_ON/lapp_on_validate_context[930]:
2d07h: //35/67E715B7800A/FOIP_ON/lapp_on_msp_event:
2d07h: //35/67E715B7800A/FOIP_ON/lapp_on_setup_mspi:
   Prepare MSPI Call Setup Request
2d07h: //35/67E715B7800A/FOIP_ON/lapp_on_setup_mspi:
   Envelope From=FAX=7771111@cisco.com
2d07h: //35/67E715B7800A/FOIP_ON/lapp_on_setup_mspi:
   Envelope To=jdoe@server.cisco.com
2d07h: //35/67E715B7800A/FOIP_ON/lapp_on_setup_mspi:
   RFC822 To Comment=dileung
2d07h: //35/67E715B7800A/FOIP_ON/lapp_on_setup_mspi:
   Faxmail Subject=hagar-c5300-bw12 subject line here
2d07h: //35/67E715B7800A/FOIP_ON/lapp_on_setup_mspi:
   Disposition Notification=
2d07h: //35/67E715B7800A/FOIP_ON/lapp_on_setup_mspi:
   Originator TSI=RFC822 From Comment=
2d07h: //35/67E715B7800A/FOIP_ON/lapp_on_setup_mspi:
   Auth/Account ID: `0'
2d07h: //35/67E715B7800A/FOIP_ON/lapp_on_setup_mspi:
   Call Setup Request To MSPI
2d07h: //37/67E715B7800A/FOIP_ON/lapp_on_setup_mspi[748]:
2d07h: //35/67E715B7800A/FOIP_ON/lapp_on_conference_fmsp_dmsp:
    Starting Conference with FMSP and DMSP
2d07h: //35/67E715B7800A/FOIP_ON/lapp_on_conference_fmsp_dmsp:
   Tiff File Created; Time=2003:06:05 22:46:48
```

The below table describes the significant fields shown in the display.

Table 10: debug fax foip all Field Descriptions

Field	Description
//35/67E715B7800A/FOIP_ON/ lapp_on_call_handoff:	The format of this message is //callid/GUID/FOIP_ON/function name: • CallEntry ID is 35. This indicates a unique call leg. • GUID is 67E715B7800A. This identifies the call. • FOIP_ONidentifies the fax mail onramp call leg. FOIP_OFF would identify an offramp call leg. • The lapp_on_call_handofffield shows that the fax mail is initiating a call handoff.
lapp_on_conference_vtsp_fmsp:	Indicates that fax mail is starting a conference for VTSP and FMSP.
lapp_on_change_state	Indicates that the fax mail is changing state.
lapp_on_conference_created	Indicates that the conference is working properly between the VTSP and FMSP.
lapp_on_setup_mspi:	Indicates that fax mail is displaying an MPSI event.
lapp_on_conference_fmsp_dmsp:	Indicates that fax mail is starting a conference for FMSP and DMSP.

debug fax mmoip aaa

To display output relating to authentication, authorization, and accounting (AAA) services using multimedia mail over IP (MMoIP) for the Store and Forward Fax feature, use the **debug fax mmoip aaa** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug fax mmoip aaa [**all**| **default**| **error** [**call [informational]**| **software [informational]**]| **inout**]

no debug fax mmoip aaa

Syntax Description

all	(Optional) Displays all MMoIP AAA debugging messages.
default	(Optional) Displays MMoIP AAA error and inout information. This option also runs if no keywords are added.
error	(Optional) Displays MMoIP AAA error messages.
call	(Optional) Displays call processing errors.
informational	(Optional) Displays minor errors and major errors. Without the **informational** keyword, only major errors are displayed.
software	(Optional) Displays software errors.
inout	(Optional) Displays MMoIP AAA in/out functions.

Command Default

Debugging is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(8)T	This command replaces the **debug mmoip aaa** command.

Examples

The following example shows output from the **debug fax mmoip aaa all** command for an onramp fax connection:

```
Router# debug fax mmoip aaa all
```

```
16:22:04: //3/D9242FD08002/MMOIP_AAA_ON/mmoip_aaa_accounting_onramp:
   UID=3
16:22:04: //3/D9242FD08002/MMOIP_AAA_ON/mmoip_aaa_accounting_onramp:
   fax_account_id_origin=NONE_ID
16:22:04: //3/D9242FD08002/MMOIP_AAA_ON/mmoip_aaa_accounting_onramp:
   fax_msg_id=00012003151904623@Router.cisco.com, Length=39
16:22:04: //3/D9242FD08002/MMOIP_AAA_ON/mmoip_aaa_accounting_onramp:
   fax_pages=0
16:22:04: //3/D9242FD08002/MMOIP_AAA_ON/mmoip_aaa_accounting_onramp:
   fax_connect_speed=disable bps
16:22:04: //3/D9242FD08002/MMOIP_AAA_ON/mmoip_aaa_accounting_onramp:
   fax_mdn_flag=FALSE
16:22:04: //3/D9242FD08002/MMOIP_AAA_ON/mmoip_aaa_accounting_onramp:
   fax_auth_status=USER NOT AUTHENTICATED
16:22:04: //3/D9242FD08002/MMOIP_AAA_ON/mmoip_aaa_accounting_onramp:
   email_server_address=172.19.140.112
16:22:04: //3/D9242FD08002/MMOIP_AAA_ON/mmoip_aaa_accounting_onramp:
   email_server_ack_flag=TRUE
16:22:04: //3/D9242FD08002/MMOIP_AAA_ON/mmoip_aaa_accounting_onramp:
   gateway_id=Router.cisco.com
16:22:04: //3/D9242FD08002/MMOIP_AAA_ON/mmoip_aaa_accounting_onramp:
   call_type=Fax Receive
16:22:04: //3/D9242FD08002/MMOIP_AAA_ON/mmoip_aaa_accounting_onramp:
   abort_cause=10
```

The below table describes the significant fields shown in the display.

Table 11: debug fax mmoip aaa all Field Descriptions

Field	Description
//3/D9242FD08002/ MMOIP_AAA_ON/ mmoip_aaa_accounting_onramp	The format of this message is //callid/GUID/module name/function name: • CallEntry ID is 3. This indicates a unique call leg. • GUID is D9242FD08002. This identifies the call. • MMOIP_AAA_ONidentifies the fax mail onrampMMOIP AAA call leg. MMOIP_AAA_OFF would identify the offramp call leg. • Themmoip_aaa_accounting_onrampfield shows that the accounting for an onramp fax is active.
fax_msg_id=00012003151904623 @Router.cisco.com	Displays the fax message ID.
gateway_id=Router.cisco.com	Displays the name of the router.
call_type=Fax Receive	Indicates that the fax is being received.

debug fax mspi

To troubleshoot the fax Mail Service Provider Interface (MSPI), use the **debug fax mspi** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug fax mspi [**all**| **default**| **detail**| **error** [**call [informational]**| **software [informational]**]| **event**| **function**| **inout**]

no debug fax mspi

Syntax Description

all	(Optional) Displays all fax MSPI debugging messages.
default	(Optional) Displays fax MSPI error and inout information. This option also runs if no keywords are added.
detail	(Optional) Displays fax MSPI background messages.
error	(Optional) Displays fax MSPI error messages.
call	(Optional) Displays call processing errors.
informational	(Optional) Displays minor errors and major errors. Without the **informational** keyword, only major errors are displayed.
software	(Optional) Displays software errors.
event	(Optional) Displays fax MSPI events.
function	(Optional) Displays fax MSPI functions.
inout	(Optional) Displays fax MSPI in/out functions.

Command Default

Debugging is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(8)T	This command replaces the **debug mspi receive** and **debug mspi send** commands.

Examples

The following is sample output from the **debug fax mspi all** command:

```
Router# debug fax mspi all

Router#
2d07h: %ISDN-6-CONNECT: Interface Serial2:30 is now connected to unknown
2d07h: //41/ACF704FA800B/MSPI_ON/mspi_call_setup_request:
   Outgoing Peer Tag=22
   Envelope From=FAX=5550121@cisco.com
   Envelope To=jdoe@server.cisco.com
   Mime Outer Type=2
2d07h: //41/ACF704FA800B/MSPI_ON/mspi_check_connect:
   MMccb(Count=0)
2d07h: //41/ACF704FA800B/MSPI_ON/mspi_check_connect:
   SMTP Connected To The Server !
2d07h: //41/ACF704FA800B/MSPI/mspi_bridge:
   MMccb(State=CONNECTED, Type=Onramp), Destination Call Id=0x2A
2d07h: //41/ACF704FA800B/MSPI_ON/mspi_xmit:
   MMccb(State=CONFERENCED, Type=Onramp, Buffer Count=0), Source Call Id=0x2A
2d07h: //41/ACF704FA800B/MSPI_ON/mspi_xmit:
   MMccb(State=CONFERENCED, Type=Onramp, Buffer Count=1), Source Call Id=0x2A
2d07h: //41/ACF704FA800B/MSPI_ON/mspi_xmit:
   MMccb(State=CONFERENCED, Type=Onramp, Buffer Count=2), Source Call Id=0x2A
2d07h: //41/ACF704FA800B/MSPI_ON/mspi_xmit:
   MMccb(State=CONFERENCED, Type=Onramp, Buffer Count=3), Source Call Id=0x2A
2d07h: //41/ACF704FA800B/MSPI_ON/mspi_onramp_buff_finished_callback:
   MMccb(Call State=CONFERENCED, Buffer Count=9)
2d07h: //41/ACF704FA800B/MSPI_ON/mspi_onramp_buff_finished_callback:
   MMccb(Call State=CONFERENCED, Buffer Count=8)
2d07h: //41/ACF704FA800B/MSPI_ON/mspi_onramp_buff_finished_callback:
   MMccb(Call State=CONFERENCED, Buffer Count=7)
2d07h: //41/ACF704FA800B/MSPI_ON/mspi_onramp_buff_finished_callback:
2d07h: //41/ACF704FA800B/MSPI_ON/mspi_xmit:
   MMccb(State=CONFERENCED, Type=Onramp, Buffer Count=0), Source Call Id=0x2A
2d07h: //41/ACF704FA800B/MSPI_ON/mspi_xmit:
   MMccb(State=CONFERENCED, Type=Onramp, Buffer Count=1), Source Call Id=0x2A
2d07h: //41/ACF704FA800B/MSPI_ON/mspi_xmit:
   MMccb(State=CONFERENCED, Type=Onramp, Buffer Count=2), Source Call Id=0x2A
2d07h: //41/ACF704FA800B/MSPI_ON/mspi_xmit:
   MMccb(State=CONFERENCED, Type=Onramp, Buffer Count=3), Source Call Id=0x2A
2d07h: //41/ACF704FA800B/MSPI_ON/mspi_xmit:
   MMccb(State=CONFERENCED, Type=Onramp, Buffer Count=4), Source Call Id=0x2A
2d07h: //41/ACF704FA800B/MSPI_ON/mspi_xmit:
   MMccb(State=CONFERENCED, Type=Onramp, Buffer Count=5), Source Call Id=0x2A
Router#
```

The below table describes the significant fields shown in the display.

Table 12: debug fax mspi all Field Descriptions

Field	Description
/41/ACF704FA800B/MSPI_ON/mspi_call_setup_request:	The format of this message is //callid/GUID/module name/function name: • CallEntry ID is 41. This indicates a unique call leg. • GUID is ACF704FA800B. This identifies the call. • MSPI_ON identifies the fax mail onramp MSPI call leg. MSPI_OFF would identify the offramp call leg. • Themspi_call_setup_request field shows that the MSPI is requesting a call setup.
Outgoing Peer Tag=22	Indicates the unique dial peer tag.
Envelope From=FAX=5550121@cisco.com	Indicates the sender of the fax mail message.
Envelope To=jdoe@server.cisco.com	Indicates the receiver of the fax mail message.
mspi_xmit:	Indicates that the MSPI is transmitting data.
State=CONFERENCED	Describes the MPSI state.
Type=Onramp	Describes whether the fax is on-ramp or off-ramp.
Buffer Count=0	Indicates the buffer count.
Source Call Id=0x2A	Identifies the source call ID.

debug fax mta

To troubleshoot the fax Mail Transfer Agent (MTA), use the **debug fax mta** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug fax mta [**all**| **default**| **detail**| **error** [**call [informational]**| **software [informational]**]| **event**| **function**| **inout**]

no debug fax mta

Syntax Description

all	(Optional) Displays all fax MTA debugging messages.
default	(Optional) Displays fax MTA error and inout information. This option also runs if no keywords are added.
detail	(Optional) Displays fax MTA background messages.
error	(Optional) Displays fax MTA error messages.
call	(Optional) Displays call processing errors.
informational	(Optional) Displays minor errors and major errors. Without the **informational** keyword, only major errors are displayed.
software	(Optional) Displays software errors.
event	(Optional) Displays fax MTA events.
function	(Optional) Displays fax MTA functions.
inout	(Optional) Displays fax MTA in/out functions.

Command Default

Debugging is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(8)T	This command replaces the **debug mta receive all**, **debug mta send all**, and **debug mta send rcpt-to** commands.

Examples

The following is sample output from the **debug fax mta all** command:

```
Router# debug fax mta all

2d07h: %ISDN-6-CONNECT: Interface Serial2:30 is now connected to unknown
2d07h: //-1/CEB9FA0B800E/SMTPC/esmtp_client_engine_new_context_guid[2177]:
2d07h: //-1/CEB9FA0B800E/SMTPC/esmtp_client_engine_open:
   from=FAX=7771111@cisco.com, to=jdoe@server.cisco.com
2d07h: //-1/CEB9FA0B800E/SMTPC/esmtp_client_engine_open[1868]:
2d07h: //-1/CEB9FA0B800E/SMTPC/esmtp_client_engine_add_headers:
   from_comment=
2d07h: //-1/CEB9FA0B800E/SMTPC/esmtp_client_engine_work_routine:
   socket 0 readable for first time
2d07h: //-1/CEB9FA0B800E/SMTPC/esmtp_client_engine_getln:
   (C)R: 220 vip2-das.cisco.com ESMTP Sendmail 8.9.3/8.9.3; Thu, 5 Jun 2003 23:24:54 -0700
2d07h: //-1/CEB9FA0B800E/SMTPC/esmtp_client_engine_writeln:
   (C)S: EHLO Router.cisco.com
2d07h: //-1/CEB9FA0B800E/SMTPC/esmtp_client_engine_write:
   return code=0
2d07h: //-1/CEB9FA0B800E/SMTPC/esmtp_client_engine_getln:
   (C)R: 250-vip2-das.cisco.com Hello [172.19.140.108], pleased to meet you
2d07h: //-1/CEB9FA0B800E/SMTPC/esmtp_client_engine_getln:
   (C)R: 250-EXPN
2d07h: //-1/CEB9FA0B800E/SMTPC/esmtp_client_engine_getln:
   (C)R: 250-VERB
2d07h: //-1/CEB9FA0B800E/SMTPC/esmtp_client_engine_getln:
   (C)R: 250-8BITMIME
2d07h: //-1/CEB9FA0B800E/SMTPC/esmtp_client_engine_getln:
   (C)R: 250-SIZE
2d07h: //-1/CEB9FA0B800E/SMTPC/esmtp_client_engine_getln:
   (C)R: 250-DSN
2d07h: //-1/CEB9FA0B800E/SMTPC/esmtp_client_engine_getln:
   (C)R: 250-ONEX
2d07h: //-1/CEB9FA0B800E/SMTPC/esmtp_client_engine_getln:
   (C)R: 250-ETRN
2d07h: //-1/CEB9FA0B800E/SMTPC/esmtp_client_engine_getln:
   (C)R: 250-XUSR
2d07h: //-1/CEB9FA0B800E/SMTPC/esmtp_client_engine_getln:
   (C)R: 250 HELP
2d07h: //-1/CEB9FA0B800E/SMTPC/esmtp_client_engine_writeln:
   (C)S: MAIL FROM:<FAX=7771111@cisco.com>
2d07h: //-1/CEB9FA0B800E/SMTPC/esmtp_client_engine_write:
   return code=0
2d07h: //-1/CEB9FA0B800E/SMTPC/esmtp_client_engine_getln:
   (C)R: 250 <FAX=7771111@cisco.com>... Sender ok
2d07h: //-1/CEB9FA0B800E/SMTPC/esmtp_client_engine_writeln:
   (C)S: RCPT TO:<jdoe@server.cisco.com>
2d07h: //-1/CEB9FA0B800E/SMTPC/esmtp_client_engine_write:
   return code=0
2d07h: //-1/CEB9FA0B800E/SMTPC/esmtp_client_engine_getln:
   (C)R: 250 <jdoe@server.cisco.com>... Recipient ok
2d07h: //-1/CEB9FA0B800E/SMTPC/esmtp_client_engine_write:
   return code=0
2d07h: //-1/CEB9FA0B800E/SMTPC/esmtp_client_engine_getln:
   (C)R: 354 Enter mail, end with "." on a line by itself
2d07h: //-1/CEB9FA0B800E/SMTPC/esmtp_client_engine_writeln:
   (C)S: Received:  by Router.cisco.com for <jdoe@server.cisco.com> (with Cisco NetWorks);
 Thu, 05 Jun 2003 23:11:09 +0000
2d07h: //-1/CEB9FA0B800E/SMTPC/esmtp_client_engine_write:
   return code=0
2d07h: //-1/CEB9FA0B800E/SMTPC/esmtp_client_engine_writeln:
   (C)S: To: "jdoe" <jdoe@server.cisco.com>
2d07h: //-1/CEB9FA0B800E/SMTPC/esmtp_client_engine_write:
   return code=0
2d07h: //-1/CEB9FA0B800E/SMTPC/esmtp_client_engine_writeln:
   (C)S: Message-ID: <00222003231109198@Router.cisco.com>
2d07h: //-1/CEB9FA0B800E/SMTPC/esmtp_client_engine_write:
   return code=0
2d07h: //-1/CEB9FA0B800E/SMTPC/esmtp_client_engine_writeln:
```

```
   (C)S: Date: Thu, 05 Jun 2003 23:11:09 +0000
2d07h: //-1/CEB9FA0B800E/SMTPC/esmtp_client_engine_write:
   return code=0
```

The below table describes the significant fields shown in the display.

Table 13: debug fax mta all Field Descriptions

Field	Description
//-1/CEB9FA0B800E/SMTPC/ esmtp_client_engine_open:	The format of this message is //callid/GUID/module name/function name: • CallEntry ID is -1. This indicates that a call leg has not been identified. • GUID is CEB9FA0B800E. This identifies the call. • SMTPCis the module name. • Theesmtp_client_engine_open field shows that the fax mail client engine is opening a session.
from=FAX=7771111@cisco.com	Indicates the sender of the fax mail message.
to=jdoe@server.cisco.com	Indicates the receiver of the fax mail message.
esmtp_client_engine_writeln:	Indicates that the fax mail client engine is writing data.
esmtp_client_engine_getln:	Indicates that the fax mail client engine is receiving data.

debug fax relay capture-log

To display debug capture log of the fax calls, use the **debug fax relay capture-log** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug fax relay capture-log {{**best-effort** | **called-number** | **calling-number**} {**all-call** | **error-only** | **failure-only**}| **file-destination** *filename* | **smart-match** {**error-only | failure-only**}}

no debug fax relay capture-log

Syntax Description

best-effort all-call	Enables best effort capture for all fax relay sessions.
best-effort error-only	Enables best effort capture for error fax relay sessions.
best-effort failure-only	Enables best effort capture for failure fax relay sessions.
called-number all-call	Enables debug capture for all fax calls processed from this called number.
called-number error-only	Enables debug capture for error fax calls processed to this called number.
called-number failure-only	Enables debug capture for failed fax calls processed to this called number.
calling-number all-call	Enables debug capture for all fax calls processed from this calling number.
calling-number error-only	Enables debug capture for error fax calls processed from this calling number.
calling-number failure-only	Enables debug capture for failed fax calls processed from this calling number.
file-destination *filename*	File URL to capture all fax data logs.
smart-match error-only	Enables debug capture for the fax relay session on DSP error history.
smart-match failure-only	Enables debug capture for the fax relay session on DSP failure history.

Command Default

No default behavior or values

Command Modes Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS 15.4(1)T	This command was introduced.
Cisco IOS XE 3.11S	This command was integrated into Cisco IOS XE 3.11S.

Examples The following command enables debugging for best effort capture for all fax relay sessions:

```
Router# debug fax relay capture-log best-effort all-call
Debugging fax relay capture-log best-effort all
```

Related Commands

Command	Description
debug fax relay event-log	Displays event debug details for all fax relay sessions or calls.
debug fax relay t30	Enables debugging messages for T.30 real-time fax.

debug fax relay event-log

To display event debug details for all fax relay sessions or calls, use the **debug fax relay event-log** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug fax relay event-log {**all-call | error-only | failure-only** | **file-destination** *filename*}

no debug fax relay event-log

Syntax Description

all-call	Displays call log for all fax calls.
error-only	Displays call log for the error fax calls.
failure-only	Displays call log for the failed fax calls.
file-destination *filename*	File URL to capture all fax data logs.

Command Default

No default behavior or values.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS 15.4(1)T	This command was introduced.
Cisco IOS XE 3.11S	This command was integrated into Cisco IOS XE 3.11S.

Examples

The following command enables event debugging details for all fax relay sessions:

```
Router# debug fax relay event-log all-call
Debugging fax relay event-log all-call
```

Related Commands

Command	Description
debug fax relay capture-log	Displays debug capture log of the fax calls.
debug fax relay t30	Enables debugging messages for T.30 real-time fax.

debug fax relay t30

To display debugging messages for T.30 real-time fax, use the **debug fax relay t30** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug fax relay t30 {**all**| **calling-number** *string*| **called-number** *string*}

no debug fax relay t30

Syntax Description

all	Enables debugging for all incoming and outgoing calls.
calling-number	Enables debugging for incoming numbers that begin with a specified string of digits.
called-number	Enables debugging for outgoing numbers that begin with a specified string of digits.
string	Digits that specify the incoming or outgoing number.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(2)XB1	The debug fax relay t30 command was introduced on Cisco AS5350, Cisco AS5400, and Cisco AS5850 access servers.
12.2(11)T	This command was integrated into Cisco IOS Release 12.2(11)T for the Cisco AS5350, Cisco AS5400, and Cisco AS5850 access servers.

Usage Guidelines

The incoming or outgoing numbers must be a valid E.164 destination. The period symbol (.) as a wildcard should not be used. Instead of a wildcard, leave the space blank to indicate that any numbers can be valid.

There are no limits to the number of debug entries. The number entered generates a match if the calling or called number matches up to the final number of the debug entry. For example, the 408555 entry would match 408555, 4085551, 4085551212, or any other number starting with 408555.

Examples

The following command enables debugging for any incoming calls that start with 408555:

```
Router# debug fax relay t30 calling-number 408555
Debugging fax relay t30 from 408555
```

The following command enables debugging for any calls received to a number starting with 555-1212:

```
Router# debug fax relay t30 called-number 4155551212
Debugging fax relay t30 to 4155551212
```

The following command displays all debug entries:

```
Router# debug fax relay t30 all
Debugging fax relay t30 from 408555
Debugging fax relay t30 to 4155551212
```

debug fddi smt-packets

To display information about Station Management (SMT) frames received by the router, use the **debug fddi smt-packets**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug fddi smt-packets

no debug fddi smt-packets

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Examples

The following is sample output from the **debug fddi smt-packets**command. In this example, an SMT frame has been output by FDDI 1/0. The SMT frame is a next station addressing (NSA) neighbor information frame (NIF) request frame with the parameters as shown.

```
Router# debug fddi smt-packets
SMT O: Fddi1/0, FC=NSA, DA=ffff.ffff.ffff, SA=00c0.eeee.be04,
 class=NIF, type=Request, vers=1, station_id=00c0.eeee.be04, len=40
 - code 1, len 8 -- 000000016850043F
 - code 2, len 4 -- 00010200
 - code 3, len 4 -- 00003100
 - code 200B, len 8 -- 0000000100000000
```

The table below describes the significant fields shown in the display.

Table 14: debug fddi smt-packets Field Descriptions

Field	Description
SMT O	SMT frame was sent from FDDI interface 1/0. Also, SMT I indicates that an SMT frame was received on the FDDI interface 1/0.
Fddi1/0	Interface associated with the frame.
FC	Frame control byte in the MAC header.
DA, SA	Destination and source addresses in FDDI form.
class	Frame class. Values can be echo frame (ECF), neighbor information frame (NIF), parameter management frame (PMF), request denied frame (RDF), status information frame (SIF), and status report frame (SRF).
type	Frame type. Values can be Request, Response, and Announce.

Field	Description
vers	Version identification. Values can be 1 or 2.
station_id	Station identification.
len	Packet size.
code 1, len 8 -- 000000016850043F	Parameter type X'0001--upstream neighbor address (UNA), parameter length in bytes, and parameter value. SMT parameters are described in the SMT specification ANSI X3T9.

debug filesystem

To enable ATA ROM monitor library (monlib) debugging messages, use the **debug filesystem**command in privileged EXEC mode. To disable ATA monlib debugging messages, use the **no** form of this command.

debug filesystem {**disk0**| **disk1**}

no debug filesystem {**disk0**| **disk1**}

Syntax Description

disk0	Selects disk 0 as the disk on which to enable or disable debugging.
disk1	Selects disk 1 as the disk on which to enable or disable debugging.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(7)T	This command was introduced.
12.2(25)S	This command was integrated into Cisco IOS Release 12.2(25)S.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

The **debug filesystem** command enables the display of ATA monlib debugging messages during boot operations.

To display the debugging messages when ROMMON accesses the PCMCIA disk, the ROMMON must have disk support. In other words, if a **dev** command is entered in ROMMON mode, the output should display the supported disks as shown in the following example:

```
rommon 1> dev

Devices in device table:
id  name
bootflash:  boot flash
slot0:  PCMCIA slot 0
slot1:  PCMCIA slot 1
disk0:  PCMCIA slot 0
disk1:  PCMCIA slot 1
eprom:  eprom
```

Examples

The following example shows how to enable ATA monlib debugging messages on disk 0, reboot the router to view ATA monlib debugging messages, and then disable ATA monlib debugging messages:

```
Router# debug filesystem disk0

rommon 1> boot disk0:c7200-is-mz.123-5.7.PI3a

Initializing ATA monitor library.......
ATA_read_sector:dev = 0
ATA_data_xfer:1:dev = 0, command = 32, nsecs = 8, sector = 3, cyl_low = 0,
                cyl_high = 0, head = 171
ATA_read_sector:dev = 0, retval = 0
dfs_openfile:Using monlib version 2
dfs_openfile:Using version info 1
dfs_openfile:finding file.. /c7200-is-mz.123-5.7.PI3a
ATA_read_sector:dev = 0
ATA_data_xfer:1:dev = 0, command = 32, nsecs = 1, sector = 15, cyl_low = 0,
                cyl_high = 0, head = 163
ATA_read_sector:dev = 0, retval = 0
ATA_read_sector:dev = 0
ATA_data_xfer:1:dev = 0, command = 32, nsecs = 128, sector = 35, cyl_low = 0,
                cyl_high = 0, head = 171
ATA_read_sector:dev = 0, retval = 0
dfs_openfile:opened file.. /c7200-is-mz.123-5.7.PI3a with fd = 0
DFSLIB_read:reading file.. fd = 0, byte_count = 4
DFSLIB_read:read from.. fd = 0, byte_count = 4, retval = 4
DFSLIB_read:reading file.. fd = 0, byte_count = 52
DFSLIB_read:read from.. fd = 0, byte_count = 52, retval = 52
DFSLIB_read:reading file.. fd = 0, byte_count = 40
DFSLIB_read:read from.. fd = 0, byte_count = 40, retval = 40
.
.
.
DFSLIB_read:reading file.. fd = 0, byte_count = 40
DFSLIB_read:read from.. fd = 0, byte_count = 40, retval = 40
DFSLIB_read:reading file.. fd = 0, byte_count = 19539160
ATA_read_sector:dev = 0
ATA_data_xfer:1:dev = 0, command = 32, nsecs = 1, sector = 15, cyl_low = 0,
                cyl_high = 0, head = 163
ATA_read_sector:dev = 0, retval = 0
ATA_read_sector:dev = 0
.
.
.
ATA_read_sector:dev = 0
ATA_data_xfer:1:dev = 0, command = 32, nsecs = 19, sector = 1, cyl_low = 38,
                cyl_high = 0, head = 169
ATA_read_sector:dev = 0, retval = 0
DFSLIB_read:read from.. fd = 0, byte_count = 19539160, retval = 19539160
Self decompressing the image
:################################################################
################################################################################################################

############################################################ [OK]
Router# no debug filesystem disk0
```

The below table describes the significant fields shown in the display.

Table 15: debug filesystem Field Descriptions

Field	Description
dev =	The number of the device being accessed.
command =	The operation that is being executed.

Field	Description
nsecs =	The number of sectors on the device.
sector =	The starting sector.
cyl_low =, cyl_high =	The starting cylinder, low and high.
head =	The head number.
retval =	The status of the operation being executed.

debug firewall

To enable debugging for events, errors and SCP communication of ASA SM, use the debug firewall command in the global configuration mode. Run the command from RP of Supervisor. To disable the debugging, use the **no** form of the command

debug firewall {**all**| **errors**| **events**| **scp**}

no debug firewall {**all**| **errors**| **events**| **scp**}

Syntax Description

all	Displays the following: The output for all errors in the processes for ASA SM, and the output of all ongoing processes
errors	Displays the output for errors in the processes for ASA SM
events	Displays the output of all ongoing processes for ASA SM
scp	Displays the debug output if SCP messages are dropped between the Supervisor SP or RP, and the linecard. It also displays the debug output for the SCP communication check between the line card and Supervisor.

Command Default

None

Command Modes

Global configuration

Command History

Release	Modification
15.2(4)S2	This command was introduced on the Cisco 7600 series routers

Usage Guidelines

Use the debug command only to troubleshoot specific problems, or during troubleshooting sessions with Cisco technical support staff.

Examples

```
debug firewall all
Router# debug firewall all
Firewall debug errors debugging is on
Firewall debug events debugging is on
Firewall debug scp debugging is on
Router#
```

If the ASA module works as expected, and no configuration is performed on the module, no logs are generated for the command. However, if you configure the firewall VLAN-group with debugging enabled, the output is as follows:

```
May  5 22:45:13.060 PDT: config_firewall_command(): csb->which = 0, csb->nvgen = 1,
csb->sense = 1, obj_1 = 0, obj_2 =, obj_2(int) =2
May  5 22:45:52.853 PDT: config_firewall_command(): csb->which = 1, csb->nvgen = 0,
csb->sense = 1, obj_1 = 900, obj_2 =100, obj_2(int) =0
May  5 22:46:45.189 PDT: config_firewall_command(): csb->which = 0, csb->nvgen = 1,
csb->sense = 1, obj_1 = 0, obj_2 =, obj_2(int) =2
```

debug flow exporter

To enable debugging output for Flexible NetFlow flow exporters, use the **debug flow exporter** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug flow exporter [**[name]** *exporter-name*] **[error] [event]** [**packets** *number*]

no debug flow exporter [**[name]** *exporter-name*] **[error] [event]** [**packets** *number*]

Syntax Description

name	(Optional) Specifies the name of a flow exporter.
exporter-name	(Optional) The name of a flow exporter that was previously configured.
error	(Optional) Enables debugging for flow exporter errors.
event	(Optional) Enables debugging for flow exporter events.
packets	(Optional) Enables packet-level debugging for flow exporters.
number	(Optional) The number of packets to debug for packet-level debugging of flow exporters. Range: 1 to 65535.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.4(9)T	This command was introduced.
12.2(31)SB2	This command was integrated into Cisco IOS Release 12.2(31)SB2.
12.0(33)S	This command was modified. Support for this command was implemented on the Cisco 12000 series routers.
12.2(33)SRC	This command was modified. Support for this command was implemented on the Cisco 7200 series routers.
12.2(33)SRE	This command was modified. Support for this command was implemented on the Cisco 7300 Network Processing Engine (NPE) series routers.
12.2(50)SY	This command was integrated into Cisco IOS Release 12.2(50)SY.

Release	Modification
Cisco IOS XE Release 3.2SE	This command was integrated into Cisco IOS XE Release 3.2SE.

Examples

The following example indicates that a flow exporter packet has been queued for process send:

```
Router# debug flow exporter
May 21 21:29:12.603: FLOW EXP: Packet queued for process send
```

Related Commands

Command	Description
clear flow exporter	Clears the Flexible NetFlow statistics for exporters.

debug flow monitor

To enable debugging output for Flexible NetFlow flow monitors, use the **debug flow monitor** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug flow monitor [error] [**[name]** *monitor-name* **[cache] [error]** [**packets** *packets*]]

no debug flow monitor [error] [**[name]** *monitor-name* **[cache] [error]** [**packets** *packets*]]

Syntax Description

error	(Optional) Enables debugging for flow monitor errors.
name	(Optional) Specifies the name of a flow monitor.
monitor-name	(Optional) The name of a flow monitor that was previously configured.
cache	(Optional) Enables debugging for the flow monitor cache.
packets	(Optional) Enables packet-level debugging for flow monitors.
packets	(Optional) The number of packets to debug for packet-level debugging of flow monitors. Range: 1 to 65535.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.4(9)T	This command was introduced.
12.2(31)SB2	This command was integrated into Cisco IOS Release 12.2(31)SB2.
12.0(33)S	This command was modified. Support for this command was implemented on the Cisco 12000 series routers.
12.2(33)SRC	This command was modified. Support for this command was implemented on the Cisco 7200 series routers.
12.2(33)SRE	This command was modified. Support for this command was implemented on the Cisco 7300 Network Processing Engine (NPE) series routers.
12.2(50)SY	This command was integrated into Cisco IOS Release 12.2(50)SY.

Release	Modification
Cisco IOS XE Release 3.2SE	This command was integrated into Cisco IOS XE Release 3.2SE.

Examples

The following example shows that the cache for FLOW-MONITOR-1 was deleted:

```
Router# debug flow monitor FLOW-MONITOR-1 cache
May 21 21:53:02.839: FLOW MON:  'FLOW-MONITOR-1' deleted cache
```

Related Commands

Command	Description
clear flow monitor	Clears the Flexible NetFlow flow monitor.

debug flow record

To enable debugging output for Flexible NetFlow flow records, use the **debug flow record** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug flow record [**[name]** *record-name*| **netflow-original**| **netflow** {**ipv4**| **ipv6**} *record* **[peer]**| **netflow-v5**| **options** {**exporter-statistics**| **interface-table**| **sampler-table**| **vrf-id-name-table**}]

no debug flow record [**[name]** *record-name*| **netflow-original**| **netflow** {**ipv4**| **ipv6**} *record* **[peer]**| **netflow-v5**| **options** {**exporter-statistics**| **interface-table**| **sampler-table**| **vrf-id-name-table**}]

Cisco Catalyst 6500 Switches in Cisco IOS Release 12.2(50)SY

debug flow record [**[name]** *record-name*| **netflow-v5**| **options** {**exporter-statistics**| **interface-table**| **sampler-table**| **vrf-id-name-table**}| **platform-original** {**ipv4**| **ipv6**} *record* [**detailed**| **error**]]

no debug flow record [**[name]** *record-name*| **netflow-v5**| **options** {**exporter-statistics**| **interface-table**| **sampler-table**| **vrf-id-name-table**}| **platform-original** {**ipv4**| **ipv6**} *record* [**detailed**| **error**]]

Cisco IOS XE Release 3.2SE

debug flow record [**[name]** *record-name*| **netflow** {**ipv4**| **ipv6**} *record* **[peer]**| **netflow-v5**| **options sampler-table**]

no debug flow record [**[name]** *record-name*| **netflow** {**ipv4**| **ipv6**} *record* **[peer]**| **netflow-v5**| **options sampler-table**]

Syntax Description

name	(Optional) Specifies the name of a flow record.
record-name	(Optional) Name of a user-defined flow record that was previously configured.
netflow-original	(Optional) Specifies the traditional IPv4 input NetFlow with origin autonomous systems.
netflow {**ipv4** | **ipv6**} *record*	(Optional) Specifies the name of the NetFlow predefined record. See the table below.
peer	(Optional) Includes peer information for the NetFlow predefined records that support the peer keyword. **Note** The **peer** keyword is not supported for every type of NetFlow predefined record. See the table below.
options	(Optional) Includes information on other flow record options.
exporter-statistics	(Optional) Includes information on the flow exporter statistics.

interface-table	(Optional) Includes information on the interface tables.
sampler-table	(Optional) Includes information on the sampler tables.
vrf-id-name-table	(Optional) Includes information on the virtual routing and forwarding (VRF) ID-to-name tables.
platform-original ipv4 *record*	Configures the flow monitor to use one of the predefined IPv4 records.
platform-original ipv6 *record*	Configures the flow monitor to use one of the predefined IPv6 records.
detailed	(Optional) Displays detailed information.
error	(Optional) Displays errors only.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.4(9)T	This command was introduced.
12.2(31)SB2	This command was integrated into Cisco IOS Release 12.2(31)SB2.
12.0(33)S	This command was modified. Support for this command was implemented on the Cisco 12000 series routers.
12.2(33)SRC	This command was modified. Support for this command was implemented on the Cisco 7200 series routers.
12.4(20)T	This command was modified. The **ipv6** keyword was added in Cisco IOS Release 12.4(20)T.
15.0(1)M	This command was modified. The **vrf-id-name-table** keyword was added.
12.2(33)SRE	This command was modified. Support for this command was implemented on the Cisco 7300 Network Processing Engine (NPE) series routers.
12.2(50)SY	This command was integrated into Cisco IOS Release 12.2(50)SY without support for the **netflow-original**, **netflow**, **ipv4**, **netflow**, **ipv6** and **peer** keywords. The **platform-original ipv4** and **platform-originalipv6** keywords were added.

Release	Modification
Cisco IOS XE Release 3.2SE	This command was integrated into Cisco IOS XE Release 3.2SE without the support for the **netflow-original**, **options exporter-statistics**, **options interface-table** and **option vrf-id-name-table** keywords.

Usage Guidelines

The table below describes the keywords and descriptions for the *record* argument.

Table 16: Keywords and Descriptions for the record Argument

Keyword	Description	IPv4 Support	IPv6 Support
as	Autonomous system record.	Yes	Yes
as-tos	Autonomous system and type of service (ToS) record.	Yes	—
bgp-nexthop-tos	BGP next-hop and ToS record.	Yes	—
bgp-nexthop	BGP next-hop record.	—	Yes
destination	Original 12.2(50)SY platform IPv4/IPv6 destination record.	Yes	Yes
destination-prefix	Destination prefix record. **Note** For IPv6, a minimum prefix mask length of 0 bits is assumed.	Yes	Yes
destination-prefix-tos	Destination prefix and ToS record.	Yes	—
destination-source	Original 12.2(50)SY platform IPv4/IPv6 destination-source record.	Yes	Yes
full	Original 12.2(50)SY platform IPv4/IPv6 full record.	Yes	Yes
interface-destination	Original 12.2(50)SY platform IPv4/IPv6 interface-destination record.	Yes	Yes

Keyword	Description	IPv4 Support	IPv6 Support
interface-destination-source	Original 12.2(50)SY platform IPv4/IPv6 interface-destination-source record.	Yes	Yes
interface-full	Original 12.2(50)SY platform IPv4/IPv6 interface-full record.	Yes	Yes
interface-source	Original 12.2(50)SY platform IPv4/IPv6 interface-source only record.	Yes	Yes
original-input	Traditional IPv4 input NetFlow.	Yes	Yes
original-output	Traditional IPv4 output NetFlow.	Yes	Yes
prefix	Source and destination prefixes record. **Note** For IPv6, a minimum prefix mask length of 0 bits is assumed.	Yes	Yes
prefix-port	Prefix port record. **Note** The **peer** keyword is not available for this record.	Yes	—
prefix-tos	Prefix ToS record.	Yes	—
protocol-port	Protocol ports record. **Note** The **peer** keyword is not available for this record.	Yes	Yes
protocol-port-tos	Protocol port and ToS record. **Note** The **peer** keyword is not available for this record.	Yes	—

Keyword	Description	IPv4 Support	IPv6 Support
source	Original 12.2(50)SY platform IPv4/IPv6 source only record.	Yes	Yes
source-prefix	Source autonomous system and prefix record. **Note** For IPv6, a minimum prefix mask length of 0 bits is assumed.	Yes	Yes
source-prefix-tos	Source prefix and ToS record.	Yes	—

Examples

The following example enables debugging for the flow record:

```
Router# debug flow record FLOW-record-1
```

Related Commands

Command	Description
flow record	Create a Flexible NetFlow flow record.

debug flow-sampler

To enable debugging output for NetFlow sampler activity, use the debug flow-sampler command in privileged EXEC mode. To disable debugging output for NetFlow sampler activity, use the no form of this command.

debug flow-sampler {**class-based**| **events**| **ipc**| **match**}

no debug flow-sampler {**class-based**| **events**| **ipc**| **match**}

Syntax Description

class-based	Displays debug messages for class-based NetFlow samplers.
events	Displays debug messages when a NetFlow sampler map is added, deleted, or applied to an interface.
ipc	Displays NetFlow sampler-related debug messages for interprocess communications (IPC) between the route processor and line cards.
match	Displays debug messages when a packet is sampled (is matched with a NetFlow sampler).

Command Default

Debugging output for NetFlow sampler activity is disabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(2)T	This command was introduced.
12.2(18)S	This command was integrated into Cisco IOS Release 12.2(18)S.
12.0(26)S	This command was integrated into Cisco IOS Release 12.0(26)S.
12.3(4)T	The **class-based** keyword was added.
12.2(27)SBC	This command was integrated into Cisco IOS Release 12.2(27)SBC.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

Because debugging output is assigned high priority in the CPU process, you should use debug commands only to troubleshoot specific problems or during troubleshooting sessions with Cisco technical support staff.

Moreover, you should use debug commands during periods of lower network traffic and fewer users. Debugging during these periods reducess the likelihood that increased debug command processing overhead will affect system use.

Examples

The following is sample output from the **debug flow-sampler events** command:

```
Router# debug flow-sampler events
Flow sampler events debugging is on
Router# configure terminal
Router(config# no flow-sampler mysampler2
Router(config)#
5d00h:  Flow: Sampler mysampler2 detached from FastEthernet0/1
5d00h: Flow: Sampler mysampler2 deleted
```

The following is sample output from the **debug flow-sampler match** command:

```
Router# debug flow-sampler match
Flow sampler match debugging is on
Router#
4d23h: Flow: Packet matched sampler mysampler1 on interface FastEthernet0/0
Router#
4d23h: Flow: Packet matched sampler mysampler1 on interface FastEthernet0/0
Router#
4d23h: Flow: Packet matched sampler mysampler1 on interface FastEthernet0/0
Router#
4d23h: Flow: Packet matched sampler mysampler1 on interface FastEthernet0/0
```

The below table describes the significant fields shown in the display.

Table 17: debug flow-sampler Field Descriptions

Field	Description
Sampler	Name of the NetFlow sampler.
id	Unique ID of the NetFlow sampler.
packets matched	Number of packets matched (sampled) for the NetFlow sampler.
mode	NetFlow sampling mode.
sampling interval is	NetFlow sampling interval (in packets).

Related Commands

Command	Description
flow-sampler	Enables a Random Sampled NetFlow sampler.
flow-sampler-map	Defines a Random Sampled NetFlow sampler map.
ip flow-export	Enables the export of NetFlow data to a collector.
mode (flow sampler map)	Specifies a Random Sampled NetFlow sampling mode and sampling rate.

Command	Description
netflow-sampler	Enables a class-based NetFlow sampler.
show flow-sampler	Displays attributes (including mode, sampling rate, and number of sampled packets) of one or all Random Sampled NetFlow samplers.
show ip flow export	Displays the statistics for the NetFlow data export.

debug fm ipv6 pbr

To enable debugging of IPv6 policy-based routing information, use the **debug fm ipv6 pbr** command in privileged EXEC mode. To disable debugging, use the **no** form of this command.

debug fm ipv6 policy [**all**| **events**| **vmrs**]

no debug fm ipv6 policy [**all**| **events**| **vmrs**]

Syntax Description

all	(Optional) Enables all policy-based routing (PBR) debugging information.
events	(Optional) Enables debugging of all IPv6 PBR events.
vmrs	(Optional) Enables debugging of PBR value mask results (VMRs).

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(33)SXI4	This command was introduced.
15.1(1)SY	This command was integrated into Cisco IOS Release 15.1(1)SY.

Usage Guidelines

Do not use the **debug fm ipv6 pbr** command unless you suspect a problem with IPv6 policy-based routing.

Examples

The following example enables debugging of IPv6 PBR information:

```
Device# debug fm ipv6 pbr
```

debug fm private-hosts

To enable debug messages for the Private Hosts feature manager, use the **debug fm private-hosts** command in privileged EXEC mode.

debug fm private-hosts {**all**| **vmr**| **unusual**| **events**}

Syntax Description

all	Enable debug messages for all Private Hosts errors and events.
vmr	Enable debug messages for the Multicast VLAN Registration (MVR) feature.
unusual	Enable debug messages for unexpected Private Hosts behavior.
events	Enable debug messages for Private Hosts events.

Command Default

This command has no default settings.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(33)SRB	This command was introduced.

Examples

The following example shows sample command output:

```
Router# debug fm private-hosts all

fm private-hosts vmr debugging is on
fm private-hosts unusual debugging is on
fm private-hosts events debugging is on
Router#
```

Related Commands

Command	Description
debug private-hosts	Enables debug messages for Private Hosts.

debug fm raguard

To display information about router advertisement (RA) guard debugging activity, use the **debug fm raguard**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug fm raguard [**all**| **events**| **error**| **vmr**]

no debug fm raguard

Syntax Description

all	(Optional) All RA guard debugging information is displayed.
events	(Optional) Information about RA guard debugging events is displayed.
error	(Optional) Information about RA guard debugging errors is displayed.
vmr	(Optional) Information about debugging value mask results (VMRs) is displayed.

Command Default

RA guard debugging information is not displayed.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(33)SXI4	This command was introduced.
12.2(54)SG	This command was modified. Support for Cisco IOS Release 12.2(54)SG was added.

Usage Guidelines

Do not use the **debug fm raguard**command unless you suspect a problem with IPv6 RA guard.

Examples

The following example enables you to view IPv6 RA guard debugging activity:

```
Router# debug fm raguard
```

debug fmsp receive

Note Effective with release 12.3(8)T, the **debug fmsp receive** command is replaced by the **debug fax fmsp** command. See the **debug fax fmsp** command for more information.

To display debugging messages for Fax Media Services Provider (FMSP) receive, use the **debug fmsp receive**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug fmsp receive [**t30**| **t38**]

no debug fmsp receive [**t30**| **t38**]

Syntax Description

t30	(Optional) Specifies the T.30 fax protocol.
t38	(Optional) Specifies the T.38 fax protocol.

Command Default

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.
12.2(8)T	This command was implemented on the Cisco 1751 access routers, Cisco 3725 access routers, and Cisco 3745 access routers.
12.2(13)T	Support for this command was implemented in Cisco 7200 series images.
12.3(8)T	This command was replaced by the **debug fax fmsp** command.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Examples

The following is sample output from the **debug fmsp receive**command:

```
Router# debug fmsp receive
*Oct 16 08:31:33.243: faxmsp_call_setup_request: call id=28
*Oct 16 08:31:33.243: faxmsp_call_setup_request: ramp data dir=ONRAMP, conf dir=DEST
*Oct 16 08:31:33.243: faxmsp_bridge(): cfid=19, srccid=28, dstcid=27
*Oct 16 08:31:33.243: faxmsp_bridge():  ramp data dir=ONRAMP, conf dir=DEST
*Oct 16 08:31:33.243: faxmsp_bridge(): Explicit caps ind. done; will wait for registry caps
 ind
```

```
*Oct 16 08:31:33.243: faxmsp_caps_ind: call id=28, src=27
*Oct 16 08:31:33.243: faxmsp_caps_ack: call id src=27
*Oct 16 08:31:33.279: faxmsp_call_setup_request: call id=29
*Oct 16 08:31:33.279: faxmsp_call_setup_request: ramp data dir=OFFRAMP, conf dir=SRC
*Oct 16 08:31:33.283: faxmsp_bridge(): cfid=20, srccid=29, dstcid=26
*Oct 16 08:31:33.283: faxmsp_bridge():  ramp data dir=OFFRAMP, conf dir=SRC
*Oct 16 08:31:33.283: faxmsp_bridge(): Explicit caps ind. done; will wait for registry caps
 ind
*Oct 16 08:31:33.283: faxmsp_caps_ind: call id=29, src=26
*Oct 16 08:31:33.283: faxmsp_caps_ack: call id src=26
*Oct 16 08:31:33.635: faxmsp_codec_download_done: call id=29
*Oct 16 08:31:33.635: faxmsp_codec_download_done: call id=28
*Oct 16 08:31:33.643: faxmsp_xmit: callid src=26, dst=29
*Oct 16 08:31:33.643: faxmsp_xmit: callid src=27, dst=28
*Oct 16 08:31:33.643: faxmsp_process_rcv_data: call id src=26, dst=29
```

Related Commands

Command	Description
debug fmsp send	Displays debugging messages for FMSP send.

debug fmsp send

Note Effective with release 12.3(8)T, the **debug fmsp send**command is replaced by the **debug fax fmsp** command. See the **debug fax fmsp** command for more information.

To display debugging messages for Fax Media Services Provider (FMSP) send, use the **debug fmsp send** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug fmsp send [**t30**| *t38*]

no debug fmsp send [*t30*| *t38*]

Syntax Description

t30	(Optional) Specifies the T.30 fax protocol.
t38	(Optional) Specifies the T.38 fax protocol.

Command Default

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.
12.2(8)T	This command was implemented on the Cisco 1751 access routers, Cisco 3725 access routers, and Cisco 3745 access routers.
12.2(13)T	This feature was implemented on the Cisco 7200 series routers.
12.3(8)T	This command was replaced by the **debug fax fmsp** command.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.

Examples

The following is sample output from the **debug fmsp send**command:

```
Router# debug fmsp send
Jan  1 05:02:56.782: faxmsp_call_setup_request: call id=21
```

```
Jan  1 05:02:56.782: faxmsp_call_setup_request: ramp data dir=OFFRAMP, conf dir=SRC
Jan  1 05:02:56.782: faxmsp_bridge(): cfid=7, srccid=21, dstcid=20
Jan  1 05:02:56.782: faxmsp_bridge():  ramp data dir=OFFRAMP, conf dir=SRC
Jan  1 05:02:56.782: faxmsp_bridge(): Explicit caps ind. done; will wait for registry caps
 ind
Jan  1 05:02:56.782: faxmsp_caps_ind: call id=21, src=20
Jan  1 05:02:56.782: faxmsp_caps_ack: call id src=20
Jan  1 05:02:57.174: faxmsp_codec_download_done: call id=21
Jan  1 05:02:57.174: faxMsp_tx_buffer callID=21
Jan  1 05:02:57.178: faxMsp_tx_buffer callID=21
Jan  1 05:02:57.178: faxMsp_tx_buffer callID=21
Jan  1 05:02:57.178: faxMsp_tx_buffer callID=21
Jan  1 05:02:57.182: faxmsp_xmit: callid src=20, dst=21
Jan  1 05:02:57.182: faxmsp_process_rcv_data: call id src=20, dst=21
Jan  1 05:03:01.814: faxmsp_xmit: callid src=20, dst=21
Jan  1 05:03:01.814: faxmsp_process_rcv_data: call id src=20, dst=21
Jan  1 05:03:01.814: faxMsp_tx_buffer callID=21
Jan  1 05:03:02.802: faxmsp_xmit: callid src=20, dst=21
Jan  1 05:03:02.802: faxmsp_process_rcv_data: call id src=20, dst=21
Jan  1 05:03:02.822: faxmsp_xmit: callid src=20, dst=21
Jan  1 05:03:02.822: faxmsp_process_rcv_data: call id src=20, dst=21
Jan  1 05:03:02.854: faxmsp_xmit: callid src=20, dst=21
Jan  1 05:03:02.854: faxmsp_process_rcv_data: call id src=20, dst=21
```

Related Commands

Command	Description
debug fax relay t30	Displays debugging messages for FMSP receive.

debug foip off-ramp

Note Effective with release 12.3(8)T, the **debug foip off-ramp**command is replaced by the **debug fax foip**command. See the **debug fax foip**command for more information.

To display debugging messages for off-ramp fax mail, use the **debug foip off-ramp**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug foip off-ramp

no debug foip off-ramp

Syntax Description

This command has no arguments or keywords.

Command Default

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.
12.2(8)T	This command was introduced on the Cisco 1751 access routers, Cisco 3725 access routers, and Cisco 3745 access routers.
12.2(13)T	This feature was implemented on the Cisco 7200 series routers.
12.3(8)T	This command was replaced by the **debug fax foip** command.

Examples

The following is sample output from the **debug foip off-ramp**command:

```
Router# debug foip off-ramp
Jan  1 02:31:17.539: lapp off: CC_EV_CALL_HANDOFF, cid(0xB)
Jan  1 02:31:17.539: loffHandoff: called number=5271714, callid=0xB
Jan  1 02:31:17.543: loffSetupPeer: cid1(0xB)
Jan  1 02:31:17.543:  destPat(5271714),matched(1),pref(5),tag(20),encap(1)
Jan  1 02:31:22.867: lapp off: CC_EV_CALL_CONNECTED, cid(0xC)
Jan  1 02:31:22.867:  st=CALL_SETTING cid(0xB,0x0,0x0,0xC),cfid(0x0,0x0,0x0)
Jan  1 02:31:22.867: loffConnected
Jan  1 02:31:22.867: loffFlushPeerTagQueue cid(11) peer list: (empty)
Jan  1 02:31:22.867: lapp off: CC_EV_CONF_CREATE_DONE, cid(0xC), cid2(0xD), cfid(0x1)
Jan  1 02:31:22.867:  st=CONFERENCING3 cid(0xB,0x0,0xD,0xC),cfid(0x0,0x0,0x1)
Jan  1 02:31:22.867: loffConfDone3
Jan  1 02:31:30.931: lapp off: CC_EV_FROM_FMSP_ON_CALL_DETAIL, cid(0xD)
Jan  1 02:31:30.931:  st=WAIT_SESS_INFO cid(0xB,0x0,0xD,0xC),cfid(0x0,0x0,0x1)
Jan  1 02:31:30.931: loffSessionInfo
Jan  1 02:31:30.931:  encd=2, resl=2, spd=26, min_scan_len=0, csid=          4085271714
```

```
Jan  1 02:31:30.931: lapp off: CC_EV_CONF_CREATE_DONE, cid(0xD), cid2(0xE), cfid(0x2)
Jan  1 02:31:30.931:  st=CONFERENCING2 cid(0xB,0xE,0xD,0xC),cfid(0x0,0x2,0x1)
Jan  1 02:31:30.931: loffConfDone2
```

Related Commands

Command	Description
debug foip on-ramp	Displays debugging messages for on-ramp fax mail.

debug foip on-ramp

Note Effective with release 12.3(8)T, the **debug foip on-ramp**command is replaced by the **debug fax foip**command. See the **debug fax foip**command for more information.

To display debugging messages for on-ramp fax mail, use the **debug foip on-ramp**commandinprivileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug foip on-ramp

no debug foip on-ramp

Syntax Description

This command has no arguments or keywords.

Command Default

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.
12.2(8)T	This command was introduced on the Cisco 1751 access routers, Cisco 3725 access routers, and Cisco 3745 access routers.
12.2(13)T	This feature was implemented on the Cisco 7200 series routers.
12.3(8)T	This command was replaced by the **debug fax foip**command.

Examples

The following is sample output from the **debug foip on-ramp**command:

```
Router# debug foip on-ramp
*Oct 16 08:07:01.947: lapp_on_application: Incoming Event: (15 = CC_EV_CALL_HANDOFF),
CID(11), DISP(0)
*Oct 16 08:07:01.947: lapp_on_call_handoff: Authentication enabled = FALSE
*Oct 16 08:07:01.947: lapp_on_call_handoff: Authentication ID = 0
*Oct 16 08:07:01.947: lapp_on_call_handoff: Authentication ID source = IVR or unknown
*Oct 16 08:07:01.947: lapp_on_call_handoff: Authentication status = SUCCESS
*Oct 16 08:07:01.947: lapp_on_call_handoff: Accounting enabled  = FALSE
*Oct 16 08:07:01.947: lapp_on_call_handoff: Accounting method list  = fax
*Oct 16 08:07:01.947: lapp_on_conference_vtsp_fmsp: Begin conferencing VTSP and FMSP...
*Oct 16 08:07:01.951: lapp_on_change_state: old state(0) new state(1)
*Oct 16 08:07:01.951: lapp_on_application: Incoming Event: (29 = CC_EV_CONF_CREATE_DONE),
CID(11), DISP(0)
*Oct 16 08:07:01.951: lapp_on_application: Current call state = 1
*Oct 16 08:07:01.951: lapp_on_conference_created: The VTSP and the FMSP are conferenced
*Oct 16 08:07:01.951: lapp_on_conference_created: Wait for FMSP call detail event
```

Related Commands

Command	Description
debug foip off-ramp	Displays debugging messages for off-ramp fax mail.

debug format

To verify the syntax of eXtensible Markup Language Programmatic Interface (XML-PI) spec files, use the **debug format**command in privileged EXEC mode. To disable debugging, use the **no** form of this command.

debug format {**all**| **error**}

no debug format {**all**| **error**}

Syntax Description

all	Specifies verbose mode to display selected debug data with comments followed by full debug output.
error	Displays minimal format error statements.

Command Default

This command is disabled by default.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.4(20)T	This command was introduced.
12.2(33)SRE	This command was modified. It was integrated into Cisco IOS Release 12.2(33)SRE.
12.2(50)SY	This command was integrated into Cisco IOS Release 12.2(50)SY.

Usage Guidelines

Use the **debug format all**command to troubleshoot errors in XML-PI spec files. The command displays XML output, the XML Schema Definition (XSD), and parsing locations. For less verbose output, use the **debug format error**command.

Examples

The following examples show how to use the verbose output from the **debug format all**command to troubleshoot spec file entries based on information collected from the **show interfaces** command.

Begin by displaying the **show interfaces** command output.

```
Router# show interfaces
FastEthernet0/0 is up, line protocol is up
  Hardware is i82543 (Livengood), address is 000b.60dc.9408 (bia 000b.60dc.9408)
  Internet address is 10.4.4.5/8
  MTU 1500 bytes, BW 100000 Kbit, DLY 100 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation ARPA, loopback not set
  Keepalive set (10 sec)
```

```
  Full-duplex, 100Mb/s, 100BaseTX/FX
  ARP type: ARPA, ARP Timeout 04:00:00
  Last input 00:00:07, output 00:00:03, output hang never
  Last clearing of "show interface" counters never
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 0
  Queueing strategy: fifo
  Output queue: 0/40 (size/max)
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     769 packets input, 121369 bytes
     Received 696 broadcasts, 0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored
     0 watchdog
     0 input packets with dribble condition detected
     959 packets output, 94185 bytes, 0 underruns
     2 output errors, 0 collisions, 1 interface resets
     0 babbles, 0 late collision, 0 deferred
     2 lost carrier, 0 no carrier
     0 output buffer failures, 0 output buffers swapped out
FastEthernet0/1 is down, line protocol is down
  Hardware is i82543 (Livengood), address is 000b.60dc.9406 (bia 000b.60dc.9406)
  MTU 1500 bytes, BW 100000 Kbit, DLY 100 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation ARPA, loopback not set
  Keepalive set (10 sec)
  Unknown duplex, Unknown Speed, 100BaseTX/FX
  ARP type: ARPA, ARP Timeout 04:00:00
  Last input never, output never, output hang never
  Last clearing of "show interface" counters never
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 0
  Queueing strategy: fifo
  Output queue: 0/40 (size/max)
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     0 packets input, 0 bytes
     Received 0 broadcasts, 0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored
     0 watchdog
     0 input packets with dribble condition detected
     0 packets output, 0 bytes, 0 underruns
     2 output errors, 0 collisions, 1 interface resets
     0 babbles, 0 late collision, 0 deferred
     2 lost carrier, 0 no carrier
     0 output buffer failures, 0 output buffers swapped out
Loopback0 is up, line protocol is up
  Hardware is Loopback
  MTU 1514 bytes, BW 8000000 Kbit, DLY 5000 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation LOOPBACK, loopback not set
  Last input never, output never, output hang never
  Last clearing of "show interface" counters never
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 0
  Queueing strategy: fifo
  Output queue: 0/0 (size/max)
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     0 packets input, 0 bytes, 0 no buffer
     Received 0 broadcasts, 0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     0 packets output, 0 bytes, 0 underruns
     0 output errors, 0 collisions, 0 interface resets
     0 output buffer failures, 0 output buffers swapped out
```

The next step is to generate XML output for the **show interfaces**command:

```
Router# show interfaces | format slot0:spec3.3.odm
<?xml version="1.0" encoding="UTF-8"?>
  <ShowInterfaces>
    <Interface>
      <Name>FastEthernet0/0</Name>
      <State>UP</State>
      <lineprotocol>up</lineprotocol>
```

```
<Hardware>i82543</Hardware>
<address>000b.60dc.9408</address>
<bia>000b.60dc.9408</bia>
<Internetaddress>10.4.4.5/8</Internetaddress>
<MTU>1500</MTU>
<BW>100000</BW>
<DLY>100</DLY>
<reliability>255/255,</reliability>
<txload>1/255,</txload>
<rxload>1/255</rxload>
<Encapsulation>ARPA,</Encapsulation>
<loopback>Notset</loopback>
<Keepalive>Set</Keepalive>
<KeepAliveInterval>10</KeepAliveInterval>
<TunnelInfo>
</TunnelInfo>
<DuplexInfo>
  <Duplextype>duplex,</Duplextype>
  <DuplexSpeed>100Mb/s,</DuplexSpeed>
</DuplexInfo>
<ARPtype>ARPA,</ARPtype>
<ARPTimeout>04:00:00</ARPTimeout>
<LMIInfo>
</LMIInfo>
<BroadcastInfo>
</BroadcastInfo>
<Lastinput>00:00:26,</Lastinput>
<output>00:00:08,</output>
<outputhang>never</outputhang>
<LastclearingOfCounters>never</LastclearingOfCounters>
<Inputqueue>
  <Size>0/75/0/0</Size>
  <Max>0/75/0/0</Max>
  <Drops>0/75/0/0</Drops>
  <flushes>0/75/0/0</flushes>
</Inputqueue>
<Totaloutputdrops>0</Totaloutputdrops>
<Queueingstrategy>fifo</Queueingstrategy>
<Outputqueue>
  <Size>0/40</Size>
  <Max>0/40</Max>
</Outputqueue>
<Conversations>
</Conversations>
<ReservedConversations>
</ReservedConversations>
<Fiveminuteinputrateinbits>0</Fiveminuteinputrateinbits>
<Fiveminuteinputrateinpkts>0</Fiveminuteinputrateinpkts>
<Fiveminuteoutputrateinbits>0</Fiveminuteoutputrateinbits>
<Fiveminuteoutputrateinpkts>0</Fiveminuteoutputrateinpkts>
<L2Switched>
</L2Switched>
<L3in>
</L3in>
<L3out>
</L3out>
<packetsinput>771</packetsinput>
<nobuffer>0</nobuffer>
<broadcasts>0</broadcasts>
<runts>0</runts>
<giants>0</giants>
<throttles>0</throttles>
<inputerrors>0</inputerrors>
<CRC>0</CRC>
<frame>0</frame>
<overrun>0</overrun>
<ignored>0</ignored>
<packetsoutput>0</packetsoutput>
<underruns>0</underruns>
<outputerrors>0</outputerrors>
<collisions>0</collisions>
<interfaceresets>0</interfaceresets>
<outputbufferfailures>0</outputbufferfailures>
```

```
      <outputbuffersswappedout>0</outputbuffersswappedout>
    </Interface>
  </ShowInterfaces>
```

Analyze the two outputs: In this case, output about only one interface is listed in the **show interfaces | format slot0:spec3.3.odm** command, but based on the original **show interfaces** command output, it was expected that there would be output about three interfaces.

Enter the following commands to enable the verbose debugging mode that displays all Operational Data Model (ODM) errors:

```
Router# debug format all
Router# show interfaces | format slot0:spec3.3.odm
```

The debug format statements are read in groups of two lines. As the following example shows, the first line describes what the attempted match was; the second line provides the offset and the byte count from the beginning of the **show interfaces**command output that the cursor of the screen scraper has reached.

```
*May  4 01:20:35.279: ODM: Could not match Property mcast
*May  4 01:20:35.279: offset 703: 5 minute output rate 0 bits/sec, 0 packets/sec
```

The following example shows where the spec file entry (SFE) caused the ODM algorithm to return a truncated XML. Notice how the offset jumps from 703 to 3001. This is a large jump that implies a search between multicast and IP multicast probably caused the screen scraper to jump too far into the text. Because the cursor is not at a buffer, this condition is the likely candidate for the error. Looking at the spec file entry and doing a manual search through the **show** command output will confirm this suspicion.

```
*May  4 01:20:35.279: offset 703: 5 minute output rate 0 bits/sec, 0 packets/sec
     786 pa
*May  4 01:20:35.279: ODM: Could not match Property mcast
*May  4 01:20:35.279: offset 703: 5 minute output rate 0 bits/sec, 0 packets/sec
     786 pa
*May  4 01:20:35.279: ODM: Could not match Property IP multicasts
*May  4 01:20:35.279: offset 3001: no buffer
     Received 0 broadcasts, 0 runts, 0 giants, 0
*May  4 01:20:35.279: ODM: Could not match Property watchdog
*May  4 01:20:35.279: offset 3122: ignored, 0 abort
     0 packets output, 0 bytes, 0 underru
*May  4 01:20:35.279: ODM: Could not match Property input packets with dribble condition
detected
Be sure to disable the debug
 command.
```

Related Commands

Command	Description
show interfaces	Displays statistics for all interfaces configured on the router or access server

debug fpm event

Note Effective with Cisco IOS Release 15.2(4)M, the **debug fpm event** command is not available in Cisco IOS software.

To display protocol information from the designated protocol header description field (PHDF), use the **debug fpm event**command in privileged EXEC mode. To disable debugging messages, use the **no** form of this command.

debug fpm event

no debug fpm event

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.4(4)T	This command was introduced.
12.2(18)ZY	This command was integrated into Cisco IOS Release 12.2(18)ZY on the Catalyst 6500 series of switches equipped with the Programmable Intelligent Services Accelerator (PISA).
15.2(4)M	This command was removed from the Cisco IOS software.

Examples

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

```
Router# debug fpm event
*Jun 21 09:22:21.607: policy-classification-inline(): matches class: class-default *Jun 21
 09:22:21.607: packet-access-control(): policy-map: fpm-policy, dir: input, match. retval:
 0x0, ip-flags: 0x80000000
```

debug frame-relay

To display debugging information about the packets received on a Frame Relay interface, use the **debug frame-relay** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug frame-relay

no debug frame-relay

Syntax Description

This command has no arguments or keywords.

Command Default

This command is disabled by default.

Command Modes

Privileged EXEC

Command History

Release	Modification
9.00	This command was introduced.
12.2(13)T	Support for Banyan VINES was removed.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

This command helps you analyze the packets that have been received. However, because the **debug frame-relay** command generates a substantial amount of output, use it only when the rate of traffic on the Frame Relay network is less than 25 packets per second.

To analyze the packets that have been *sent* on a Frame Relay interface, use the **d ebug frame-relay packet** command.

Examples

The following is sample output from the **debug frame-relay** command:

```
Router# debug frame-relay
Serial0(i): dlci 500(0x7C41), pkt type 0x809B, datagramsize 24
Serial1(i): dlci 1023(0xFCF1), pkt type 0x309, datagramsize 13
Serial0(i): dlci 500(0x7C41), pkt type 0x809B, datagramsize 24
Serial1(i): dlci 1023(0xFCF1), pkt type 0x309, datagramsize 13
Serial0(i): dlci 500(0x7C41), pkt type 0x809B, datagramsize 24
```

The below table describes the significant fields shown in the display.

Table 18: debug frame-relay Field Descriptions

Field	Description
Serial0(i):	Indicates that serial interface 0 has received this Frame Relay datagram as input.
dlci 500(0x7C41)	Indicates the value of the data-link connection identifier (DLCI) for this packet in decimal (and q922). In this case, 500 has been configured as the multicast DLCI.

Field	Description
pkt type 0x809B	

Field	Description
	Indicates the packet type code. Possible supported signalling message codes are as follows: • 0x308--Signalling message; valid only with a DLCI of 0 • 0x309--LMI message; valid only with a DLCI of 1023 Possible supported Ethernet type codes are: • 0x0201--IP on a 3 MB net • 0x0201--Xerox ARP on 10 MB networks • 0xCC--RFC 1294 (only for IP) • 0x0600--XNS • 0x0800--IP on a 10 MB network • 0x0806--IP ARP • 0x0808--Frame Relay Address Resolution Protocol (ARP) Possible High-Level Data Link Control (HDLC) type codes are as follows: • 0x6001--DEC Maintenance Operation Protocol (MOP) booting protocol • 0x6002--DEC MOP console protocol • 0x6003--DECnet Phase IV on Ethernet • 0x6004--DEC LAT on Ethernet • 0x8005--HP Probe • 0x8035--RARP • 0x8038--DEC spanning tree • 0x809b--Apple EtherTalk • 0x80f3--AppleTalk ARP • 0x8019--Apollo domain • 0x8137--Internetwork Packet Exchange (IPX) • 0x9000--Ethernet loopback packet IP • 0x1A58--IPX, standard form • 0xFEFE--Connectionless Network Service (CLNS)

Field	Description
	• 0xEFEF--End System-to-Intermediate System (ES-IS) • 0x1998--Uncompressed TCP • 0x1999--Compressed TCP • 0x6558--Serial line bridging
datagramsize 24	Indicates size of this datagram (in bytes).

debug frame-relay adjacency

To display information pertaining to an adjacent node that has one or more Frame Relay permanent virtual circuit (PVC) bundles, use the **debug frame-relay adjacency** command in privileged EXEC mode. To stop displaying the adjacent node information, use the **no** form of this command.

debug frame-relay adjacency {**pvc** [*dlci*]| **vc-bundle** [*vc-bundle-name*]}

no debug frame-relay adjacency {**pvc** [*dlci*]| **vc-bundle** [*vc-bundle-name*]}

Syntax Description

pvc	Displays information regarding the adjacent PVC only.
dlci	(Optional) Data-link connection identifier for a specific PVC.
vc-bundle	Displays information regarding the adjacent PVC bundle and its members.
vc-bundle-name	(Optional) Name of the PVC bundle.

Command Default

No default behaviors or values.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(13)T	This command was introduced.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.

Usage Guidelines

Use this command to monitor adjacency activity and status for an adjacent node.

Note Debug messages that are prefixed with “FR_ADJ” (instead of “FR-ADJ”) indicate serious failures in the Frame Relay PVC bundle performance. Contact the Cisco Technical Assistance Center (TAC) if you see debugging messages with this prefix.

Examples

The following sample output from the **debug frame-relay adjacency vc-bundle** command shows PVC bundle "MP-4-dynamic" going down. Each bundle member PVC is marked for removal from the CEF adjacency table, and then the adjacency for the PVC bundle itself is marked for removal. The adjacencies are actually removed from the table later when a background clean-up process runs.

```
Router# debug frame-relay adjacency vc-bundle MP-4-dynamic
00:46:35: FR-ADJ: vcb MP-4-dynamic: ip 10.2.2.2: member 400: removing adj
00:46:35: FR-ADJ: vcb MP-4-dynamic: ip 10.2.2.2: member 401: removing adj
00:46:35: FR-ADJ: vcb MP-4-dynamic: ip 10.2.2.2: member 402: removing adj
00:46:35: FR-ADJ: vcb MP-4-dynamic: ip 10.2.2.2: member 403: removing adj
00:46:35: FR-ADJ: vcb MP-4-dynamic: ip 10.2.2.2: member 404: removing adj
00:46:35: FR-ADJ: vcb MP-4-dynamic: ip 10.2.2.2: member 405: removing adj
00:46:35: FR-ADJ: vcb MP-4-dynamic: ip 10.2.2.2: member 406: removing adj
00:46:35: FR-ADJ: vcb MP-4-dynamic: ip 10.2.2.2: member 407: removing adj
00:46:35: FR-ADJ: vcb MP-4-dynamic: ip 10.2.2.2: removing primary adj
```

Related Commands

Command	Description
debug frame-relay vc-bundle	Displays information pertaining to all the PVC bundles configured on the router.

debug frame-relay callcontrol

To display Frame Relay Layer 3 (network layer) call control information, use the **debug frame-relay callcontrol**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug frame-relay callcontrol

no debug frame-relay callcontrol

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

The **debug frame-relay callcontrol**command is used specifically for observing FRF.4/Q.933 signalling messages and related state changes. The FRF.4/Q.933 specification describes a state machine for call control. The signalling code implements the state machine. The debug statements display the actual event and state combinations.

The Frame Relay switched virtual circuit (SVC) signalling subsystem is an independent software module. When used with the **debug frame-relay networklayerinterfac e** command, the **debug frame-relay callcontrol** command provides a better understanding of the call setup and teardown sequence. The **debug frame-relay networklayerinterface** command provides the details of the interactions between the signalling subsystem on the router and the Frame Relay subsystem.

Examples

State changes can be observed during a call setup on the calling party side. The **debug frame-relay networklayerinterface** command shows the following state changes or transitions:

```
STATE_NULL -> STATE_CALL_INITIATED -> STATE_CALL_PROCEEDING->STATE_ACTIVE
```

The following messages are samples of output generated during a call setup on the calling side:

```
6d20h: U0_SetupRequest: Serial0
6d20h: L3SDL: Ref: 1, Init: STATE_NULL, Rcvd: SETUP_REQUEST, Next: STATE_CALL_INITIATED
6d20h: U1_CallProceeding: Serial0
6d20h: L3SDL: Ref: 1, Init: STATE_CALL_INITIATED, Rcvd: MSG_CALL_PROCEEDING, Next:
     STATE_CALL_PROCEEDING
6d20h: U3_Connect: Serial0
6d20h: L3SDL: Ref: 1, Init: STATE_CALL_PROCEEDING, Rcvd: MSG_CONNECT, Next: STATE_ACTIVE
6d20h:
```

The following messages are samples of output generated during a call setup on the called party side. Note the state transitions as the call goes to the active state:

```
STATE_NULL -> STATE_CALL_PRESENT-> STATE_INCOMING_CALL_PROCEEDING->STATE_ACTIVE
1w4d: U0_Setup: Serial2/3
1w4d: L3SDL: Ref: 32769, Init: STATE_NULL, Rcvd: MSG_SETUP, Next: STATE_CALL_PRESENT 1w4d:
 L3SDL: Ref: 32769, Init: STATE_CALL_PRESENT, Rcvd: MSG_SETUP, Next:
     STATE_INCOMING_CALL_PROC 1w4d: L3SDL: Ref: 32769, Init: STATE_INCOMING_CALL_PROC,
     Rcvd: MSG_SETUP, Next: STATE_ACTIVE
```

The below table explains the possible call states.

Table 19: Frame Relay Switched Virtual Circuit Call States

Call State	Description
Null	No call exists.
Call Initiated	User has requested the network to establish a call.
Outgoing Call Proceeding	User has received confirmation from the network that the network has received all call information necessary to establish the call.
Call Present	User has received a request to establish a call but has not yet responded.
Incoming Call Proceeding	User has sent acknowledgment that all call information necessary to establish the call has been received (for an incoming call).
Active	On the called side, the network has indicated that the calling user has been awarded the call. On the calling side, the remote user has answered the call.
Disconnect Request	User has requested that the network clear the end-to-end call and is waiting for a response.
Disconnect Indication	User has received an invitation to disconnect the call because the network has disconnected the call.
Release Request	User has requested that the network release the call and is waiting for a response.

Related Commands

Command	Description
debug fax relay t30	Displays debugging information about the packets that are received on a Frame Relay interface.
debug frame-relay networklayerinterface	Displays NLI information.

debug frame-relay end-to-end keepalive

To display debug messages for the Frame Relay End-to-End Keepalive feature, use the debug frame-relay end-to-end keepalive command. Use the no form of this command to disable the display of debug messages.

debug frame-relay end-to-end keepalive {**events**| **packet**}

no debug frame-relay end-to-end keepalive {**events**| **packet**}

Syntax Description

events	Displays keepalive events.
packet	Displays keepalive packets sent and received.

Command History

Release	Modification
12.0(5)T	This command was introduced.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

We recommend that both commands be enabled.

Examples

The following examples show typical output from the **debug frame-relay end-to-end keepalive packet** command. The following example shows output for an outgoing request packet:

```
EEK (o, Serial0.1 DLCI 200): 1 1 1 3 2 4 3
```

The seven number fields that follow the colon signify the following:

Field	Description
first (example value = 1)	Information Element (IE) type.
second (example value = 1)	IE length.
third (example value = 1)	Report ID. 1 = request, 2 = reply.
fourth (example value = 3)	Next IE type. 3 = LIV ID (Keepalive ID).
fifth (example value = 2)	IE length. (This IE is a Keepalive IE.)
sixth (example value = 4)	Send sequence number.
seventh (example value = 3)	Receive sequence number.

The following example shows output for an incoming reply packet:

```
EEK (i, Serial0.1 DLCI 200): 1 1 2 3 2 4 4
```

The seven number fields that follow the colon signify the following:

Field	Description
first (example value = 1)	Information Element (IE) type.
second (example value = 1)	IE length.
third (example value = 2)	Report ID. 1 = request, 2 = reply.
fourth (example value = 3)	Next IE type. 3 = LIV ID (Keepalive ID).
fifth (example value = 2)	IE length. (This IE is a Keepalive IE.)
sixth (example value = 4)	Send sequence number.
seventh (example value = 4)	Receive sequence number.

The following example shows typical output from the **debug frame-relay end-to-end keepalive events** command:

```
EEK SUCCESS (request, Serial0.2 DLCI 400)
EEK SUCCESS (reply, Serial0.1 DLCI 200)
EEK sender timeout (Serial0.1 DLCI 200)
```

debug frame-relay events

To display debugging information about Frame Relay Address Resolution Protocol (ARP) replies on networks that support a multicast channel and use dynamic addressing, use the **debug frame-relay events command in**privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug frame-relay events

no debug frame-relay events

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
11.3	This command was introduced.
12.0(23)S	This command was integrated into Cisco IOS Release 12.0(23)S for the Frame Relay over MPLS feature.
12.2(14)S	This command was integrated into Cisco IOS Release 12.2(14)S.
12.2(15)T	This command was integrated into Cisco IOS Release 12.2(15)T.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

This command is useful for identifying the cause of end-to-end connection problems during the installation of a Frame Relay network or node.

Note Because the **debug frame-relay events** command does not generate much output, you can use it at any time, even during periods of heavy traffic, without adversely affecting other users on the system.

Examples

The following is sample output from the **debug frame-relay events** command:

```
Router# debug frame-relay events
Serial2(i): reply rcvd 172.16.170.26 126
Serial2(i): reply rcvd 172.16.170.28 128
Serial2(i): reply rcvd 172.16.170.34 134
Serial2(i): reply rcvd 172.16.170.38 144
Serial2(i): reply rcvd 172.16.170.41 228
Serial2(i): reply rcvd 172.16.170.65 325
```

As the output shows,the **debug frame-relay events** command returns one specific message type. The first line, for example, indicates that IP address 172.16.170.26 sent a Frame Relay ARP reply; this packet was

received as input on serial interface 2. The last field (126) is the data-link connection identifier (DLCI) to use when communicating with the responding router.

For Frame Relay over MPLS, the following is sample output for the **debug frame-relay events** command. The command output shows the status of the VCs.

```
Router# debug frame-relay events

Frame Relay events debugging is on
```

This example shows the messages that are displayed when you shut the core-facing interface on a provider edge (PE) router:

```
04:40:38:%SYS-5-CONFIG_I: Configured from console by consolenf t
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)# interface hssi2/0
Router(config-if)# shut
04:40:43:%OSPF-5-ADJCHG: Process 10, Nbr 12.12.12.12 on Hssi2/0 from FULL to DOWN, Neighbor
 Down: Interface down or detached
04:40:43: FRoMPLS [12.12.12.12, 100]: PW pvc_status set INACTIVE
04:40:43: FRoMPLS [12.12.12.12, 100]: Setting pw segment DOWN
04:40:43: FRoMPLS [12.12.12.12, 100]: Setting connection DOWN
04:40:43: FRoMPLS [12.12.12.12, 101]: PW pvc_status set INACTIVE
04:40:43: FRoMPLS [12.12.12.12, 101]: Setting pw segment DOWN
04:40:43: FRoMPLS [12.12.12.12, 101]: Setting connection DOWN
04:40:45:%LINK-5-CHANGED: Interface Hssi2/0, changed state to administratively down
04:40:46:%LINEPROTO-5-UPDOWN: Line protocol on Interface Hssi2/0, changed state to down
```

This example shows the messages that are displayed when you enable the core-facing interface on a PE router:

```
Router(config-if)# no shut
04:40:56:%LINK-3-UPDOWN: Interface Hssi2/0, changed state to up
04:40:57:%LINEPROTO-5-UPDOWN: Line protocol on Interface Hssi2/0, changed state to up
04:41:06:%OSPF-5-ADJCHG: Process 10, Nbr 12.12.12.12 on Hssi2/0 from LOADING to FULL, Loading
 Done
04:41:19: FRoMPLS [12.12.12.12, 100]: PW pvc_status set ACTIVE
04:41:19: FRoMPLS [12.12.12.12, 100]: Setting pw segment UP
04:41:19: FRoMPLS [12.12.12.12, 101]: PW pvc_status set ACTIVE
04:41:19: FRoMPLS [12.12.12.12, 101]: Setting pw segment UP
```

This example shows the messages that are displayed when you shut the edge-facing interface on a PE router:

```
Router(config)# interface pos4/0
Router(config-if)# shut
04:42:50: FRoMPLS [12.12.12.12, 100]: acmgr_circuit_down
04:42:50: FRoMPLS [12.12.12.12, 100]: Setting connection DOWN
04:42:50: FRoMPLS [12.12.12.12, 100]: PW pvc_status set INACTIVE
04:42:52:%LINK-5-CHANGED: Interface POS4/0, changed state to administratively down
04:42:53:%LINEPROTO-5-UPDOWN: Line protocol on Interface POS4/0, changed state to down
```

This example shows the messages that are displayed when you enable the edge-facing interface on a PE router:

```
Router(config)# interface pos4/0
Router(config-if)# no shut
04:43:20:%LINK-3-UPDOWN: Interface POS4/0, changed state to up
c72-33-2(config-if)#
04:43:20: FRoMPLS [12.12.12.12, 100]: Local up, sending acmgr_circuit_up
04:43:20: FRoMPLS [12.12.12.12, 100]: PW nni_pvc_status set ACTIVE
04:43:20: FRoMPLS [12.12.12.12, 100]: PW pvc_status set ACTIVE
04:43:20: FRoMPLS [12.12.12.12, 100]: Setting pw segment UP
```

debug frame-relay foresight

To observe Frame Relay traces relating to traffic shaping with router ForeSight enabled, use the **debug frame-relay foresight**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug frame-relay foresight

no debug frame-relay foresight

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Examples

The following is sample output that shows the display message returned in response to the **debug frame-relay foresight** command:

```
Router# debug frame-relay foresight
FR rate control for DLCI 17 due to ForeSight msg
```

This message indicates the router learned from the ForeSight message that data-link connection identifier (DLCI) 17 is now experiencing congestion. The output rate for this circuit should be slowed down, and in the router this DLCI is configured to adapt traffic shaping in response to foresight messages.

Related Commands

Command	Description
show frame-relay pvc	Displays statistics about PVCs for Frame Relay interfaces.

debug frame-relay fragment

To display information related to Frame Relay fragmentation on a permanent virtual circuit (PVC), use the **debug frame-relay fragment** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug frame-relay fragment [**event**| **interface** *type number dlci*]

no debug frame-relay fragment [**event**| **interface** *type number dlci*]

Syntax Description

event	(Optional) Displays event or error messages related to Frame Relay fragmentation.
interface	(Optional) Displays fragments received or sent on the specified interface.
type	(Optional) The interface type for which you wish to display fragments received or sent.
number	(Optional) The Interface number.
dlci	(Optional) The data-link connection identifier (DLCI) value of the PVC for which you wish to display fragments received or sent.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(3)XG	This command was introduced.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

This command will display event or error messages related to Frame Relay fragmentation; it is only enabled at the PVC level on the selected interface.

This command is not supported on the Cisco MC3810 networking device for fragments received by a PVC configured via the **voice-encap** command.

Examples

The following is sample output from the **debug frame-relay fragment** command:

```
Router# debug frame-relay fragment interface serial 0/0 109
This may severely impact network performance.
```

```
You are advised to enable 'no logging console debug'. Continue?[confirm]
Frame Relay fragment/packet debugging is on
Displaying fragments/packets on interface Serial0/0  dlci 109 only
Serial0/0(i): dlci 109, rx-seq-num 126, exp_seq-num 126, BE bits set, frag_hdr 04 C0 7E
Serial0/0(o): dlci 109, tx-seq-num 82, BE bits set, frag_hdr 04 C0 52
```

The following is sample output from the **debug frame-relay fragment event** command:

```
Router# debug frame-relay fragment event

This may severely impact network performance.
You are advised to enable 'no logging console debug'. Continue?[confirm]
Frame Relay fragment event/errors debugging is on
Frame-relay reassembled packet is greater than MTU size, packet dropped on serial 0/0
      dlci 109
Unexpected B bit  frame rx on serial0/0 dlci 109, dropping pending segments
Rx an out-of-sequence packet on serial 0/0 dlci 109, seq_num_received 17
      seq_num_expected 19
```

Related Commands

Command	Description
debug ccfrf11 session	Displays the ccfrf11 function calls during call setup and teardown.
debug ccsip all	Displays the ccswvoice function calls during call setup and teardown.
debug ccsw voice vofr-session	Displays the ccswvoice function calls during call setup and teardown.
debug voice vofr	Displays Cisco trunk and FRF.11 trunk call setup attempts; shows which dial peer is used in the call setup.
debug vpm error	Displays the behavior of the Holst state machine.
debug vtsp port	Displays the behavior of the VTSP state machine.
debug vtsp vofr subframe	Displays the first 10 bytes (including header) of selected VoFR subframes for the interface.

debug frame-relay hqf

To display debug messages for Frame Relay (FR) hierarchical queueing framework (HQF) events, use the **debug frame-relay hqf**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug frame-relay hqf

no debug frame-relay hqf

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(28)SB	This command was introduced.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

Use the **debug frame-relay hqf** command to track which quality of service (QoS) features are being used on an interface. QoS for a given FR interface changes depending on the commands being used.

Note You cannot configure weighted fair queueing (WFQ) with HQF; they are mutually exclusive.

To use HQF on an interface, you must complete the following tasks:

- Install an interface level service policy without legacy queueing or payload compression.
- Attach a Modular Quality of Service (QoS) Command-Line Interface (CLI) (MQC) service policy to a permanent virtual circuit (PVC) with no legacy restrictions.

This task is accomplished by adding a service policy to a frame map class. A valid MQC service policy shapes all traffic via the class default and has a child policy to support any further traffic classification, as shown in the following example:

```
policy-map llq
 class voice
  priority 32
policy-map shape1
 class class-default
  shape average 96000
  service-policy llq
policy-map shape2
 class class-default
  shape average 128000
  service-policy llq
map-class frame-relay mqc-class1
```

```
 service-policy output shape1
map-class frame-relay mqc-class2
 service-policy output shape2
interface serial4/0
 encapsulation frame-relay
 frame-relay class mqc-class1 <----- Map-class installed
 frame-relay interface-dlci 16 <----- Inherits map-class1
 frame-relay interface-dlci 17
 class mqc-class2  <----- Map-class installed for DLCI 17
```

Examples

The following is sample output from the **debug frame-relay hqf**command:

```
Router# debug frame-relay hqf
debug frame-relay hqf is enabled
```

```
Router# show running-configuration
.
.
.
00:25:54: %SYS-5-CONFIG_I: Configured from console by console serial4/1
Building configuration...
Current configuration : 167 bytes
!
interface Serial4/1
 serial restart-delay 0
 service-policy output shape
end
```

The following commands and subsequent output show events that occur when HQF is enabled or disabled as a result of queueing changes at the interface level while debugging is on:

```
Router# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# interface serial4/1
Router(config-if)# policy-map shape
Router(config-pmap)# class class-default
Router(config-pmap-c)# shape average 128000 1000
Router(config-pmap-c)# interface serial4/1
Router(config-if)# encapsulation frame-relay
Router(config-if)# frame-relay fragment 80 end-to-end
Router(config-if)# service-policy output shape
Router(config-if)# frame-relay map ip 10.0.0.1 16 payload frf9 stac
00:26:52: Serial4/1- Setting up interface for legacy QOS. <---Indicates legacy QoS is being
 installed on an interface.
00:26:52: Legacy fair-queueing installed on interface. <---Indicates that legacy QoS is
being installed and HQF is being removed. You see this only with interface fragmentation
and service policies since these policies must be able to support both QoS mechanisms. This
 usually means that either payload compression has been enabled on an interface or legacy
queueing has been set up on the main interface.
Router(config-if)# no frame-relay map ip 10.0.0.1 16 payload frf9 stac
00:27:08: Serial4/1- Setting up HQF/MQC QOS. <---Indicates that the last legacy restriction
 has been removed and HQF is being installed on the interface.
00:27:08: Serial4/1- Setting up interface for legacy QOS. <--- Indicates that legacy QoS
is being installed on the interface.
Router# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# interface serial4/1
Router(config-if)# frame-relay map ip 10.0.0.1 16
Router(config-if)# no service-policy output shape
Router(config-if)# no frame-relay fragment 80 end-to-end
```

The following commands and subsequent output show events that occur when HQF is enabled or disabled as a result of queueing changes at the PVC level while debugging is on:

```
Router# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# interface serial4/1
```

```
Router(config-if)# map-class frame-relay frts-shape
Router(config-map-class)# frame-relay fragment 80
Router(config-map-class)# service-policy output shape
Router(config-map-class)# interface serial4/1
Router(config-if)# frame-relay interface-dlci 16
Router(config-fr-dlci)# class frts-shape
00:28:54: Serial4/1- Setting up HQF/MQC QOS. <---Indicates that the last legacy restriction
 has been removed and that HQF is being installed on the interface.
Router(config-fr-dlci)# no class frts-shape
00:29:02: Serial4/1- Setting up interface for legacy QOS. <--- Indicates that legacy QoS
has been installed on the interface.
```

Related Commands

Command	Description
show debug	Displays active debug output.

debug frame-relay informationelements

To display information about Frame Relay Layer 3 (network layer) information element parsing and construction, use the **debug frame-relay informationelements**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug frame-relay informationelements

no debug frame-relay informationelements

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

Within the FRF.4/Q.933 signalling specification, messages are divided into subunits called information elements. Each information element defines parameters specific to the call. These parameters can be values configured on the router, or values requested from the network.

The **debug frame-relay informationelements**command shows the signalling message in hexadecimal format. Use this command to determine parameters being requested and granted for a call.

Caution Use the **debug frame-relay informationelements**command when the **debug frame-relay callcontrol** command does not explain why calls are not being set up.

Note The **debug frame-relay informationelements**command displays a substantial amount of information in bytes. You must be familiar with FRF.4/Q.933 to decode the information contained within the debug output.

Examples

The following is sample output from the **debug frame-relay informationelements** command. In this example, each information element has a length associated with it. For those with odd-numbered lengths, only the specified bytes are valid, and the extra byte is invalid. For example, in the message “Call Ref, length: 3, 0x0200 0x0100,” only “02 00 01” is valid; the last “00” is invalid.

```
lw0d# debug frame-relay informationelements
Router: Outgoing MSG_SETUP
Router: Dir: U --> N, Type: Prot Disc, length: 1, 0x0800
Router: Dir: U --> N, Type: Call Ref, length: 3, 0x0200 0x0100
Router: Dir: U --> N, Type: Message type, length: 1, 0x0500
Router: Dir: U --> N, Type: Bearer Capability, length: 5, 0x0403 0x88A0 0xCF00
Router: Dir: U --> N, Type: DLCI, length: 4, 0x1902 0x46A0
Router: Dir: U --> N, Type: Link Lyr Core, length: 27, 0x4819 0x090B 0x5C0B 0xDC0A
Router:                   0x3140 0x31C0 0x0B21 0x4021
Router:                   0xC00D 0x7518 0x7598 0x0E09
Router:                   0x307D 0x8000
Router: Dir: U --> N, Type: Calling Party, length: 12, 0x6C0A 0x1380 0x3837 0x3635
Router:                   0x3433 0x3231
Router: Dir: U --> N, Type: Calling Party Subaddr, length: 4, 0x6D02 0xA000
```

```
Router: Dir: U --> N, Type: Called Party, length: 11, 0x7009 0x9331 0x3233 0x3435
Router:                  0x3637 0x386E
Router: Dir: U --> N, Type: Called Party Subaddr, length: 4, 0x7102 0xA000
Router: Dir: U --> N, Type: Low Lyr Comp, length: 5, 0x7C03 0x88A0 0xCE65
Router: Dir: U --> N, Type: User to User, length: 4, 0x7E02 0x0000
```

The following table explains the information elements shown in the example.

Table 20: Information Elements in a Setup Message

Information Element	Description
Prot Disc	Protocol discriminator.
Call Ref	Call reference.
Message type	Message type such as setup, connect, and call proceeding.
Bearer Capability	Coding format such as data type, and Layer 2 and Layer 3 protocols.
DLCI	Data-link connection identifier.
Link Lyr Core	Link-layer core quality of service (QoS) requirements.
Calling Party	Type of source number (X121/E164) and the number.
Calling Party Subaddr	Subaddress that originated the call.
Called Party	Type of destination number (X121/E164) and the number.
Called Party Subaddr	Subaddress of the called party.
Low Lyr Comp	Coding format, data type, and Layer 2 and Layer 3 protocols intended for the end user.
User to User	Information between end users.

Related Commands

Command	Description
debug frame-relay callcontrol	Displays Frame Relay Layer 3 (network layer) call control information.

debug frame-relay ip tcp header-compression

To display debugging information about TCP/IP header compression on Frame Relay interfaces, use the **debug frame-relay ip tcp header-compression**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug frame-relay ip tcp header-compression

no debug frame-relay ip tcp header-compression

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
10.0	This command was introduced.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.4(9)T	This command was modified to display debugging output for control protocol frames for Frame Relay Forum Implementation Agreement (FRF) .20.
12.4(11)T	This command was modified to display debugging output for Enhanced Compressed Real-Time Transport Protocol (ECRTP).

Usage Guidelines

The **debug frame-relay ip tcp header-compression**command shows the control packets that are passed to initialize IP header compression (IPHC) on a permanent virtual circuit (PVC). For Cisco IPHC, typically two packets are passed: one sent and one received per PVC. (Inverse Address Resolution Protocol (InARP) packets are sent on PVCs that do not have a mapping defined between a destination protocol address and the data-link connection identifier (DLCI) or Frame Relay PVC bundle that connects to the destination address.) For FRF .20 IPHC, typically four packets are passed per PVC.

Debug messages are displayed only if the IPHC control protocol is renegotiated (for an interface or PVC state change or for a configuration change).

Examples

The following is sample output from the **debug frame-relay**ip tcp header-compression command when Cisco IPHC (not FRF .20 IPHC) is configured in the IPHC profile:

```
Router# debug frame-relay ip tcp header-compression
*Nov 14 09:22:07.991: InARP REQ: Tx compr_flags 43 *Nov 14 09:22:08.103: InARP RSP: Rx
compr_flags: 43
```

The following is sample output from the **debug frame-relay**ip tcp header-compression command when FRF .20 IPHC (without either Real-time Transport Protocol (RTP) or ECRTP) is configured in the IPHC profile:

```
Router# debug frame-relay ip tcp header-compression
FRF20(DLCI 16): Rxed Request, state 0
     : ident 0, tot len 19, conf_opts FE, len 15
       negotiation codes 1, version 1
  Par: IPV4, len 12, TCP_SPACE 16, NON_TCP_SPACE 0,
       F_MAX_PERIOD 256, F_MAX_TIME 5, MAX_HEADER 168 FRF20(DLCI 16): Txed Ack, state 0
     : ident 0, tot len 19, conf_opts FE, len 15
       negotiation codes 1, version 1
  Par: IPV4, len 12, TCP_SPACE 16, NON_TCP_SPACE 0,
       F_MAX_PERIOD 256, F_MAX_TIME 5, MAX_HEADER 168 FRF20(DLCI 16): Txed Request, state
0
     : ident 3, tot len 19, conf_opts FE, len 15
       negotiation codes 0, version 1
  Par: IPV4, len 12, TCP_SPACE 16, NON_TCP_SPACE 0,
       F_MAX_PERIOD 256, F_MAX_TIME 5, MAX_HEADER 168 FRF20(DLCI 16): Rxed Ack, state 2
     : ident 3, tot len 19, conf_opts FE, len 15
       negotiation codes 0, version 1
  Par: IPV4, len 12, TCP_SPACE 16, NON_TCP_SPACE 0,
       F_MAX_PERIOD 256, F_MAX_TIME 5, MAX_HEADER 168 *Nov 14 09:18:37.019:
FRF20(DLCI 16): STARTING IPHC
```

The following is sample output from the **debug frame-relay**ip tcp header-compression command when FRF .20 IPHC and RTP are configured in the IPHC profile:

```
Router# debug frame-relay ip tcp header-compression
FRF20(DLCI 16): Txed Request, state 1
     : ident 0, tot len 21, conf_opts FE, len 17
       negotiation codes 1, version 1
  Par: IPV4, len 14, TCP_SPACE 16, NON_TCP_SPACE 16,
       F_MAX_PERIOD 256, F_MAX_TIME 5, MAX_HEADER 168
01:33:06: Subopt: rtp enabled
```

The following is sample output from the **debug frame-relay**ip tcp header-compression command when FRF .20 IPHC and ECRTP are configured in the IPHC profile:

```
Router# debug frame-relay ip tcp header-compression
FRF20(DLCI 16): Txed Request, state 1
     : ident 0, tot len 21, conf_opts FE, len 17
       negotiation codes 1, version 1
  Par: IPV4, len 14, TCP_SPACE 16, NON_TCP_SPACE 16,
       F_MAX_PERIOD 256, F_MAX_TIME 5, MAX_HEADER 168
01:33:06: Subopt: ecrtp enabled
```

The below table describes the significant fields shown in the displays.

Table 21: debug frame-relay ip tcp header-compression Field Descriptions

Field	Description
InARP REQ: Tx	Indicates that an InARP request was sent or received. Following are the possible values: • InARP REQ Tx--An InARP request was sent. • InARP REQ Rx--An InARP request was received.

Field	Description
InARP RSP: Rx	Indicates that an InARP response was sent or received. Following are the possible values: • InARP REQ Tx--An InARP response was sent. • InARP REQ Rx--An InARP response was received.
compr_flags: 43	Compression flags that Frame Relay peers use to negotiate Cisco IPHC options. It consists of a bit mask, and the number is displayed in hexadecimal format. Following are the bits: • 0x0001--TCP IPHC • 0x0002--RTP IPHC • 0x0004--Passive TCP compression • 0x0008--Passive RTP compression • 0x0040--Frame Relay IPHC options
FRF20(DLCI 16)	Indicates that the DLCI for this packet is configured with FRF .20 IPHC.
Txed Request	Direction of the IPHC control protocol message. Following are the possible values: • Txed Request • Txed Ack • Rxed Request • Rxed Ack Txed (transmitted) or Rxed (received) indicates the message direction, and Request or Ack (acknowledgement) indicates the message type. A peer sends a request indicating its configuration, and the other peer replies with an acknowledgement indicating its configuration. The lowest configuration value of this two-frame exchange sets the parameters in one direction. This means that typically four frames are exchanged in total: two Request/Ack pairs, with each pair negotiating the parameters in one direction.

Field	Description
state 1	State of the FRF .20 IPHC protocol request. Following are the possible values: 0--FRF20_DISABLED. FRF .20 is disabled (because of an inactive PVC, an interface that is down, or a configuration mismatch). 1--FRF20_REQ_SENT. An FRF .20 control protocol request has been sent. 2--FRF20_REQ_RXED. An FRF .20 control protocol request has been received. 3--FRF20_WAIT_REQ. An FRF .20 control protocol request has been sent and acknowledged, and the local end is waiting for a request from the peer. 4--FRF20_OPERATIONAL. The FRF .20 control protocol is successfully negotiated, and frames can be compressed.
ident 0	Identifier. This is the transaction number used to correlate an FRF .20 control protocol request with an acknowledgement. This number is the same in messages that correspond to each other.
tot len 21	Sum (in bytes) of the lengths of the following: • All parameters • Negotiation codes • Identifier • Suboptions for each parameter set (IPV4 or IPV6)
conf_opts FE	Type of PPP parameter (expressed in hexadecimal). For FRF .20, the only possible value is FE (254 in decimal).
len 17	Total length of all parameters (in bytes).
negotiation codes 1	Negotiation state with the peer. Following are the possible values: • 0--Reply with response only. • 1--Reply with response and initiate request. With a response only, sending a response frame completes the negotiation. With a response and initiate request, the local peer also must send a request.

Field	Description
version 1	Version of the FRF .20 control protocol.
Par	List of parameters and values.
IPV4	Datagram type. The value is always IPV4, because Cisco IPHC does not support IPv6.
len 14	Total length (in bytes) of all parameters starting with IP type and ending with associated suboptions (if any). The value is greater than or equal to 12 depending on the suboptions.
TCP_SPACE 16	Maximum value of a TCP context identifier (CID) in the space of context identifiers allocated for TCP. Range: 3-255. Default value: 16. A value of zero means that TCP headers are not being compressed.
NON_TCP_SPACE 16	Maximum value of a context identifier (CID) in the space of context identifiers allocated for non-TCP. Range: 3-1000. Cisco routers do not support the maximum value (65535) of the FRF .20 specification. Default value: 16. A value of zero means that non-TCP headers are not being compressed. These context identifiers are carried in COMPRESSED_NON_TCP, COMPRESSED_UDP and COMPRESSED_RTP packet headers.
F_MAX_PERIOD 256	Largest number of compressed non-TCP headers that can be sent without sending a full header. Range: 1-65535. Default value: 256. A value of zero indicates infinity, which means that the number of consecutive COMPRESSED_NON_TCP headers is unlimited.
F_MAX_TIME 5	Maximum time interval (in seconds) between full (uncompressed) headers. Range: 1-255. Default value: 5. A value of zero indicates infinity (meaning that no full headers will be transmitted).
MAX_HEADER 168	Largest header size (in bytes) that can be compressed. Range: 60-168. Cisco routers do not support the full range of values (60-65535) of the FRF .20 specification. Default value: 168.
01:33:06	Timestamp of the debug command output.
Subopt	Compression suboptions that are enabled. The value is either rtp or ecrtp.

debug frame-relay lapf

To display Frame Relay switched virtual circuit (SVC) Layer 2 information, use the **debug frame-relay lapf** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug frame-relay lapf

no debug frame-relay lapf

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

Use the **debug frame-relay lapf** command to troubleshoot the data-link control portion of Layer 2 that runs over data-link connection identifier (DLCI) 0. Use this command only if you have a problem bringing up Layer 2. You can use the **show interface serial** command to determine the status of Layer 2. If it shows a Link Access Procedure, Frame Relay (LAPF) state of down, Layer 2 has a problem.

Examples

The following is sample output from the **debug frame-relay lapf**command. In this example, a line being brought up indicates an exchange of set asynchronous balanced mode extended (SABME) and unnumbered acknowledgment (UA) commands. A SABME is initiated by both sides, and a UA is the response. Until the SABME gets a UA response, the line is not declared to be up. The p/f value indicates the poll/final bit setting. TX means send, and RX means receive.

```
Router# debug frame-relay lapf
Router: *LAPF Serial0 TX -> SABME Cmd p/f=1
Router: *LAPF Serial0 Enter state 5
Router: *LAPF Serial0 RX <- UA Rsp p/f=1
Router: *LAPF Serial0 lapf_ua_5
Router: *LAPF Serial0 Link up!
Router: *LAPF Serial0 RX <- SABME Cmd p/f=1
Router: *LAPF Serial0 lapf_sabme_78
Router: *LAPF Serial0 TX -> UA Rsp p/f=1
```

In the following example, a line in an up LAPF state should see a steady exchange of RR (receiver ready) messages. TX means send, RX means receive, and N(R) indicates the receive sequence number.

```
Router# debug frame-relay lapf
Router: *LAPF Serial0 T203 expired, state = 7
Router: *LAPF Serial0 lapf_rr_7
Router: *LAPF Serial0 TX -> RR Rsp p/f=1, N(R)= 3
Router: *LAPF Serial0 RX <- RR Cmd p/f=1, N(R)= 3
Router: *LAPF Serial0 lapf_rr_7
Router: *LAPF Serial0 TX -> RR Rsp p/f=1, N(R)= 3
Router: *LAPF Serial0 RX <- RR Cmd p/f=1, N(R)= 3
Router: *LAPF Serial0 lapf_rr_7
```

debug frame-relay lmi

To display information on the local management interface (LMI) packets exchanged by the router and the Frame Relay service provider, use the **debug frame-relay lmi**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug frame-relay lmi [**interface** *name*]

no debug frame-relay lmi [**interface** *name*]

Syntax Description

interface *name*	(Optional) The name of interface.

Command Modes

Privileged EXEC

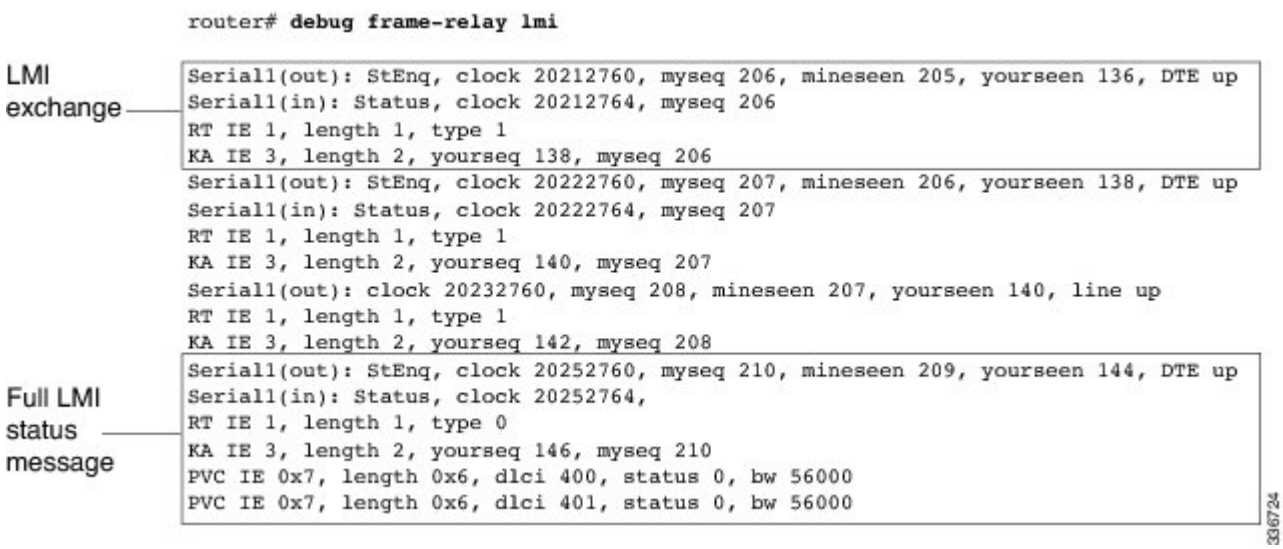
Usage Guidelines

You can use this command to determine whether the router and the Frame Relay switch are sending and receiving LMI packets properly.

Note Because the **debug frame-relay lmi** command does not generate much output, you can use it at any time, even during periods of heavy traffic, without adversely affecting other users on the system.

Examples

The following is sample output from the **debug frame-relay lmi** command:



The first four lines describe an LMI exchange. The first line describes the LMI request the router has sent to the switch. The second line describes the LMI reply the router has received from the switch. The third and

fourth lines describe the response to this request from the switch. This LMI exchange is followed by two similar LMI exchanges. The last six lines consist of a full LMI status message that includes a description of the two permanent virtual circuits (PVCs) of the router.

The below table describes the significant fields shown in the first line of the display.

Table 22: debug frame-relay lmi Field Descriptions

Field	Description
Serial1(out)	Indicates that the LMI request was sent out on serial interface 1.
StEnq	Command mode of message, as follows: • StEnq--Status inquiry • Status--Status reply
clock 20212760	System clock (in milliseconds). Useful for determining whether an appropriate amount of time has transpired between events.
myseq 206	Myseq counter maps to the CURRENT SEQ counter of the router.
yourseen 136	Yourseen counter maps to the LAST RCVD SEQ counter of the switch.
DTE up	Line protocol up/down state for the DTE (user) port.

The below table describes the significant fields shown in the third and fourth lines of the display.

Table 23: debug frame-relay lmi Field Descriptions

Field	Description
RT IE 1	Value of the report type information element.
length 1	Length of the report type information element (in bytes).
type 1	Report type in RT IE.
KA IE 3	Value of the keepalive information element.
length 2	Length of the keepalive information element (in bytes).
yourseq 138	Yourseq counter maps to the CURRENT SEQ counter of the switch.

Field	Description
myseq 206	Myseq counter maps to the CURRENT SEQ counter of the router.

The below table describes the significant fields shown in the last line of the display.

Table 24: debug frame-relay lmi Field Descriptions

Field	Description
PVC IE 0x7	Value of the PVC information element type.
length 0x6	Length of the PVC IE (in bytes).
dlci 401	DLCI decimal value for this PVC.
status 0	Status value. Possible values include the following: • 0x00--Added/inactive • 0x02--Added/active • 0x04--Deleted • 0x08--New/inactive • 0x0a--New/active
bw 56000	Committed information rate (in decimal) for the DLCI.

debug frame-relay multilink

To display debug messages for multilink Frame Relay bundles and bundle links, use the **debug frame-relay multilink**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug frame-relay multilink [**control** [**mfr** *number*| **serial** *number*]]

no debug frame-relay multilink

Syntax Description

control	(Optional) Displays incoming and outgoing bundle link control messages and bundle link status changes.
mfr *number*	(Optional) Displays information for a specific bundle interface.
serial *number*	(Optional) Displays information for a specific bundle link interface.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(17)S	This command was introduced.
12.0(24)S	This command was introduced on VIP-enabled Cisco 7500 series routers.
12.2(8)T	This command was integrated into Cisco IOS Release 12.2(8)T.
12.2(14)S	This command was integrated into Cisco IOS Release 12.2(14)S.
12.2(17b)SXA	This command was integrated into Cisco IOS Release 12.2(17b)SXA.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.

Usage Guidelin

Caution Using the **debug frame-relay multilink** command without the **control** keyword could severely impact router performance and is not recommended.

Using the **debug frame-relay multilink** command without the **mfr** or **serial** keywords displays error conditions that occur at the bundle layer.

Examples

The following example shows output from the **debug frame-relay multilink** command for bundle "MFR0," which has three bundle links:

```
Router# debug frame-relay multilink control MFR0
00:42:54:Serial5/3(o):msg=Add_link, Link=Serial5/3, Bundle=MFR0, Link id=Serial5/3, BL
state=Idle
E1 00 01 01 07 4D 46 52 30 00
00:42:54:Serial5/2(o):msg=Add_link, Link=Serial5/2, Bundle=MFR0, Link id=Serial5/2, BL
state=Idle
E1 00 01 01 07 4D 46 52 30 00
00:42:54:Serial5/1(o):msg=Add_link, Link=Serial5/1, Bundle=MFR0, Link id=Serial5/1, BL
state=Idle
E1 00 01 01 07 4D 46 52 30 00
00:42:54:%LINK-3-UPDOWN:Interface MFR0, changed state to down
00:42:54:Serial5/3(i):msg=Add_link_ack, Link=Serial5/3, Bundle=MFR0, Link id=Serial5/3, BL
 state=Add_sent
E1 00 02 01 07 4D 46 52 30 00
00:42:54:Serial5/2(i):msg=Add_link_ack, Link=Serial5/2, Bundle=MFR0, Link id=Serial5/2, BL
 state=Add_sent
E1 00 02 01 07 4D 46 52 30 00
00:42:54:Serial5/1(i):msg=Add_link_ack, Link=Serial5/1, Bundle=MFR0, Link id=Serial5/1, BL
 state=Add_sent
E1 00 02 01 07 4D 46 52 30 00
00:42:54:%SYS-5-CONFIG_I:Configured from console by console
00:43:00:Serial5/1(i):msg=Add_link, Link=Serial5/1, Bundle=MFR0, Link id=Serial5/1, BL
state=Ack_rx
E1 00 01 01 07 4D 46 52 30 00
00:43:00:Serial5/1(o):msg=Add_link_ack, Link=Serial5/1, Bundle=MFR0, Link id=Serial5/1, BL
 state=Ack_rx
E1 00 02 01 07 4D 46 52 30 00
00:43:00:%LINK-3-UPDOWN:Interface MFR0, changed state to up
00:43:00:Serial5/1(i):msg=Hello, Link=Serial5/1, Bundle=MFR0, Linkid=Serial5/1, BL state=Up
E1 00 04 03 06 30 A7 E0 54 00
00:43:00:Serial5/1(o):msg=Hello_ack, Link=Serial5/1, Bundle=MFR0, Link id=Serial5/1, BL
state=Up
E1 00 05 03 06 90 E7 0F C2 06
00:43:01:Serial5/2(i):msg=Add_link, Link=Serial5/2, Bundle=MFR0, Link id=Serial5/2, BL
state=Ack_rx
E1 00 01 01 07 4D 46 52 30 00
00:43:01:Serial5/2(o):msg=Add_link_ack, Link=Serial5/2, Bundle=MFR0, Link id=Serial5/2, BL
 state=Ack_rx
E1 00 02 01 07 4D 46 52 30 00
00:43:01:Serial5/2(i):msg=Hello, Link=Serial5/2, Bundle=MFR0, Linkid=Serial5/2, BL state=Up
E1 00 04 03 06 30 A7 E0 54 00
00:43:01:Serial5/2(o):msg=Hello_ack, Link=Serial5/2, Bundle=MFR0, Link id=Serial5/2, BL
state=Up
E1 00 05 03 06 90 E7 0F C2 06
00:43:01:%LINEPROTO-5-UPDOWN:Line protocol on Interface Serial5/1, changed state to up
00:43:01:Serial5/3(i):msg=Add_link, Link=Serial5/3, Bundle=MFR0, Link id=Serial5/3, BL
state=Ack_rx
E1 00 01 01 07 4D 46 52 30 00
00:43:01:Serial5/3(o):msg=Add_link_ack, Link=Serial5/3, Bundle=MFR0, Link id=Serial5/3, BL
 state=Ack_rx
E1 00 02 01 07 4D 46 52 30 00
00:43:01:Serial5/3(i):msg=Hello, Link=Serial5/3, Bundle=MFR0, Linkid=Serial5/3, BL state=Up
E1 00 04 03 06 30 A7 E0 54 00
```

```
00:43:01:Serial5/3(o):msg=Hello_ack, Link=Serial5/3, Bundle=MFR0, Link id=Serial5/3, BL
state=Up
E1 00 05 03 06 90 E7 0F C2 06
00:43:02:%LINEPROTO-5-UPDOWN:Line protocol on Interface Serial5/2 , changed state to up
00:43:02:%LINEPROTO-5-UPDOWN:Line protocol on Interface Serial5/3 , changed state to up
```

The table below describes the significant fields shown in the display.

Table 25: debug frame-relay multilink Field Descriptions

Field	Description
msg	Type of bundle link control message that was sent or received.
Link	Interface number of the bundle link.
Bundle	Bundle with which the link is associated.
Link id	Bundle link identification name.
BL state	Operational state of the bundle link.

Related Commands

Command	Description
show frame-relay multilink	Displays configuration information and statistics about multilink Frame Relay bundles and bundle links.

debug frame-relay networklayerinterface

To display Network Layer Interface (NLI) information, use the **debug frame-relay networklayerinterface**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug frame-relay networklayerinterface

no debug frame-relay networklayerinterface

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

The Frame Relay switched virtual circuit (SVC) signaling subsystem is decoupled from the rest of the router code by means of the NLI intermediate software layer.

The **debug frame-relay networklayerinterface**command shows activity within the network-layer interface when a call is set up or torn down. All output that contains an *NL* relates to the interaction between the Q.933 signaling subsystem and the NLI.

Note The **debug frame-relay networklayerinterface** command has no significance to anyone not familiar with the inner workings of the Cisco IOS software. This command is typically used by service personnel to debug problem situations.

Examples

The following is sample output from the **debug frame-relay networklayerinterface** command. This example displays the output generated when a call is set up. The second example shows the output generated when a call is torn down.

```
Router# debug frame-relay networklayerinterface
Router: NLI STATE: L3_CALL_REQ, Call ID 1 state 0
Router: NLI: Walking the event table 1
Router: NLI: Walking the event table 2
Router: NLI: Walking the event table 3
Router: NLI: Walking the event table 4
Router: NLI: Walking the event table 5
Router: NLI: Walking the event table 6
Router: NLI: Walking the event table 7
Router: NLI: Walking the event table 8
Router: NLI: Walking the event table 9
Router: NLI: NL0_L3CallReq
Router: NLI: State: STATE_NL_NULL, Event: L3_CALL_REQ, Next: STATE_L3_CALL_REQ
Router: NLI: Enqueued outgoing packet on holdq
Router: NLI: Map-list search: Found maplist bermuda
Router: daddr.subaddr 0, saddr.subaddr 0, saddr.subaddr 0
Router: saddr.subaddr 0, daddr.subaddr 0, daddr.subaddr 0
Router: nli_parameter_negotiation
Router: NLI STATE: NL_CALL_CNF, Call ID 1 state 10
Router: NLI: Walking the event table 1
Router: NLI: Walking the event table 2
Router: NLI: Walking the event table 3
```

```
Router: NLI: NLx_CallCnf
Router: NLI: State: STATE_L3_CALL_REQ, Event: NL_CALL_CNF, Next: STATE_NL_CALL_CNF
Router: Checking maplist "junk"
Router: working with maplist "bermuda"
Router: Checking maplist "bermuda"
Router: working with maplist "bermuda"
Router: NLI: Emptying holdQ, link 7, dlci 100, size 104
Router# debug frame-relay networklayerinterface
Router: NLI: L3 Call Release Req for Call ID 1
Router: NLI STATE: L3_CALL_REL_REQ, Call ID 1 state 3
Router: NLI: Walking the event table 1
Router: NLI: Walking the event table 2
Router: NLI: Walking the event table 3
Router: NLI: Walking the event table 4
Router: NLI: Walking the event table 5
Router: NLI: Walking the event table 6
Router: NLI: Walking the event table 7
Router: NLI: Walking the event table 8
Router: NLI: Walking the event table 9
Router: NLI: Walking the event table 10
Router: NLI: NLx_L3CallRej
Router: NLI: State: STATE_NL_CALL_CNF, Event: L3_CALL_REL_REQ, Next: STATE_L3_CALL_REL_REQ

Router: NLI: junk: State: STATE_NL_NULL, Event: L3_CALL_REL_REQ, Next: STATE_NL_NULL
Router: NLI: Map-list search: Found maplist junk
Router: daddr.subaddr 0, saddr.subaddr 0, saddr.subaddr 0
Router: saddr.subaddr 0, daddr.subaddr 0, daddr.subaddr 0
Router: nli_parameter_negotiation
Router: NLI STATE: NL_REL_CNF, Call ID 1 state 0
Router: NLI: Walking the event table 1
Router: NLI: Walking the event table 2
Router: NLI: Walking the event table 3
Router: NLI: Walking the event table 4
Router: NLI: Walking the event table 5
Router: NLI: Walking the event table 6
Router: NLI: Walking the event table 7
Router: NLI: NLx_RelCnf
Router: NLI: State: STATE_NL_NULL, Event: NL_REL_CNF, Next: STATE_NL_NULL
```

The below table describes the significant states and events shown in the display.

Table 26: NLI State and Event Descriptions

State and Event	Description
L3_CALL_REQ	Internal call setup request. Network layer indicates that an SVC is required.
STATE_NL_NULL	Call in initial state--no call exists.
STATE_L3_CALL_REQ	Setup message sent out and waiting for a reply. This is the state the network-layer state machine changes to when a call request is received from Layer 3 but no confirmation has been received from the network.
NL_CALL_CNF	Message sent from the Q.933 signalling subsystem to the NLI asking that internal resources be allocated for the call.
STATE_L3_CALL_CNF	Q.933 state indicating that the call is active. After the network confirms a call request using a connect message, the Q.933 state machine changes to this state.

State and Event	Description
STATE_NL_CALL_CNF	Internal software state indicating that software resources are assigned and the call is up. After Q.933 changes to the STATE_L3_CALL_CNF state, it sends an NL_CALL_CNF message to the network-layer state machine, which then changes to the STATE_NL_CALL_CNF state.
L3_CALL_REL_REQ	Internal request to release the call.
STATE_L3_CALL_REL_REQ	Internal software state indicating the call is in the process of being released. At this point, the Q.933 subsystem is told that the call is being released and a disconnect message goes out for the Q.933 subsystem.
NL_REL_CNF	Indication from the Q.933 signalling subsystem that the signalling subsystem is releasing the call. After receiving a release complete message from the network indicating that the release process is complete, the Q.933 subsystem sends an NL_REL_CNF event to the network-layer subsystem.

Related Commands

Command	Description
debug frame-relay callcontrol	Displays Frame Relay Layer 3 (network layer) call control information.

debug frame-relay packet

To display information on packets that have been sent on a Frame Relay interface, use the **debug frame-relay packet** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug frame-relay packet [**interface** *name* [**dlci** *value*]]

no debug frame-relay packet [**interface** *name* [**dlci** *value*]]

Syntax Description

interface *name*	(Optional) Name of interface or subinterface.
dlci *value*	(Optional) Data-link connection indentifier (DLCI) decimal value.

Command Modes

Privileged EXEC

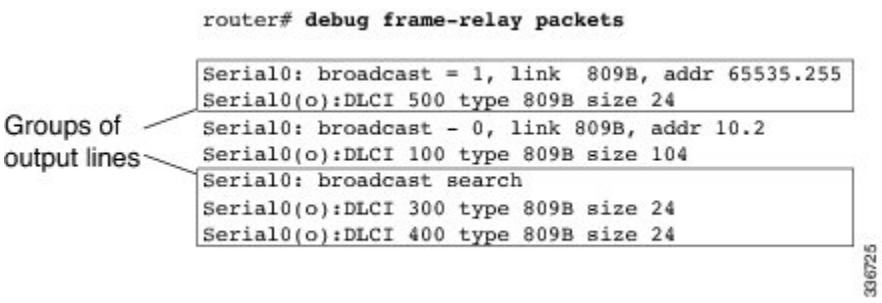
Usage Guidelines

This command helps you analyze the packets that are sent on a Frame Relay interface. Because the **debug frame-relay packet** command generates a substantial amount of output, only use it when traffic on the Frame Relay network is fewer than 25 packets per second. Use the options to limit the debugging output to a specific DLCI or interface.

To analyze the packets *received* on a Frame Relay interface, use the **debug frame-relay** command.

Examples

The following is sample output from the **debug frame-relay packet**command:



The **debug frame-relay packet**output consists of groups of output lines; each group describes a Frame Relay packet that has been sent. The number of lines in the group can vary, depending on the number of DLCIs on which the packet was sent. For example, the first two pairs of output lines describe two different packets, both of which were sent out on a single DLCI. The last three lines describe a single Frame Relay packet that was sent out on two DLCIs.

The below table describes the significant fields shown in the display.

Table 27: debug frame-relay packet Field Descriptions

Field	Description
Serial0:	Interface that has sent the Frame Relay packet.
broadcast = 1	Destination of the packet. Possible values include the following: • broadcast = 1--Broadcast address • broadcast = 0--Particular destination • broadcast search--Searches all Frame Relay map entries for this particular protocol that include the **broadcast**keyword.
link 809B	Link type, as documented in the **debug frame-relay**command.
addr 65535.255	Destination protocol address for this packet. In this case, it is an AppleTalk address.
Serial0(o):	(o) indicates that this is an output event.
DLCI 500	Decimal value of the DLCI.
type 809B	Packet type, as documented under the **debug frame-relay** command.
size 24	Size of this packet (in bytes).

The following lines describe a Frame Relay packet sent to a particular address; in this case AppleTalk address 10.2:

```
Serial0: broadcast - 0, link 809B, addr 10.2
Serial0(o):DLCI 100 type 809B size 104
```

The following lines describe a Frame Relay packet that went out on two different DLCIs, because two Frame Relay map entries were found:

```
Serial0: broadcast search
Serial0(o):DLCI 300 type 809B size 24
Serial0(o):DLCI 400 type 809B size 24
```

The following lines do not appear. They describe a Frame Relay packet sent to a true broadcast address.

```
Serial1: broadcast search
Serial1(o):DLCI 400 type 800 size 288
```

debug frame-relay ppp

To display debugging information, use the **debug frame-relay ppp**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug frame-relay ppp

no debug frame-relay ppp

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

This command displays error messages for link states and Local Management Interface (LMI) status changes for PPP over Frame Relay sessions.

To debug process-switched packets, use the **debug frame-relay packet** or **debug ppp packet** commands. To analyze the packets that have been *sent* on a Frame Relay interface, use the **debug frame-relay packe t** command.

The **debug frame-relay ppp**command is generated from process-level switching only and is not CPU intensive.

Examples

The following shows output from the **debug frame-relay ppp** command where the encapsulation failed for VC 100.

```
Router# debug frame-relay ppp
FR-PPP: encaps failed for FR VC 100 on Serial0 down
FR-PPP: input- Serial0 vc or va down, pak dropped
```

The following shows the output from the **debug frame relay ppp**and **debug frame-relay packet** commands**.**This example shows a virtual interface (virtual interface 1) establishing a PPP connection over PPP.

```
Router# debug frame-relay ppp
Router# debug frame-relay packet
Vi1 LCP: O CONFREQ [Closed] id 1 len 10
Vi1 LCP:    MagicNumber 0xE0638565 (0x0506E0638565)
Serial2/1(o): dlci 201(0x3091), NLPID 0x3CF(PPP), datagramsize 16
Vi1 PPP: I pkt type 0xC021, datagramsize 14
Vi1 LCP: I CONFACK [REQsent] id 1 len 10
Vi1 LCP:    MagicNumber 0xE0638565 (0x0506E0638565)
Vi1 PPP: I pkt type 0xC021, datagramsize 14
Vi1 LCP: I CONFREQ [ACKrcvd] id 6 len 10
Vi1 LCP:    MagicNumber 0x000EAD99 (0x0506000EAD99)
Vi1 LCP: O CONFACK [ACKrcvd] id 6 len 10
Vi1 LCP:    MagicNumber 0x000EAD99 (0x0506000EAD99)
Serial2/1(o): dlci 201(0x3091), NLPID 0x3CF(PPP), datagramsize 16
Vi1 IPCP: O CONFREQ [Closed] id 1 len 10
Vi1 IPCP:    Address 170.100.9.10 (0x0306AA64090A)
Serial2/1(o): dlci 201(0x3091), NLPID 0x3CF(PPP), datagramsize 16
Vi1 PPP: I pkt type 0x8021, datagramsize 14
Vi1 IPCP: I CONFREQ [REQsent] id 1 len 10
Vi1 IPCP:    Address 170.100.9.20 (0x0306AA640914)
Vi1 IPCP: O CONFACK [REQsent] id 1 len 10
Vi1 IPCP:    Address 170.100.9.20 (0x0306AA640914)
Serial2/1(o): dlci 201(0x3091), NLPID 0x3CF(PPP), datagramsize 16
```

```
Vi1 PPP: I pkt type 0x8021, datagramsize 14
Vi1 IPCP: I CONFACK [ACKsent] id 1 len 10
Vi1 IPCP:    Address 170.100.9.10 (0x0306AA64090A)
Vi1 PPP: I pkt type 0xC021, datagramsize 16
Vi1 LCP: I ECHOREQ [Open] id 1 len 12 magic 0x000EAD99
Vi1 LCP: O ECHOREP [Open] id 1 len 12 magic 0xE0638565
Serial2/1(o): dlci 201(0x3091), NLPID 0x3CF(PPP), datagramsize 18
Vi1 LCP: O ECHOREQ [Open] id 1 len 12 magic 0xE0638565
Serial2/1(o): dlci 201(0x3091), NLPID 0x3CF(PPP), datagramsize 18
Vi1 LCP: echo_cnt 4, sent id 1, line up
```

The following shows the output for the **debug frame-relay ppp** and **debug frame-relay packet** commands that report a failed PPP over Frame Relay session. The problem is due to a challenge handshake authentication protocol (CHAP) failure.

```
Router# debug frame-relay ppp
Router# debug frame-relay packet
Vi1 LCP: O CONFREQ [Listen] id 24 len 10
Vi1 LCP:    MagicNumber 0xE068EC78 (0x0506E068EC78)
Serial2/1(o): dlci 201(0x3091), NLPID 0x3CF(PPP), datagramsize 16
Vi1 PPP: I pkt type 0xC021, datagramsize 19
Vi1 LCP: I CONFREQ [REQsent] id 18 len 15
Vi1 LCP:    AuthProto CHAP (0x0305C22305)
Vi1 LCP:    MagicNumber 0x0014387E (0x05060014387E)
Vi1 LCP: O CONFACK [REQsent] id 18 len 15
Vi1 LCP:    AuthProto CHAP (0x0305C22305)
Vi1 LCP:    MagicNumber 0x0014387E (0x05060014387E)
Serial2/1(o): dlci 201(0x3091), NLPID 0x3CF(PPP), datagramsize 21
Vi1 PPP: I pkt type 0xC021, datagramsize 14
Vi1 LCP: I CONFACK [ACKsent] id 24 len 10
Vi1 LCP:    MagicNumber 0xE068EC78 (0x0506E068EC78)
Vi1 PPP: I pkt type 0xC223, datagramsize 32
Vi1 CHAP: I CHALLENGE id 12 len 28 from "krishna"
Vi1 LCP: O TERMREQ [Open] id 25 len 4
Serial2/1(o): dlci 201(0x3091), NLPID 0x3CF(PPP), datagramsize 10
Vi1 PPP: I pkt type 0xC021, datagramsize 8
Vi1 LCP: I TERMACK [TERMsent] id 25 len 4
Serial2/1(i): dlci 201(0x3091), pkt type 0x2000, datagramsize 303
%SYS-5-CONFIG_I: Configured from console by console
Vi1 LCP: TIMEout: Time 0x199580 State Listen
```

debug frame-relay pseudowire

To display events and error conditions that occur when binding a Frame Relay data-link connection identifier (DLCI) to a pseudowire, use the **debug frame-relay pseudowire**command in privileged EXEC mode. To disable the display of these events and error conditions, use the **no** form of this command.

debug frame-relay pseudowire

no debug frame-relay pseudowire

Syntax Description

This command contains no arguments or keywords.

Command Default

DLCI events and errors are not displayed.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(26)S	This command was introduced.
12.2(25)S	This command was integrated into Cisco IOS Release 12.2(25)S.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.4(11)T	This command was integrated into Cisco IOS Release 12.4(11)T.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.

Usage Guidelines

The following are examples of Frame Relay pseudowire events:

- Command-line interface (CLI) provisioning events
- Pseudowire circuit status updates
- Failures occurring during the management of these events

Examples

The following example enables the display of Frame Relay pseudowire events. In this example, the interface has been shut down and then enabled.

```
Router# debug frame-relay pseudowire
Router(config)# interface hssi1/0/0
Router(config-if)# shutdown
09:18:33.303: FRoPW [10.15.15.15, 100]: acmgr_circuit_down
09:18:33.303: FRoPW [10.15.15.15, 100]: SW AC update circuit state to down
09:18:33.303: FRoPW [10.15.15.15, 100]: Setting connection DOWN
09:18:35.299: %LINK-5-CHANGED: Interface Hssi1/0/0, changed state to administratively down
```

```
09:18:36.299: %LINEPROTO-5-UPDOWN: Line protocol on Interface Hssi1/0/0, changed state to
down
Router(config-if)# no shutdown
09:18:41.919: %LINK-3-UPDOWN: Interface Hssi1/0/0, changed state to up
09:18:41.919: FRoPW [10.15.15.15, 100]: Local up, sending acmgr_circuit_up
09:18:41.919: FRoPW [10.15.15.15, 100]: Setting pw segment UP
09:18:41.919: FRoPW [10.15.15.15, 100]: PW nni_pvc_status set ACTIVE
09:18:41.919: label_oce_get_label_bundle: flags 14 label 28
09:18:42.919: %LINEPROTO-5-UPDOWN: Line protocol on Interface Hssi1/0/0, changed state to
up
```

The below table describes the significant fields shown in the display.

Table 28: debug frame-relay pseudowire Field Descriptions

Field	Description
Time (09.18.41)	When the event occurred (in hours, minutes, and seconds).
[10.15.15.15, 100]	10.15.15.15 is the IP address of the peer provider edge (PE) router. 100 is the DLCI number of the Frame Relay permanent virtual circuit (PVC) used for this pseudowire.

debug frame-relay redundancy

To debug Frame Relay and Multilink Frame Relay redundancy on the networking device, use the **debug frame-relay redundancy** command in privileged EXEC mode. To disable the display of debugging output, use the **no** form of this command.

debug frame-relay redundancy

no debug frame-relay redundancy

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(22)S	This command was introduced on the Cisco 7500 series and Cisco 10000 series Internet routers.
12.2(18)S	This command was integrated into Cisco IOS Release 12.2(18)S on Cisco 7500 series routers.
12.2(20)S	Support was added for the Cisco 7304 router. The Cisco 7500 series router is not supported in Cisco IOS Release 12.2(20)S.
12.0(28)S	SSO support was added to the Multilink Frame Relay feature on the Cisco 12000 series Internet router.
12.2(25)S	SSO support was added to the Multilink Frame Relay feature on the Cisco 12000 series Internet router.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.

Usage Guidelines

Use this command to debug Frame Relay synchronization problems. The **debug frame-relay redundancy** command logs synchronization events and errors.

Examples

The following example displays debug messages regarding Frame Relay redundancy on the networking device:

```
Router# debug frame-relay redundancy
```

Related Commands

Command	Description
frame-relay redundancy auto-sync lmi-sequence-numbers	Configures LMI synchronization parameters.

debug frame-relay switching

To display debugging messages for switched Frame Relay permanent virtual circuits (PVCs), use the **debug frame-relay switching**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug frame-relay switching interface *interface dlci* [**interval** *interval*]

no debug frame-relay switching

Syntax Description

interface *interface*	The name of the Frame Relay interface.
dlci	The DLCI number of the switched PVC to be debugged.
interval *interval*	(Optional) Interval in seconds at which debugging messages will be updated.

Command Default

The default interval is 1 second.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(12)S	This command was introduced.
12.1(5)T	This command was integrated into Cisco IOS Release 12.1(5)T.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

The **debug frame-relay switching** command can be used only on switched Frame Relay PVCs, not terminated PVCs.

Debug statistics are displayed only if they have changed.

Note Although statistics are displayed at configured intervals, there may be a delay between the occurrence of a debug event (such as a packet drop) and the display of that event. The delay may be as much as the configured interval plus 10 seconds.

Examples

The following is sample output from the **debug frame-relay switching** command:

```
Router# debug frame-relay switching interface s2/1 1000 interval 2
 Frame Relay switching debugging is on
 Display frame switching debug on interface Serial2/1 dlci 1000
 1d02h: Serial2/1 dlci 1000: 32 packets switched to Serial2/0 dlci 1002
 1d02h: Serial2/1 dlci 1000: 1800 packets output
 1d02h: Serial2/1 dlci 1000: 4 packets dropped - outgoing PVC inactive
 1d02h: Serial2/1 dlci 1000: Incoming PVC status changed to ACTIVE
 1d02h: Serial2/1 dlci 1000: Outgoing PVC status changed to ACTIVE
 1d02h: Serial2/1 dlci 1000: Incoming interface hardware module state changed to UP
 1d02h: Serial2/1 dlci 1000: Outgoing interface hardware module state changed to UP
```

debug frame-relay vc-bundle

To display information about the Frame Relay permanent virtual circuit (PVC) bundles that are configured on a router, use the **debug frame-relay vc-bundle** command in privileged EXEC mode. To stop the display, use the **no** form of this command.

debug frame-relay vc-bundle {**detail**| **state-change**} [*vc-bundle-name*]

no debug frame-relay vc-bundle {**detail**| **state-change**} [*vc-bundle-name*]

Syntax Description

detail	Displays detailed information about the members of the bundle specified by *vc-bundle-name*. Displays detailed information about the members of all PVC bundles if *vc-bundle-name* is not specified.
state-change	Displays information pertaining only to the state changes of the PVC bundle and PVC bundle members specified by *vc-bundle-name*. Displays state-change information for all PVC bundles and bundle members if *vc-bundle-name* is not specified.
vc-bundle-name	(Optional) Specifies a particular PVC bundle.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(13)T	This command was introduced.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.

Usage Guidelines

Use this command to monitor state changes and Inverse ARP activity for one or all of the PVC bundles and bundle members configured on a router.

Note Debugging messages that are prefixed with “FR_VCB” (instead of “FR-VCB”) indicate serious failures in the Frame Relay PVC bundle performance. Contact the Cisco Technical Assistance Center (TAC) if you see debugging messages with this prefix.

Examples

The following is sample output from the **debug frame-relay vc-bundle**commandthat shows Inverse ARP information for the PVC bundle. PVC bundle member 406 is the only PVC in the bundle to handle Inverse ARP packets. The Inverse ARP packets coming in on other bundle member PVCs are dropped.

```
Router# debug frame-relay vc-bundle
00:23:48:FR-VCB:MP-4-dynamic:inarp received on elected member 406
00:23:48:FR-VCB:MP-4-dynamic:installing dynamic map
00:23:48:FR-VCB:MP-4-dynamic:dropping inarp received on member 407
00:23:52:FR-VCB:MP-4-dynamic:sending inarp pkt on member 406
```

In the following example the PVC bundle goes down because the protected group goes down. All information about active transmission on each PVC is removed.

```
00:58:27:FR-VCB:MP-4-dynamic:member 402 state changed to DOWN
00:58:27:FR-VCB:MP-4-dynamic:protected group is DOWN
00:58:27:FR-VCB:MP-4-dynamic:state changed to DOWN
00:58:27:FR-VCB:MP-4-dynamic:active table reset
```

The following is sample output from the **debug frame-relay vc-bundle detail** command. State change and Inverse ARP activity is displayed for all PVC bundles and bundle members on the router.

```
Router# debug frame-relay adjacency vc-bundle detail
00:33:40: FR-VCB: MP-4-dynamic: member 404 state changed to UP
00:33:40: FR-VCB: MP-4-dynamic: active table update
00:33:40: FR-VCB: MP-3-static: sending inarp pkt on member 300
00:33:41: FR-VCB: MP-3-static: inarp received on elected member 300
00:33:48: FR-VCB: MP-3-static: inarp received on elected member 300
00:33:48: FR-VCB: MAIN-1-static: dropping inarp received on member 100
00:33:48: FR-VCB: MP-4-dynamic: dropping inarp received on member 404
00:33:48: FR-VCB: MP-4-dynamic: dropping inarp received on member 405
00:33:48: FR-VCB: P2P-5: dropping inarp received on member 507
00:33:48: FR-VCB: MP-3-static: dropping inarp received on member 303
00:33:48: FR-VCB: MAIN-2-dynamic: dropping inarp received on member 202
00:33:48: FR-VCB: MAIN-1-static: dropping inarp received on member 107
00:33:48: FR-VCB: MP-3-static: dropping inarp received on member 305
00:33:48: FR-VCB: MAIN-1-static: dropping inarp received on member 105
00:33:49: FR-VCB: P2P-5: dropping inarp received on member 505
00:33:49: FR-VCB: P2P-5: dropping inarp received on member 504
00:33:49: FR-VCB: P2P-5: dropping inarp received on member 503
00:33:49: FR-VCB: P2P-5: dropping inarp received on member 502
00:33:49: FR-VCB: P2P-5: dropping inarp received on member 501
```

Related Commands

Command	Description
debug frame-relay adjacency	Displays information pertaining to an adjacent node that has one or more Frame Relay PVC bundles.

debug frame-relay virtual

To display debugging messages for the virtual Frame Relay interface, use the **debug frame-relay virtual**command in privileged EXEC mode.

debug frame-relay virtual *destination interface*

Syntax Description

destination interface	Enables the debugging messages for that specific interface.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

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

Usage Guidelines

Use the **debug frame-relay virtual** command to display debugging messages for the virtual Frame Relay interface. The **debug frame-relay virtual** command produces output only when problems occur.

Examples

The following example shows the output if one of the routers has not been configured. This output occurs when the other end is trying to send the receiving box Frame Relay packets.

```
VFR: Radio1/0 has no VFR for 00:00:C068:6F:AA
```

Related Commands

Command	Description
frame-relay over radio	Links the virtual Frame Relay interface to the specified radio interface and destination MAC address.
interface virtual-framerelay	Defines the virtual interface and then associates the interface with a specific wireless connection.
show virtual-framerelay	Shows the output of the **interface virtual-frame relay** command.

debug fras error

To display information about Frame Relay access support (FRAS) protocol errors, use the **debug fras error**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug fras error

no debug fras error

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

For complete information on the FRAS process, use the **debug fras message** along with the **debug fras error** command.

Examples

The following is sample output from the **debug fras error**command. This example shows that no logical connection exists between the local station and remote station in the current setup.

```
Router# debug fras error
FRAS: No route, lmac 1000.5acc.7fb1 rmac 4fff.0000.0000, lSap=0x4, rSap=0x4
FRAS: Can not find the Setup
```

Related Commands

Command	Description
debug cls message	Displays information about CLS messages.
debug fras message	Displays general information about FRAS messages.
debug fras state	Displays information about FRAS data-link control state changes.

debug fras-host activation

To display the Logical Link Control, Type 2 (LLC2) session activation and deactivation frames (such as XID, SABME, DISC, UA) that are being handled by the Frame Relay access support (FRAS) host, use the **debug fras-host activation** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug fras-host activation

no debug fras-host activation

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

If many LLC2 sessions are being activated or deactivated at any time, this command may generate a substantial amount of output to the console.

Examples

The following is sample output from the **debug fras-host activation** command:

```
Router# debug fras-host activation
FRHOST: Snd      TST C to HOST, DA = 4001.3745.1088 SA = 400f.dddd.001e DSAP = 0x00 SSAP =
 0x04
FRHOST: Fwd  BNN   XID to HOST, DA = 4001.3745.1088 SA = 400f.dddd.001e DSAP = 0x04 SSAP =
 0x04
FRHOST: Fwd HOST   XID to  BNN, DA = 400f.dddd.001e SA = 4001.3745.1088 DSAP = 0x04 SSAP =
 0x05
FRHOST: Fwd  BNN   XID to HOST, DA = 4001.3745.1088 SA = 400f.dddd.001e DSAP = 0x04 SSAP =
 0x04
FRHOST: Fwd HOST SABME to  BNN, DA = 400f.dddd.001e SA = 4001.3745.1088 DSAP = 0x04 SSAP =
 0x04
FRHOST: Fwd  BNN    UA to HOST, DA = 4001.3745.1088 SA = 400f.dddd.001e DSAP = 0x04 SSAP =
 0x05
```

The first line indicates that the FRAS Host sent a TEST Command to the host. In the second line, the FRAS Host forwards an XID frame from a BNN device to the host. In the third line, the FRAS Host forwards an XID from the host to the BNN device.

The below table describes the significant fields shown in the display.

Table 29: debug fras-host activation Field Descriptions

Field	Description
DA	Destination MAC address of the frame.
SA	Source MAC address of the frame.
DSAP	Destination SAP of the frame.
SSAP	Source SAP of the frame.

debug fras-host error

To enable the Frame Relay access support (FRAS) Host to send error messages to the console, use the **debug fras-host error**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug fras-host error

no debug fras-host error

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Examples

The following is sample output from the **debug fras-host error**command when the I-field in a TEST Response frame from a host does not match the I-field of the TEST Command sent by the FRAS Host:

```
Router# debug fras-host error
FRHOST: SRB TST R Protocol Violation - LLC I-field not maintained.
```

debug fras-host packet

To see which Logical Link Control, type 2 (LLC2) session frames are being handled by the Frame Relay access support (FRAS) Host, use the **debug fras-host packet**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug fras-host packet

no debug fras-host packet

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

Caution Use this command with great care. If many LLC2 sessions are active and passing data, this command may generate a substantial amount of output to the console and impact device performance.

Examples

The following is sample output from the **debug fras-host packet** command:

```
Router# debug fras-host packet
FRHOST: Snd      TST C to HOST, DA = 4001.3745.1088 SA = 400f.dddd.001e DSAP = 0x00 SSAP =
 0x04
FRHOST: Fwd  BNN   XID to HOST, DA = 4001.3745.1088 SA = 400f.dddd.001e DSAP = 0x04 SSAP =
 0x04
FRHOST: Fwd HOST   XID to  BNN, DA = 400f.dddd.001e SA = 4001.3745.1088 DSAP = 0x04 SSAP =
 0x05
FRHOST: Fwd  BNN   XID to HOST, DA = 4001.3745.1088 SA = 400f.dddd.001e DSAP = 0x04 SSAP =
 0x04
FRHOST: Fwd HOST SABME to  BNN, DA = 400f.dddd.001e SA = 4001.3745.1088 DSAP = 0x04 SSAP =
 0x04
FRHOST: Fwd  BNN    UA to HOST, DA = 4001.3745.1088 SA = 400f.dddd.001e DSAP = 0x04 SSAP =
 0x05
FRHOST: Fwd HOST LLC-2 to  BNN, DA = 400f.dddd.001e SA = 4001.3745.1088 DSAP = 0x04 SSAP =
 0x04
FRHOST: Fwd  BNN LLC-2 to HOST, DA = 4001.3745.1088 SA = 400f.dddd.001e DSAP = 0x04 SSAP =
 0x05
FRHOST: Fwd HOST LLC-2 to  BNN, DA = 400f.dddd.001e SA = 4001.3745.1088 DSAP = 0x04 SSAP =
 0x04
FRHOST: Fwd  BNN LLC-2 to HOST, DA = 4001.3745.1088 SA = 400f.dddd.001e DSAP = 0x04 SSAP =
 0x04
```

The **debug fras-host packet** output contains all of the output from the **debug fras-host activation** command and additional information. The first six lines of this sample display are the same as the output from the **debug fras-host activation**command. The last lines show LLC-2 frames being sent between the Frame Relay Boundary Network Node (BNN) device and the host.

The below table describes the significant fields shown in the display.

Table 30: debug fras-host packet Field Descriptions

Field	Description
DA	Destination MAC address of the frame.
SA	Source MAC address of the frame.
DSAP	Destination service access point (SAP) of the frame.
SSAP	Source SAP of the frame.

debug fras-host snmp

To display messages to the console describing Simple Network Management Protocol (SNMP) requests to the Frame Relay access support (FRAS) Host MIB, use the **debug fras-host snmp**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug fras-host snmp

no debug fras-host snmp

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

Use of this command may result in a substantial amount of output to the screen. Only use this command for problem determination.

Examples

The following is sample output from the **debug fras-host snmp** command. In this example, the MIB variable k_frasHostConnEntry_get() is providing SNMP information for the FRAS host.

```
Router# debug fras-host snmp
k_frasHostConnEntry_get(): serNum = -1, vRingIfIdx = 31, frIfIdx = 12
Hmac = 4001.3745.1088, frLocSap = 4, Rmac = 400f.dddd.001e, frRemSap = 4
```

The below table describes the significant fields shown in the display.

Table 31: debug fras-host snmp Field Descriptions

Field	Description
serNum	Serial number of the SNMP request.
vRingIfIdx	Interface index of a virtual Token Ring.
frIfIdx	Interface index of a Frame Relay serial interface.
Hmac	MAC address associated with the host for this connection.
frLocSap	SAP associated with the host for this connection.
Rmac	MAC address associated with the FRAD for this connection.
frRemSap	LLC 2 SAP associated with the FRAD for this connection.

debug fras message

To display general information about Frame Relay access support (FRAS) messages, use the **debug fras message**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug fras message

no debug fras message

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

For complete information on the FRAS process, use the **debug fras error** command along with the **debug fras message** command.

Examples

The following is sample output from the **debug fras message**command. This example shows incoming Cisco Link Services (CLS) primitives.

```
Router# debug fras message
FRAS: receive 4C23
FRAS: receive CC09
```

Related Commands

Command	Description
debug cls message	Limits output for some debugging commands based on the interfaces.
debug fras error	Displays information about FRAS protocol errors.
debug fras state	Displays information about FRAS data-link control state changes.

debug fras state

To display information about Frame Relay access support (FRAS) data-link control link-state changes, use the **debug fras state**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug fras state

no debug fras state

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Examples

The following is sample output from the **debug fras state**command. This example shows the state changing from a *request open station is sent* state to an *exchange XID* state.

Possible states are the following: reset, request open station is sent, exchange xid, connection request is sent, signal station wait, connection response wait, connection response sent, connection established, disconnect wait, and number of link states.

```
Router# debug fras state
FRAS: TR0 (04/04) oldstate=LS_RQOPNSTNSENT, input=RQ_OPNSTN_CNF
FRAS: newstate=LS_EXCHGXID
```

Related Commands

Command	Description
debug cls message	Limits output for some debug commands based on the interfaces.
debug fras error	Displays information about FRAS protocol errors.
debug fras message	Displays general information about FRAS messages.

debug ftpserver

To display information about the FTP server process, use the **debug ftpserver**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ftpserver

no debug ftpserver

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Examples

The following is sample output from the **debug ftpserver**command:

```
Router# debug ftpserver
Mar  3 10:21:10: %FTPSERVER-6-NEWCONN: FTP Server - new connection made.
-Process= "TCP/FTP Server", ipl= 0, pid= 53
Mar  3 10:21:10: FTPSRV_DEBUG:FTP Server file path: 'disk0:'
Mar  3 10:21:10: FTPSRV_DEBUG:(REPLY)  220
Mar  3 10:21:10: FTPSRV_DEBUG:FTProuter IOS-FTP server (version 1.00) ready.
Mar  3 10:21:10: FTPSRV_DEBUG:FTP Server Command received: 'USER aa'
Mar  3 10:21:20: FTPSRV_DEBUG:(REPLY)  331
Mar  3 10:21:20: FTPSRV_DEBUG:Password required for 'aa'.
Mar  3 10:21:20: FTPSRV_DEBUG:FTP Server Command received: 'PASS aa'
Mar  3 10:21:21: FTPSRV_DEBUG:(REPLY)  230
Mar  3 10:21:21: FTPSRV_DEBUG:Logged in.
Mar  3 10:21:21: FTPSRV_DEBUG:FTP Server Command received: 'SYST'
Mar  3 10:21:21: FTPSRV_DEBUG:(REPLY)  215
Mar  3 10:21:21: FTPSRV_DEBUG:Cisco IOS Type: L8 Version: IOS/FTP 1.00
Mar  3 10:21:21: FTPSRV_DEBUG:FTP Server Command received: 'PWD'
Mar  3 10:21:35: FTPSRV_DEBUG:(REPLY)  257
Mar  3 10:21:39: FTPSRV_DEBUG:FTP Server Command received: 'CWD disk0:/syslogd.d'r/'
Mar  3 10:21:45: FTPSRV_DEBUG:FTP Server file path: 'disk0:/syslogd.dir'
Mar  3 10:21:45: FTPSRV_DEBUG:(REPLY)  250
Mar  3 10:21:45: FTPSRV_DEBUG:CWD command successful.
Mar  3 10:21:45: FTPSRV_DEBUG:FTP Server Command received: 'PORT 171,69,30,20,22',32
Mar  3 10:21:46: FTPSRV_DEBUG:(REPLY)  200
Mar  3 10:21:46: FTPSRV_DEBUG:PORT command successful.
Mar  3 10:21:46: FTPSRV_DEBUG:FTP Server Command received: 'LIST'
Mar  3 10:21:47: FTPSRV_DEBUG:FTP Server file path: 'disk0:/syslogd.dir/.'
Mar  3 10:21:47: FTPSRV_DEBUG:(REPLY)  220
Mar  3 10:23:11: FTPSRV_DEBUG:Opening ASCII mode data connection for file list.
Mar  3 10:23:11: FTPSRV_DEBUG:(REPLY)  226
Mar  3 10:23:12: FTPSRV_DEBUG:Transfer complete.
Mar  3 10:23:12: FTPSRV_DEBUG:FTP Server Command received: 'TYPE I'
Mar  3 10:23:14: FTPSRV_DEBUG:(REPLY)  200
Mar  3 10:23:14: FTPSRV_DEBUG:Type set to I.
Mar  3 10:23:14: FTPSRV_DEBUG:FTP Server Command received: 'PORT 171,69,30,20,22',51
Mar  3 10:23:20: FTPSRV_DEBUG:(REPLY)  200
Mar  3 10:23:20: FTPSRV_DEBUG:PORT command successful.
Mar  3 10:23:20: FTPSRV_DEBUG:FTP Server Command received: 'RETR syslogd.1'
Mar  3 10:23:21: FTPSRV_DEBUG:FTP Server file path: 'disk0:/syslogd.dir/syslogd.1'
Mar  3 10:23:21: FTPSRV_DEBUG:FTPSERVER: Input path passed Top-dir(disk0:/syslogd.dir/)
test.
Mar  3 10:23:21: FTPSRV_DEBUG:(REPLY)  150
Mar  3 10:23:21: FTPSRV_DEBUG:Opening BINARY mode data connection for syslogd.1 (607317
bytes).
Mar  3 10:23:21: FTPSRV_DEBUG:(REPLY)  226
Mar  3 10:23:29: FTPSRV_DEBUG:Transfer complete.
```

The sample output corresponds to the following FTP client session. In this example, the user connects to the FTP server, views the contents of the top-level directory, and gets a file.

```
FTPclient% ftp
FTProuter
Connected to FTProuter.cisco.com.
220 FTProuter IOS-FTP server (version 1.00) ready.
Name (FTProuter:me): aa
331 Password required for 'aa'.
Password:
230 Logged in.
Remote system type is Cisco.
ftp> pwd
257 "disk0:/syslogd.dir/" is current directory.
ftp> dir
200 PORT command successful.
150 Opening ASCII mode data connection for file list.
syslogd.1
syslogd.2
syslogd.3
syslogd.4
syslogd.5
syslogd.6
syslogd.7
syslogd.8
syslogd.9
syslogd.cur
226 Transfer complete.
ftp> bin
200 Type set to I.
ftp> get syslogd.1
200 PORT command successful.
150 Opening BINARY mode data connection for syslogd.1 (607317 bytes).
226 Transfer complete.
607317 bytes received in 7.7 seconds (77 Kbytes/s)
ftp>
```

The following **debug ftpserver** command output indicates that no top-level directory is specified. Therefore, the client cannot access any location on the FTP server. Use the **ftp-server topdir** command to specify the top-level directory.

```
Mar  3 10:29:14: FTPSRV_DEBUG:(REPLY)  550
Mar  3 10:29:14: FTPSRV_DEBUG:Access denied to 'disk0:'
```

debug gatekeeper gup

To display the Gatekeeper Update Protocol (GUP) events or Abstract Syntax Notation 1 (ASN.1) details, use the **debug gatekeeper gup**command inprivileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug gatekeeper gup {**events**| **asn1**}

no debug gatekeeper gup {**events**| **asn1**}

Syntax Description

events	Displays a message whenever a GUP announcement is sent or received. GUP is the protocol used between individual gatekeepers in a cluster, which keeps all the gatekeepers synchronized with all endpoints registered on the cluster.
asn1	ASN.1 library. ASN.1 is an International Telecommunication Union (ITU) standard for protocol syntax and message encoding. Entering this keyword causes a packet dump of all GUP announcement messages.

Command Default

Debugging is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.1(5)XM	This command was introduced.
12.2(2)T	This command was integrated into Cisco IOS Release 12.2(2)T.
12.2(2)XB1	This command was implemented on the Cisco AS5850 universal gateway.

Examples

The following example shows how to enable a packet dump of all GUP announcement messages:

```
Router# debug gatekeeper gup asn1
00:10:21:ENCODE BUFFER::= 00 0A2A8648 86F70C0A 00000120 001E8001
86A08001 86A00547 656E6576 614E0000 00000142 80004700 65006E00 65007600
61080050 00610072 00690073 00000000 0000
00:10:21:
00:10:21:PDU ::=
value GUP_Information ::=
```

```
protocolIdentifier { 1 2 840 113548 10 0 0 1 }
message announcementIndication :
announcementInterval 30
endpointCapacity 100000
callCapacity 100000
hostName '47656E657661'H
percentMemory 39
percentCPU 0
currentCalls 0
currentEndpoints 0
zoneInformation
gatekeeperIdentifier {"Geneva"}
altGKIdentifier {"Paris"}
totalBandwidth 0
interzoneBandwidth 0
remoteBandwidth 0
RAW_BUFFER::=
00 0A2A8648 86F70C0A 00000120 001E800B 858A8001 86A00144 80007400 6F007200 6E006100 64006F00
 2D006700 6B120063 00790063 006C006F 006E0065 002D0067 006B0000 00000000
*Mar  3 15:40:31:
*Mar  3 15:40:31:Sending GUP ANNOUNCEMENT INDICATION to 172.18.195.140RAW_BUFFER::=
00 0A2A8648 86F70C0A 00000120 001E800A EF8A8001 86A00144 80006300 79006300 6C006F00 6E006500
 2D006700 6B120074 006F0072 006E0061 0064006F 002D0067 006B0000 00000000
*Mar  3 15:40:31:PDU DATA = 60EAB248
value GUP_Information ::=
protocolIdentifier { 1 2 840 113548 10 0 0 1 }
message announcementIndication :
{
announcementInterval 30
endpointCapacity 716682
callCapacity 100000
zoneInformation
gatekeeperIdentifier {"cyclone-gk"}
altGKIdentifier {"tornado-gk"}
totalBandwidth 0
interzoneBandwidth 0
remoteBandwidth 0
Mar  3 15:40:31:Received GUP ANNOUNCEMENT INDICATION from 172.18.195.140
u all
All possible debugging has been turned off
```

Related Commands

Command	Description
load-balance	Configures load balancing.

debug gatekeeper load

To display gatekeeper load-balancing debug events, use the **debug gatekeeper load**command inprivileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug gatekeeper load events

no debug gatekeeper load events

Syntax Description

events	Displays a message whenever a load-balancing message is sent or received.

Command Default

Debugging is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.1(5)XM	This command was introduced.
12.2(2)T	This command was integrated into Cisco IOS Release 12.2(2)T.
12.2(2)XB1	This command was implemented on the Cisco AS5850 universal gateway.

Examples

The following is sample output for the **debug gatekeeper load** command.

Note The following output examples are independent of each other and would not ordinarily be seen at the same time.

```
Router# debug gatekeeper load

Router#
Router# s
how debugging
gk load-balancing debug level = Events
Router#
gk_load_overloaded:Overloaded, 5-second CPU utilization too high
gk_load_overloaded:Overloaded due to excessive calls/endpoints
gk_load_balance_endpt_request:load balance occurred. New load_balance_count=2
```

Related Commands

Command	Description
load-balance	Configures load balancing.

debug gatekeeper server

To trace all the message exchanges between the Cisco IOS Gatekeeper and the external applications, use the **debug gatekeeper server** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug gatekeeper server

no debug gatekeeper server

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.1(1)T	This command was introduced.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.

Usage Guidelines

Use this command to see information about a Gatekeeper server. This command shows any errors that occur in sending messages to the external applications or in parsing messages from the external applications.

Examples

The following example shows debugging information about a Gatekeeper server:

```
Router# debug gatekeeper servers

Router# show debug
  Gatekeeper:
    Gatekeeper Server Messages debugging is on
```

To turn the Gatekeeper server debugging message off, see the following examples:

```
Router# no debug all

Router# no debug gatekeeper servers
```

Related Commands

Command	Description
show gatekeeper server	Displays information about the Gatekeeper servers configured on your network by ID.

debug ggsn quota-server

To display debug information related to quota server processing on the GGSN, use the **debug ggsn quota-server**privilege EXEC command.

debug ggsn quota-server [**detail**| **packets [dump]**| **events**| **parsing**| **errors**]

Syntax Description

detail	Displays extended details about quota server operations on the GGSN.
packets	Displays packets sent between the quota server process on the GGSN and the CSG. Optionally, displays output in hexadecimal notation.
events	Displays events related to quota server processing on the GGSN.
parsing	Displays details about GTP TLV parsing between the quota server and the Content Services Gateway.
errors	Displays errors related to quota server processing on the GGSN.

Command Default

No default behavior or values.

Command Modes

Privilege EXEC (#)

Command History

Release	Modification
12.3(14)YQ	This command was introduced.
12.4(9)T	This command was integrated into Cisco IOS Release 12.4(9)T.

Usage Guidelines

This command is useful for system operators and development engineers if problems are encountered with communication between the GGSN quota server process and the CSG.

Examples

The following example enables the display of detailed quota server processing debug output--pre-allocated quota and quota push:

```
Router#debug ggsn quota-server detail
ggsn quota-server details debugging is on
```

```
Router#
Jun  2 02:40:39.391: GGSN-QS:Encoding QUOTA PUSH REQUEST
Jun  2 02:40:39.391: GGSN-QS:Adding TLV USER_INDEX
Jun  2 02:40:39.391: GGSN-QS:  IP Address: 3.3.3.1  User ID: 12345
Jun  2 02:40:39.391: GGSN-QS:Adding TLV SERVICE_ID: 1
Jun  2 02:40:39.391: GGSN-QS:Adding TLV QUADRANS_GRANTED
Jun  2 02:40:39.391: GGSN-QS:  Quadrans: 1250  Threshold: 1000  Units: SECONDS
Jun  2 02:40:39.391: GGSN-QS:Adding TLV QUADRANS_GRANTED
Jun  2 02:40:39.391: GGSN-QS:  Quadrans: 5000  Threshold: 5000  Units: BYTES_IP
Jun  2 02:40:39.391: GGSN-QS:Adding TLV TIMEOUT: 50000
Jun  2 02:40:39.391: GGSN-QS:Adding TLV TARIFF_TIME: 1147698000
Jun  2 02:40:39.391: GGSN-QS:Sending QUOTA_PUSH_REQ from QS (4.4.4.4:3386) to CSG
(30.1.1.1:3386)
Jun  2 02:40:39.395: pak=0x6523B5B0, datagramstart=0x200143D8, network_start=0x200143BC
datagramsize 91
Jun  2 02:40:39.395: GGSN-QS msgtype 0xF0, seq 1, len 85, from 4.4.4.4:3386 to 30.1.1.1:3386
200143D0:                    0FF00055 00017E01          .p.U..~.
200debug ggsn quota-server detail143E0: FC005001 31000000 4A002E00 46001400  |.P.1...J...F...
200143F0: 09030303 01313233 34350015 00013100  .....12345....1.
20014400: 2D000E00 00000000 0004E201 03000003  -.........b.....
20014410: E8002D00 0E000000 00000013 88020300  h.-.............
20014420: 00138800 17000400 00C35000 4D000444  .........CP.M..D
20014430: 687B50                               h{P
Jun  2 02:40:39.395: GGSN-QS:Received Data Record Transfer Response from (30.1.1.1:3386)
Sequence number 1
Jun  2 02:40:39.395: GGSN-QS:Cause = 128
Jun  2 02:40:39.395: GGSN-QS:Request Responded Sequence Number = 1
Jun  2 02:40:39.395: GGSN-QS:Private Ext IE length 32 QM Rsp length 29
Jun  2 02:40:39.395: GGSN-QS:Received message QUOTA_PUSH_RESP from CSG
Jun  2 02:40:39.395: GGSN-QS:UserIndex TLV: IP Address 3.3.3.1 UserName/MSISDN 12345
Jun  2 02:40:39.395: GGSN-QS:Session ID TLV: 1736898353
Jun  2 02:40:39.395: GGSN-QS:Service ID TLV: 1
Jun  2 02:40:39.399: GGSN-QS:Detected real CSG 30.1.1.1 for virtual CSG 30.1.1.1
Jun  2 02:40:39.399: GGSN-QS:real CSG newly detected
ggsn quota-server details debugging is on
Router#
```

debug glbp errors

To display debugging messages about Gateway Load Balancing Protocol (GLBP) error conditions, use the **debug glbp errors** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug glbp errors

no debug glbp errors

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(14)S	This command was introduced.
12.2(15)T	This command was integrated into Cisco IOS Release 12.2(15)T.
12.2(17b)SXA	This command was integrated into Cisco IOS Release 12.2(17b)SXA.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Examples

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

```
Router# debug glbp errors
GLBP Errors debugging is on
1d19h: GLBP: Fa0/0 API active virtual address 10.21.8.32 not found
1d19h: GLBP: Fa0/0 API active virtual address 10.21.8.32 not found
1d19h: GLBP: Fa0/0 API active virtual address 10.21.8.32 not found
```

Related Commands

Command	Description
debug condition glbp	Displays debugging messages about GLBP that match specific conditions.

debug glbp events

To display debugging messages about Gateway Load Balancing Protocol (GLBP) events that are occurring, use the **debug glbp events** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug glbp events [**all**| **api**| **cache**| **detail**| **ha**| **icmp**| **protocol**| **redundancy**| **terse**| **track**]

no debug glbp events [**all**| **api**| **cache**| **detail**| **ha**| **icmp**| **protocol**| **redundancy**| **terse**| **track**]

Syntax Description

all	(Optional) Displays all debugging output about GLBP events.
api	(Optional) Displays GLBP API events.
cache	(Optional) Displays GLBP client cache events.
detail	(Optional) Displays detailed debugging output about GLBP events.
ha	(Optional) Displays GLBP high-availability (HA) events.
icmp	(Optional) Displays GLBP Internet Control Message Protocol (ICMP) events.
protocol	(Optional) Displays GLBP protocol events.
redundancy	(Optional) Displays GLBP redundancy events.
terse	(Optional) Displays a limited range of debugging output about GLBP events.
track	(Optional) Displays GLBP tracking events.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(14)S	This command was introduced.
12.2(15)T	This command was integrated into Cisco IOS Release 12.2(15)T.
12.2(17b)SXA	This command was integrated into Cisco IOS Release 12.2(17b)SXA.

Release	Modification
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(31)SB2	This command was enhanced to display information about GLBP support of Stateful Switchover (SSO). The **ha** keyword was added.
12.4(15)T	The **cache** keyword was added.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.

Examples

The following is sample output from the **debug glbp events**command when the **terse** keyword is specified:

```
Router# debug glbp events terse
GLBP Events debugging is on
    (protocol, redundancy, track)
```

The following is sample output from the **debug glbp events**command on an active RP displaying an interface shutdown event:

```
Router# debug glbp events
GLBP Events debugging is on
*Sep 15 09:14:53.583: GLBP: Et0/0 API Software interface going down
*Sep 15 09:14:53.583: GLBP: Et0/0 API Software interface going down
*Sep 15 09:14:53.583: GLBP: Et0/0 Interface down
*Sep 15 09:14:53.583: GLBP: Et0/0 1.1 Listen: e/Forwarder disabled
*Sep 15 09:14:53.583: GLBP: Et0/0 1.1 Listen -> Init
*Sep 15 09:14:53.583: GLBP: Et0/0 Fwd 1.1 HA Encoded (state Init) into sync buffer
*Sep 15 09:14:53.583: GLBP: Et0/0 1.2 Active: e/Forwarder disabled
*Sep 15 09:14:53.583: GLBP: Et0/0 1.2 Active -> Init
*Sep 15 09:14:53.583: %GLBP-6-FWDSTATECHANGE: Ethernet0/0 Grp 1 Fwd 2 state Active -> Init

*Sep 15 09:14:53.583: GLBP: Et0/0 Fwd 1.2 HA Encoded (state Init) into sync buffer
*Sep 15 09:14:53.583: GLBP: Et0/0 1 Standby: e/GLBP disabled
*Sep 15 09:14:53.583: GLBP: Et0/0 1 Active router IP is unknown, was 172.24.1.2
*Sep 15 09:14:53.583: GLBP: Et0/0 1 Standby router is unknown, was local
*Sep 15 09:14:53.583: GLBP: Et0/0 1 Standby -> Init
*Sep 15 09:14:53.583: GLBP: Et0/0 Grp 1 HA Encoded (state Init) into sync buffer
*Sep 15 09:14:55.583: %LINK-5-CHANGED: Interface Ethernet0/0, changed state to
administratively down
*Sep 15 09:14:55.587: GLBP: API Hardware state change
*Sep 15 09:14:56.595: %LINEPROTO-5-UPDOWN: Line protocol on Interface Ethernet0/0, changed
 state to down
```

The following is sample output from the **debug glbp events**command on a standby RP displaying an interface shutdown event:

```
RouterRP-standby# debug glbp events
GLBP Events debugging is on
.
.
.
*Sep 15 09:14:53.691: GLBP: Et0/0 Fwd 1.1 HA sync, state Listen -> Init
*Sep 15 09:14:53.691: GLBP: Et0/0 Fwd 1.2 HA sync, state Active -> Init
*Sep 15 09:14:53.691: GLBP: Et0/0 Grp 1 HA sync, state Standby -> Init
```

The following is sample output from the **debug glbp events**command when the **cache** keyword is specified:

```
Router# debug glbp events cache
```

```
GLBP Events debugging is on (cache)
Jun 30 08:57:50.171: GLBP: Et0/0 1 Added client cache entry for 7bcf.e03d.d3bd
Jun 30 08:57:50.171: GLBP: Et0/0 1 Added client cache entry for c5e8.46eb.8a86
Jun 30 08:57:50.171: GLBP: Et0/0 1 Added client cache entry for 69e5.9d95.0f7e
Jun 30 08:57:50.171: GLBP: Et0/0 1 Added client cache entry for 986e.d98a.1607
Jun 30 08:57:50.171: GLBP: Et0/0 1 Added client cache entry for 1843.ee62.f62e
Jun 30 08:57:50.171: GLBP: Et0/0 1 Added client cache entry for 5f4c.cfc4.5dc1
```

Related Commands

Command	Description
debug condition glbp	Displays debugging messages about GLBP that match specific conditions.

debug glbp packets

To display summary information about Gateway Load Balancing Protocol (GLBP) packets being sent or received, use the **debug glbp packets** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug glbp packets [**all**| **detail**| **hello**| **reply**| **request**| **terse**]

no debug glbp packets [**all**| **detail**| **hello**| **reply**| **request**| **terse**]

Syntax Description

all	(Optional) Displays all debugging output about GLBP packets.
detail	(Optional) Displays detailed debugging output about GLBP packets.
hello	(Optional) Displays debugging output about GLBP hello packets.
reply	(Optional) Displays debugging output about GLBP reply packets.
request	(Optional) Displays debugging output about GLBP request packets.
terse	(Optional) Displays a limited range of debugging output about GLBP packets.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(14)S	This command was introduced.
12.2(15)T	This command was integrated into Cisco IOS Release 12.2(15)T.
12.2(17b)SXA	This command was integrated into Cisco IOS Release 12.2(17b)SXA.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Examples

The following sample output from the **debug glbp packets**command shows debugging output about GLBP hello packets:

```
Router# debug glbp packets hello
GLBP Packets debugging is on
    (Hello)
1d19h: GLBP: Fa0/0 Grp 10 Hello  out 10.21.8.32 VG Active  pri 254 vIP 10.21.8.10 1
1d19h: GLBP: Fa0/0 Grp 10 Hello  out 10.21.8.32 VG Active  pri 254 vIP 10.21.8.10 1
1d19h: GLBP: Fa0/0 Grp 10 Hello  out 10.21.8.32 VG Active  pri 254 vIP 10.21.8.10 1
1d19h: GLBP: Fa0/0 Grp 10 Hello  out 10.21.8.32 VG Active  pri 254 vIP 10.21.8.10 1
```

Related Commands

Command	Description
debug condition glbp	Displays debugging messages about GLBP that match specific conditions.

debug glbp terse

To display a limited range of debugging messages about Gateway Load Balancing Protocol (GLBP) errors, events, and packets, use the **debug glbp terse** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug glbp terse

no debug glbp terse

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(14)S	This command was introduced.
12.2(15)T	This command was integrated into Cisco IOS Release 12.2(15)T.
12.2(17b)SXA	This command was integrated into Cisco IOS Release 12.2(17b)SXA.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.

Examples

The following is sample output from the **debug glbp terse**command:

```
Router# debug glbp terse
GLBP:
  GLBP Errors debugging is on
  GLBP Events debugging is on
    (protocol, redundancy, track)
  GLBP Packets debugging is on
    (Request, Reply)
```

Related Commands

Command	Description
debug condition glbp	Displays debugging messages about GLBP that match specific conditions.
debug glbp errors	Displays debugging messages about GLBP errors.
debug glbp events	Displays debugging messages about GLBP events.

Command	Description
debug glbp packets	Displays debugging messages about GLBP packets.

debug gprs category fsm event

To display debug information related to service-aware GGSN category events, and state transactions, use the **debug gprs category fsm event**privilege EXEC command.

debug gprs category fsm event

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values.

Command Modes

Privilege EXEC (#)

Command History

Release	Modification
12.3(14)YQ	This command was introduced.
12.4(9)T	This command was integrated into Cisco IOS Release 12.4(9)T.

Usage Guidelines

This command is useful for system operators and development engineers if problems are encountered with eGGSN processing.

Examples

Examples

The following example enables the display of eGGSN events and state transactions--pre-allocated quota. This is PDP context create, prepaid user data transfer, and then context teardown.

PDP Context Create:

```
Router#debug gprs category fsm event
eGGSN category fsm event debugging is on
Router#
Jun  2 02:55:08.491: GPRS:1234050000000010:created service-aware subblock
Jun  2 02:55:11.383: GPRS:1234050000000010:it is the only one PDP of the user, need CCR msg
Jun  2 02:55:11.383: GPRS:1234050000000010:sent ccr_init
Jun  2 02:55:11.823: GPRS:1234050000000010:create new category 1
Jun  2 02:55:11.823: GPRS:1234050000000010:shdb 0xFB00001C created for category 1 (handle
0x8C000007)
Jun  2 02:55:11.823: GPRS:1234050000000010:successfully create a category
Jun  2 02:55:14.623: GPRS:1234050000000010:created sync_object for CREATE_PDP
Jun  2 02:55:14.623: GPRS:1234050000000010:get 1 impacted categories into sync_object for
CREATE_PDP
Jun  2 02:55:14.623: GPRS:1234050000000010:insert category 1 from sync_object for CREATE_PDP
Jun  2 02:55:14.623: GPRS:1234050000000010:number of really impacted by CREATE_PDP = 1
Jun  2 02:55:14.623: GPRS:1234050000000010:FSM_ggsn_rcvd_quota
Jun  2 02:55:14.623: GPRS:1234050000000010:category 1 trans from INIT to PENDING QP on event
 CCA_QUOTA
Jun  2 02:55:14.627: GPRS:1234050000000010:FSM_ggsn_rcvd_qp_ack_in_qp
Jun  2 02:55:14.627: GPRS:1234050000000010:remove category 1 from sync_object for CREATE_PDP
```

```
 0 still pending in the sync_object
Jun  2 02:55:14.627: GPRS:1234050000000010:send Create PDP Context Res to SGSN
Jun  2 02:55:14.627: GPRS:1234050000000010:delete sync object for CREATE_PDP, it has 0
categories
Jun  2 02:55:14.627: GPRS:1234050000000010:category 1 trans from PENDING QP to AUTHORIZED
on event CSG_QP_ACK
Router#
Router#
```

PDP Context Delete:

```
Router#
Jun  2 02:55:31.455: GPRS:1234050000000010:look up category by 1 found 65EEB128
Jun  2 02:55:31.455: GPRS:1234050000000010:FSM_ggsn_rcvd_stop
Jun  2 02:55:31.455: GPRS:category 1 report usage queue size = 2
Jun  2 02:55:31.455: GPRS:1234050000000010:usage unit has total_octets 0
Jun  2 02:55:31.455: GPRS:1234050000000010:usage unit has total_octets 300
Jun  2 02:55:31.455: GPRS:1234050000000010:category 1 , usage 6615E470
Jun  2 02:55:31.455: GPRS:1234050000000010:no sync_object for service stop
Jun  2 02:55:31.455: %GPRSFLTMG-4-CHARGING: GSN: 0.0.0.0, TID: 0000000000000000, APN: NULL,
 Reason: 1, unexpected CSG usage report cause
Jun  2 02:55:31.455: GPRS:1234050000000010:send CCR_UPDATE to DCCA server return ok
Jun  2 02:55:31.455: GPRS:releasing 2 usages in category
Jun  2 02:55:31.455: GPRS:release_usage_parameter
Jun  2 02:55:31.455: GPRS:1234050000000010:category 1 trans from AUTHORIZED to IDLE on event
 CSG_SERVICE_STOP
Jun  2 02:55:34.939: GPRS:1234050000000010:eggsn_get_final_usage_report
Jun  2 02:55:34.939: GPRS:1234050000000010:freeing all categories
Jun  2 02:55:34.939: GPRS:1234050000000010:delete_category 1
Jun  2 02:55:34.939: GPRS:1234050000000010:freeing service-aware subblock
Router#
```

Example 2--PDPs without Pre-Allocated Quota

The following example enables the display of eGGSN events and state transactions--for PDPs without pre-allocated quota.

PDP Context Create:

```
Router#debug gprs category fsm event
eGGSN category fsm event debugging is on
Router#
Jun  2 02:58:45.727: GPRS:1234050000000010:created service-aware subblock
Jun  2 02:58:48.623: GPRS:1234050000000010:it is the only one PDP of the user, need CCR msg
Jun  2 02:58:48.623: GPRS:1234050000000010:sent ccr_init
Router#
```

PDP Context Delete:

```
Router#
Jun  2 02:59:06.975: GPRS:1234050000000010:eggsn_get_final_usage_report
Jun  2 02:59:06.975: GPRS:1234050000000010:freeing all categories
Jun  2 02:59:06.975: GPRS:1234050000000010:freeing service-aware subblock
Router
```

debug gprs charging

To display information about general packet radio service (GPRS) charging functions on the gateway GPRS support node (GGSN), use the **debug gprs charging** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug gprs charging {**events**| **packets**}

no debug gprs charging {**events**| **packets**}

Syntax Description

events	Displays events related to GPRS charging processing on the GGSN.
packets	Displays GPRS charging packets that are sent between the GGSN and the charging gateway.

Command Default

No default behavior or values

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.1(1)GA	This command was introduced.
12.1(3)T	This command was integrated into Cisco IOS Release 12.1(3)T.
12.2(4)MX	This command was integrated into Cisco IOS Release 12.2(4)MX.
12.2(8)YD	This command was integrated into Cisco IOS Release 12.2(8)YD.
12.2(8)B	This command was integrated into Cisco IOS Release 12.2(8)B.
12.2(8)YY	This command was integrated into Cisco IOS Release 12.2(8)YY.
12.3(4)T	This command was integrated into Cisco IOS Release 12.3(4)T.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.

Usage Guidelines

This command is useful for system operators if problems are encountered with GPRS charging functions.

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

Examples

The following example enables the display of events related to GPRS charging events on the GGSN:

```
Router# debug gprs charging events
```

The following example enables the display of GPRS charging packets sent between the GGSN and the charging gateway:

```
Router# debug gprs charging events
```

debug gprs dcca

To display troubleshooting information about Diameter Credit Control Application (DCCA) processing on the gateway GPRS support node (GGSN), use the **debug gprs dcca**privilege EXEC command.

debug gprs dcca

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values.

Command Modes

Privilege EXEC (#)

Command History

Release	Modification
12.3(14)YQ	This command was introduced.
12.4(9)T	This command was integrated into Cisco IOS Release 12.4(9)T.

Usage Guidelines

This command is useful for system operators and development engineers if Diameter protocol problems are encountered on the GGSN.

Examples

Examples

The following is a sample of DCCA debug information with pre-allocated quota.

```
Router#debug gprs dcca
Router#
Jun  2 03:13:45.827: GPRS:1234050000000010:GPRS:DCCA: 3GPP-IMSI : 214350000000000
Jun  2 03:13:45.831: GPRS:1234050000000010:GPRS:DCCA: 3GPP-Charging-Id : 613053186
Jun  2 03:13:45.831: GPRS:1234050000000010:GPRS:DCCA: 3GPP-PDP-Type : 0
Jun  2 03:13:45.831: GPRS:1234050000000010:GPRS:DCCA: 3GPP-CG-Address : 20.1.1.1
Jun  2 03:13:45.831: GPRS:1234050000000010:GPRS:DCCA: 3GPP-QoS-Profile :
99-0911012964FFFF1100FFFF
Jun  2 03:13:45.831: GPRS:1234050000000010:GPRS:DCCA: 3GPP-SGSN-Address : 11.20.1.1
Jun  2 03:13:45.831: GPRS:1234050000000010:GPRS:DCCA: 3GPP-GGSN-Address : 10.20.61.1
Jun  2 03:13:45.831: GPRS:1234050000000010:GPRS:DCCA: 3GPP-IMSI-MCC-MNC : 21435
Jun  2 03:13:45.831: GPRS:1234050000000010:GPRS:DCCA: 3GPP-GGSN-MCC-MNC : 001002
Jun  2 03:13:45.831: GPRS:1234050000000010:GPRS:DCCA: 3GPP-NSAPI : 1
Jun  2 03:13:45.831: GPRS:1234050000000010:GPRS:DCCA: 3GPP-Selection-Mode : 0
Jun  2 03:13:45.831: GPRS:1234050000000010:3GPP-Charging-Char : 0100
Jun  2 03:13:45.831: GPRS:1234050000000010:GPRS DCCA: Starting Tx timer , value = 100000
Jun  2 03:13:45.831: GPRS:1234050000000010:DCCA FSM:Event = CCR_INITIAL, Old State = IDLE,
 New State = PENDING_I
Jun  2 03:13:46.287: GPRS:1234050000000010:GPRS DCCA: Result-Code = 2001
Jun  2 03:13:46.287: GPRS:1234050000000010:GPRS DCCA: Stopping Tx timer
Jun  2 03:13:46.287: GPRS:1234050000000010:GPRS DCCA: Result-Code for Category  : 1  = 2001
Jun  2 03:13:46.287: GPRS:1234050000000010:DCCA FSM:Event = CCA_SUCCESS, Old State =
PENDING_I, New State = OPEN
```

```
Router#
Router#show gprs gtp pdp tid  1234050000000010 ser all
Diameter Credit Control: Enabled
Current Billing status: Prepaid
Reason to convert to postpaid: N/A
Charging Profile Index: 1
DCCA profile name: dcca-profile1, Source: charging profile
Rule base id: 1,  Source: DCCA server
ServiceID  State         Quota(octets)
 1        AUTHORIZED        5000
Router#
PDP being deleted
```

Examples

The following is a sample of DCCA debug information without pre-allocated quota.

```
Router#show debug
GPRS:
  GPRS DCCA Events debugging is on
Router#
Jun  2 03:05:07.743: GPRS:1234050000000010:GPRS:DCCA: 3GPP-IMSI : 214350000000000
Jun  2 03:05:07.743: GPRS:1234050000000010:GPRS:DCCA: 3GPP-Charging-Id : 613053181
Jun  2 03:05:07.743: GPRS:1234050000000010:GPRS:DCCA: 3GPP-PDP-Type : 0
Jun  2 03:05:07.743: GPRS:1234050000000010:GPRS:DCCA: 3GPP-CG-Address : 20.1.1.1
Jun  2 03:05:07.743: GPRS:1234050000000010:GPRS:DCCA: 3GPP-QoS-Profile :
99-0911012964FFFF1100FFFF
Jun  2 03:05:07.743: GPRS:1234050000000010:GPRS:DCCA: 3GPP-SGSN-Address : 11.20.1.1
Jun  2 03:05:07.743: GPRS:1234050000000010:GPRS:DCCA: 3GPP-GGSN-Address : 10.20.61.1
Jun  2 03:05:07.743: GPRS:1234050000000010:GPRS:DCCA: 3GPP-IMSI-MCC-MNC : 21435
Jun  2 03:05:07.743: GPRS:1234050000000010:GPRS:DCCA: 3GPP-GGSN-MCC-MNC : 001002
Jun  2 03:05:07.743: GPRS:1234050000000010:GPRS:DCCA: 3GPP-NSAPI : 1
Jun  2 03:05:07.743: GPRS:1234050000000010:GPRS:DCCA: 3GPP-Selection-Mode : 0
Jun  2 03:05:07.743: GPRS:1234050000000010:3GPP-Charging-Char : 0100
Jun  2 03:05:07.743: GPRS:1234050000000010:GPRS DCCA: Starting Tx timer , value = 100000
Jun  2 03:05:07.743: GPRS:1234050000000010:DCCA FSM:Event = CCR_INITIAL, Old State = IDLE,
 New State = PENDING_I
Jun  2 03:05:08.167: GPRS:1234050000000010:GPRS DCCA: Result-Code = 2001
Jun  2 03:05:08.167: GPRS:1234050000000010:GPRS DCCA: Stopping Tx timer
Jun  2 03:05:08.167: GPRS:1234050000000010:DCCA FSM:Event = CCA_SUCCESS, Old State =
PENDING_I, New State = OPEN
Router#
gprs5-72b#sgpt 1234050000000010 ser all
Diameter Credit Control: Enabled
Current Billing status: Prepaid
Reason to convert to postpaid: N/A
Charging Profile Index: 1
DCCA profile name: dcca-profile1, Source: charging profile
Rule base id: 1,  Source: DCCA server
ServiceID  State         Quota(octets)
gprs5-72b#clear gprs gtp pdp all
PDP deleted
Router#
Jun  2 03:05:28.459: GPRS:1234050000000010:GPRS:DCCA: 3GPP-IMSI : 214350000000000
Jun  2 03:05:28.459: GPRS:1234050000000010:GPRS:DCCA: 3GPP-Charging-Id : 613053181
Jun  2 03:05:28.459: GPRS:1234050000000010:GPRS:DCCA: 3GPP-PDP-Type : 0
Jun  2 03:05:28.459: GPRS:1234050000000010:GPRS:DCCA: 3GPP-CG-Address : 20.1.1.1
Jun  2 03:05:28.459: GPRS:1234050000000010:GPRS:DCCA: 3GPP-QoS-Profile :
99-0911012964FFFF1100FFFF
Jun  2 03:05:28.459: GPRS:1234050000000010:GPRS:DCCA: 3GPP-SGSN-Address : 11.20.1.1
Jun  2 03:05:28.459: GPRS:1234050000000010:GPRS:DCCA: 3GPP-GGSN-Address : 10.20.61.1
Jun  2 03:05:28.459: GPRS:1234050000000010:GPRS:DCCA: 3GPP-IMSI-MCC-MNC : 21435
Jun  2 03:05:28.459: GPRS:1234050000000010:GPRS:DCCA: 3GPP-GGSN-MCC-MNC : 001002
Jun  2 03:05:28.459: GPRS:1234050000000010:GPRS:DCCA: 3GPP-NSAPI : 1
Jun  2 03:05:28.459: GPRS:1234050000000010:GPRS:DCCA: 3GPP-Selection-Mode : 0
Jun  2 03:05:28.459: GPRS:1234050000000010:3GPP-Charging-Char : 0100
Jun  2 03:05:28.463: GPRS:1234050000000010:GPRS DCCA: Stopping Tx timer
Jun  2 03:05:28.463: GPRS:1234050000000010:DCCA FSM:Event = CCR_FINAL, Old State = OPEN,
New State = PENDING_T
Jun  2 03:05:28.463: GPRS:1234050000000010:GPRS DCCA: Stopping Tx timer
Jun  2 03:05:28.871: GPRS:GPRS DCCA: DCCA request was cancelled, Droping AAA reply
Router#
```

```
Router#sgpt 1234050000000010 ser all
%ERROR: Cannot find the PDP
Router#
```

debug gprs dfp

To display debug messages for GPRS DFP weight calculation, use the **debug gprs dfp**privileged EXEC command. To disable debugging output, use the **no** form of this command.

debug gprs dfp

no debug gprs dfp

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values.

Command History

Release	Modification
12.1(9)E	This command was introduced.
12.2(4)MX	This command was incorporated in Cisco IOS Release 12.2(4)MX.
12.2(8)YD	This command was incorporated in Cisco IOS Release 12.2(8)YD.
12.2(8)YW	This command was incorporated in Cisco IOS Release 12.2(8)YW.
12.3(2)XB	This command was incorporated in Cisco IOS Release 12.3(2)XB.
12.3(8)XU	This command was incorporated in Cisco IOS Release 12.3(8)XU.
12.3(11)YJ	This command was incorporated in Cisco IOS Release 12.3(11)YJ.
12.3(14)YQ	This command was incorporated in Cisco IOS Release 12.3(14)YQ.

Usage Guidelines

See the following caution before using debug commands:

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

This command displays debug messages for GPRS DFP weight calculation. To display debug messages for the DFP agent subsystem, use the **debug ip dfp agent**command.

Examples

The following example configures a debug session to check all GPRS DFP weight calculation:

```
Router# debug gprs dfp
GPRS DFP debugging is on
Router#
```

The following example stops all debugging:

```
Router# no debug all
All possible debugging has been turned off
Router#
```

debug gprs dhcp

To display information about Dynamic Host Configuration Protocol (DHCP) processing on the GGSN, use the **debug gprs dhcp**privileged EXEC command. To disable debugging output, use the **no** form of this command.

debug gprs dhcp

no debug gprs dhcp

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values.

Command History

Release	Modification
12.2(4)MX	This command was introduced.
12.2(8)YD	This command was incorporated in Cisco IOS Release 12.2(8)YD.
12.2(8)YW	This command was incorporated in Cisco IOS Release 12.2(8)YW.
12.3(2)XB	This command was incorporated in Cisco IOS Release 12.3(2)XB.
12.3(8)XU	This command was incorporated in Cisco IOS Release 12.3(8)XU.
12.3(11)YJ	This command was incorporated in Cisco IOS Release 12.3(11)YJ.
12.3(14)YQ	This command was incorporated in Cisco IOS Release 12.3(14)YQ.

Usage Guidelines

This command is useful for system operators and development engineers if problems are encountered with DHCP processing on the GGSN. To display standard debug messages between the DHCP client on the router and a DHCP server, you can also use the **debug dhcp** or **debug dhcp detail** commands with the **debug gprs dhcp** command.

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

Examples

The following example shows sample output for DHCP processing on the GGSN :

```
Router# debug gprs dhcp
2d13h: GPRS:DHCP req:TID 1111111100000099, Req 1
2d13h: GPRS:Requesting IP address for pdp 1111111100000099 from server 172.16.0.8 tableid
```

```
0
2d13h: GPRS:DHCP ip allocation pass (10.88.17.43) for pdp 1111111100000099
2d13h: GPRS:Using DHCP ip address 10.88.17.43 for pdp 1111111100000099
```

The following example shows sample output for standard debug messaging for DHCP processing on the router between the DHCP client and a DHCP server:

```
2d13h: DHCP: proxy allocate request
2d13h: DHCP: new entry. add to queue
2d13h: DHCP: SDiscover attempt # 1 for entry:
2d13h: DHCP: SDiscover: sending 283 byte length DHCP packet
2d13h: DHCP: SDiscover with directed serv 172.16.0.8, 283 bytes
2d13h: DHCP: XID MATCH in dhcpc_for_us()
2d13h: DHCP: Received a BOOTREP pkt
2d13h: DHCP: offer received from 172.16.0.8
2d13h: DHCP: SRequest attempt # 1 for entry:
2d13h: DHCP: SRequest- Server ID option: 172.16.0.8
2d13h: DHCP: SRequest- Requested IP addr option: 10.88.17.43
2d13h: DHCP: SRequest placed lease len option: 604800
2d13h: DHCP: SRequest: 301 bytes
2d13h: DHCP: SRequest: 301 bytes
2d13h: DHCP: XID MATCH in dhcpc_for_us()
2d13h: DHCP: Received a BOOTREP pkt
2d13h: DHCP Proxy Client Pooling: ***Allocated IP address: 10.88.17.43
```

Related Commands

Command	Description
debug dhcp	Displays debug messages between the DHCP client on the router and a DHCP server.

debug gprs gtp

To display information about the GPRS Tunneling Protocol (GTP), use the **debug gprs gtp**privileged EXEC command. To disable debugging output, use the **no** form of this command.

debug gprs gtp {**events**| **messages**| **packets**}

no debug gprs gtp {**events**| **messages**| **packets**}

Syntax Description

events	Displays events related to GTP processing on the GGSN.
messages	Displays GTP signaling messages that are sent between the SGSN and GGSN.
packets	Displays GTP packets that are sent between the SGSN and GGSN.

Command Default

No default behavior or values.

Command History

Release	Modification
12.1(1)GA	This command was introduced.
12.1(5)T	This command was integrated in Cisco IOS Release 12.1(5)T.
12.2(4)MX	This command was incorporated in Cisco IOS Release 12.2(4)MX, and the **ppp** {**details** | **events**} option was added.
12.2(8)YD	This command was incorporated in Cisco IOS Release 12.2(8)YD.
12.2(8)YW	This command was incorporated in Cisco IOS Release 12.2(8)YW.
12.3(2)XB	This command was incorporated in Cisco IOS Release 12.3(2)XB.
12.3(8)XU	This command was incorporated in Cisco IOS Release 12.3(8)XU.
12.3(11)YJ	This command was incorporated in Cisco IOS Release 12.3(11)YJ.
12.3(14)YQ	This command was incorporated in Cisco IOS Release 12.3(14)YQ.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.

Usage Guidelines

This command is useful for system operators and development engineers if problems are encountered with communication between the GGSN and the SGSN using GTP.

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

Examples

The following example enables the display of events related to GTP processing on the GGSN :

```
Router# debug gprs gtp events
```

The following example enables the display of GTP signaling messages :

```
Router# debug gprs gtp messages
```

The following example enables the display of GTP packets sent between the SGSN and GGSN :

```
Router# debug gprs gtp packets
```

The following example enables the display of GTP PPP events between the SGSN and GGSN :

```
Router# debug gprs gtp ppp events
```

The following example enables the display of detailed GTP PPP debug output along with GTP PPP events between the SGSN and GGSN :

```
Router# debug gprs gtp ppp details
Router# debug gprs gtp ppp events
```

debug gprs gtp parsing

To display information about the parsing of GPRS Tunneling Protocol (GTP) information elements (IEs) in signaling requests, use the **debug gprs gtp parsing**privileged EXEC command. To disable debugging output, use the **no** form of this command.

debug gprs gtp parsing

no debug gprs gtp parsing

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values.

Command History

Release	Modification
12.2(4)MX	This command was introduced.
12.2(8)YD	This command was incorporated in Cisco IOS Release 12.2(8)YD.
12.2(8)YW	This command was incorporated in Cisco IOS Release 12.2(8)YW.
12.3(2)XB	This command was incorporated in Cisco IOS Release 12.3(2)XB.
12.3(8)XU	This command was incorporated in Cisco IOS Release 12.3(8)XU.
12.3(11)YJ	This command was incorporated in Cisco IOS Release 12.3(11)YJ.
12.3(14)YQ	This command was incorporated in Cisco IOS Release 12.3(14)YQ.

Usage Guidelines

This command is useful for system operators and development engineers to verify parsing of GTP IEs in signaling requests that are received by GDM or by the GGSN. If the packet is parsed successfully, you will receive a message along with the TID for the packet as shown in the following example:

```
GPRS:TID:7300000000000000:Packet Parsed successfully
```

The **debug gprs gtp parsing** command can be used to verify GDM or GGSN processing of IEs.

Note Because the **debug gprs gtp parsing**command generates a significant amount of output, use it only when traffic on the GPRS network is low, so other activity on the system is not adversely affected.

Examples

The following example enables the display of debug messages that occur while GDM or the GGSN parses GTP IEs :

```
Router# debug gprs gtp parsing
```

debug gprs gtp ppp

To display information about PPP PDP type processing on the GGSN, use the **debug gprs gtp ppp**privileged EXEC command. To disable debugging output, use the **no** form of this command.

debug gprs gtp ppp {**events**| **details**}

no debug gprs gtp ppp {**events**| **details**}

Syntax Description

events	Displays messages specific to certain conditions that are occurring during PPP PDP type processing.
details	Displays more extensive and lower-level messages related to PPP PDP type processing.

Command Default

No default behavior or values.

Command History

Release	Modification
12.2(4)MX	This command was introduced.
12.2(8)YD	This command was incorporated in Cisco IOS Release 12.2(8)YD.
12.2(8)YW	This command was incorporated in Cisco IOS Release 12.2(8)YW.
12.3(2)XB	This command was incorporated in Cisco IOS Release 12.3(2)XB.
12.3(8)XU	This command was incorporated in Cisco IOS Release 12.3(8)XU.
12.3(11)YJ	This command was incorporated in Cisco IOS Release 12.3(11)YJ.
12.3(14)YQ	This command was incorporated in Cisco IOS Release 12.3(14)YQ.

Usage Guidelines

This command is useful for system operators and development engineers if problems are encountered with PPP PDP type processing on the GGSN.

You can enable both forms of the **debug gprs gtp ppp** command at the same time, as separate command line entries. The **events** keyword generates output specific to certain conditions that are occurring, which helps qualify the output being received using the **details** option.

> **Note**
> Because the **debug gprs gtp ppp**command generates a significant amount of output, use it only when traffic on the GPRS network is low, so other activity on the system is not adversely affected.

Examples

The following debug examples provide sample output for a create PDP context request and clear PDP context using PPP PDP type on the GGSN. The examples show output while both debug events and details are enabled on the GGSN.

Example 1

The following example displays details and events output related to PPP PDP context processing for a create PDP context requested received by the GGSN :

```
Router# debug gprs gtp ppp events
GTP PPP events display debugging is on
Router# debug gprs gtp ppp details
GTP PPP details display debugging is on
7200b#
3d23h: GPRS:
3d23h: GTP-PPP Fa1/0: Create new gtp_ppp_info
3d23h: GPRS:
3d23h: GTP-PPP: domain gprs.cisco.com not in any VPDN group
3d23h: GPRS:
3d23h: GTP-PPP: aaa-group accounting not configured under APN gprs.cisco.com
3d23h: GPRS:GTP-PPP: Don't cache internally generated pak's header
3d23h: %LINK-3-UPDOWN: Interface Virtual-Access2, changed state to up
3d23h: GPRS:
3d23h: GTP-PPP Vi2: gtp_ppp_cstate_react changing states
3d23h: GPRS:GTP-PPP: pdp_entry 0x62F442A4, recv ppp data pak
3d23h: GPRS:GTP-PPP Vi2: proc_udp_input pak's linktype = 30
3d23h: GPRS:GTP-PPP: pdp_entry 0x62F442A4, recv ppp data pak
3d23h: GPRS:GTP-PPP Vi2: proc_udp_input pak's linktype = 30
3d23h: GPRS:GTP-PPP: pdp_entry 0x62F442A4, recv ppp data pak
3d23h: GPRS:GTP-PPP Vi2: proc_udp_input pak's linktype = 30
3d23h: GPRS:
3d23h: GTP-PPP: Vi2: Concat names user00 & gprs.cisco.com
3d23h: GPRS:
3d23h: GTP-PPP: New username after concat: user00@gprs.cisco.com
3d23h: GPRS:
3d23h: GTP-PPP: Vi2: Concat names user00@gprs.cisco.com & gprs.cisco.com
3d23h: GPRS:
3d23h: GTP-PPP: New username after concat: user00@gprs.cisco.com
3d23h: GPRS:GTP-PPP: pdp_entry 0x62F442A4, recv ppp data pak
3d23h: GPRS:GTP-PPP Vi2: proc_udp_input pak's linktype = 30
3d23h: GPRS:GTP-PPP: pdp_entry 0x62F442A4, recv ppp data pak
3d23h: GPRS:GTP-PPP Vi2: proc_udp_input pak's linktype = 30
3d23h: GPRS:GTP-PPP: pdp_entry 0x62F442A4, recv ppp data pak
3d23h: GPRS:GTP-PPP Vi2: proc_udp_input pak's linktype = 30
3d23h: GPRS:GTP-PPP: pdp_entry 0x62F442A4, recv ppp data pak
3d23h: GPRS:GTP-PPP Vi2: proc_udp_input pak's linktype = 30
3d23h: %LINEPROTO-5-UPDOWN: Line protocol on Interface Virtual-Access2, changed state to
up
3d23h: GPRS:GTP-PPP: pdp_entry 0x62F442A4, recv ppp data pak
3d23h: GPRS:GTP-PPP Vi2: proc_udp_input pak's linktype = 30
3d23h: GPRS:GTP-PPP: pdp_entry 0x62F442A4, recv ppp data pak
3d23h: GPRS:GTP-PPP Vi2: proc_udp_input pak's linktype = 30
3d23h: GPRS:
3d23h: GTP-PPP Vi2: gtp_ppp_protocol_up is notified about intf UP
3d23h: GPRS:
```

3d23h: GTP-PPP Vi2: PDP w/ MS addr 98.102.0.1 inserted into IP radix tree

Examples

The following example displays both details and events related to PPP PDP type processing after clearing PDP contexts on the GGSN :

```
Router# clear gprs gtp pdp-context all
3d23h: GPRS:GTP-PPP: pdp_entry 0x62F442A4, recv ppp data pak
3d23h: GPRS:GTP-PPP Vi2: proc_udp_input pak's linktype = 30
3d23h: GPRS:GTP-PPP: pdp_entry 0x62F442A4, recv ppp data pak
3d23h: GPRS:GTP-PPP Vi2: proc_udp_input pak's linktype = 30
3d23h: GPRS:
3d23h: GTP-PPP Vi2: gtp_ppp_pdp_terminate shutting down the vaccess
3d23h: GPRS:
3d23h: GTP-PPP Vi2: gtp_ppp_pdp_shut_va shutting down intf
3d23h: %LINK-3-UPDOWN: Interface Virtual-Access2, changed state to down
3d23h: GPRS:
3d23h: GTP-PPP Vi2: gtp_ppp_cstate_react changing states
3d23h: GPRS:
3d23h: GTP-PPP Vi2: gtp_ppp_free_va resetting intf vectors
3d23h: %LINEPROTO-5-UPDOWN: Line protocol on Interface Virtual-Access2, changed state to
down
```

debug gprs gtp ppp-regeneration

To display information about PPP regeneration processing on the GGSN, use the **debug gprs gtp ppp-regeneration**privileged EXEC command. To disable debugging output, use the **no** form of this command.

debug gprs gtp ppp-regeneration {**events**| **details**}

no debug gprs gtp ppp-regeneration {**events**| **details**}

Syntax Description

events	Displays messages specific to certain conditions that are occurring during PPP regeneration processing.
details	Displays more extensive and lower-level messages related to PPP regeneration processing.

Command Default

No default behavior or values.

Command History

Release	Modification
12.2(4)MX	This command was introduced.
12.2(8)YD	This command was incorporated in Cisco IOS Release 12.2(8)YD.
12.2(8)YW	This command was incorporated in Cisco IOS Release 12.2(8)YW.
12.3(2)XB	This command was incorporated in Cisco IOS Release 12.3(2)XB.
12.3(8)XU	This command was incorporated in Cisco IOS Release 12.3(8)XU.
12.3(11)YJ	This command was incorporated in Cisco IOS Release 12.3(11)YJ.
12.3(14)YQ	This command was incorporated in Cisco IOS Release 12.3(14)YQ.

Usage Guidelines

This command is useful for system operators and development engineers if problems are encountered with communication between GDM and a GGSN.

You can enable both forms of the **debug gprs gtp ppp-regeneration** command at the same time, as separate command line entries. The **events** keyword generates output specific to certain conditions that are occurring, which helps qualify the output being received using the **details** option.

Note Because the **debug gprs gtp ppp-regeneration**command generates a significant amount of output, use it only when traffic on the GPRS network is low, so other activity on the system is not adversely affected.

Examples

The following debug examples provide sample output for a create PDP context request and clear PDP context using PPP regeneration on the GGSN. The examples show output while both debug events and details are enabled on the GGSN.

Example 1

The following example displays details and events output related to PPP regeneration processing for a create PDP context requested received by the GGSN :

```
Router# debug gprs gtp ppp-regeneration details
GTP PPP regeneration details display debugging is on
Router# debug gprs gtp ppp-regeneration events
GTP PPP regeneration events display debugging is on
06:24:02: PPP-REGEN state counters: pending counter is 0
06:24:02:               State[IDLE] counter is 0
06:24:02:               State[AUTHORIZING] counter is 0
06:24:02:               State[VPDN CONNECTING] counter is 0
06:24:02:               State[PPP NEGOTIATING] counter is 0
06:24:02:               State[PPP CONNECTED] counter is 0
06:24:02:               State[PPP TERMINATING] counter is 0
06:24:02: PPP-REGEN state counters: pending counter is 1
06:24:02:               State[IDLE] counter is 1
06:24:02:               State[AUTHORIZING] counter is 0
06:24:02:               State[VPDN CONNECTING] counter is 0
06:24:02:               State[PPP NEGOTIATING] counter is 0
06:24:02:               State[PPP CONNECTED] counter is 0
06:24:02:               State[PPP TERMINATING] counter is 0
06:24:02: GPRS:1011111111500001:Authen: PAP username: tomy1@corporate_1.com
06:24:02: GPRS:1011111111500001:Session timer started
06:24:02: GPRS:Processing PPP regen reqQ
06:24:02: GPRS:1011111111500001:Processing Initiate PPP  regen from reqQ
06:24:02: GPRS:1011111111500001:got event [REQUEST PPP REGEN] in state [IDLE]
06:24:02: PPP-REGEN state counters: pending counter is 1
06:24:02:               State[IDLE] counter is 0
06:24:02:               State[AUTHORIZING] counter is 1
06:24:02:               State[VPDN CONNECTING] counter is 0
06:24:02:               State[PPP NEGOTIATING] counter is 0
06:24:02:               State[PPP CONNECTED] counter is 0
06:24:02:               State[PPP TERMINATING] counter is 0
06:24:02: GPRS:1011111111500001:state [IDLE->AUTHORIZING] on event [REQUEST PPP REGEN]
06:24:02: GPRS:1011111111500001:Got VPN authorization info
06:24:02: GPRS:1011111111500001:got event [AUTHOR SUCCESS] in state [AUTHORIZING]
06:24:02: PPP-REGEN state counters: pending counter is 1
06:24:02:               State[IDLE] counter is 0
06:24:02:               State[AUTHORIZING] counter is 0
06:24:02:               State[VPDN CONNECTING] counter is 1
06:24:02:               State[PPP NEGOTIATING] counter is 0
06:24:02:               State[PPP CONNECTED] counter is 0
06:24:02:               State[PPP TERMINATING] counter is 0
06:24:02: GPRS:1011111111500001:state [AUTHORIZING->VPDN CONNECTING] on event [AUTHOR
SUCCESS]
06:24:02: GPRS:1011111111500001:Author succeeded, establishing the tunnel
06:24:02: GPRS:1011111111500001:Create/Clone vaccess to negotiate PPP
06:24:02: GPRS:1011111111500001:no need to set NS ppp_config
06:24:02: GPRS:1011111111500001:MS no static IP addr. Get one via IPCP
06:24:02: GPRS:1011111111500001:VPDN to inform PPP regen: CONNECTED
06:24:02: GPRS:1011111111500001:got event [VPDN CONNECTED] in state [VPDN CONNECTING]
06:24:02: PPP-REGEN state counters: pending counter is 1
06:24:02:               State[IDLE] counter is 0
06:24:02:               State[AUTHORIZING] counter is 0
06:24:02:               State[VPDN CONNECTING] counter is 0
```

```
06:24:02:               State[PPP NEGOTIATING] counter is 1
06:24:02:               State[PPP CONNECTED] counter is 0
06:24:02:               State[PPP TERMINATING] counter is 0
06:24:02: GPRS:1011111111500001:state [VPDN CONNECTING->PPP NEGOTIATING] on event [VPDN
CONNECTED]
06:24:02: GPRS:1011111111500001:Start PPP negotiations on vaccess
06:24:02: %LINK-3-UPDOWN: Interface Virtual-Access2, changed state to up
06:24:02: GPRS:1011111111500001:IPCP is up
06:24:02: GPRS:1011111111500001:LNS allocates 10.100.1.1 for MS
06:24:02: GPRS:1011111111500001:IP addr 10.100.1.1 is negotiated for MS
06:24:02: GPRS:1011111111500001:PPP connected
06:24:02: GPRS:1011111111500001:got event [PPP NEGOTIATED] in state [PPP NEGOTIATING]
06:24:02: PPP-REGEN state counters: pending counter is 0
06:24:02:               State[IDLE] counter is 0
06:24:02:               State[AUTHORIZING] counter is 0
06:24:02:               State[VPDN CONNECTING] counter is 0
06:24:02:               State[PPP NEGOTIATING] counter is 0
06:24:02:               State[PPP CONNECTED] counter is 1
06:24:02:               State[PPP TERMINATING] counter is 0
06:24:02: GPRS:1011111111500001:state [PPP NEGOTIATING->PPP CONNECTED] on event [PPP
NEGOTIATED]
06:24:02: GPRS:1011111111500001:PPP succeeded negotiation, session established
06:24:02: GPRS:1011111111500001:Session timer stopped
06:24:03: %LINEPROTO-5-UPDOWN: Line protocol on Interface Virtual-Access2, changed state
to up
```

Examples

The following example displays both details and events related to PPP regeneration processing after clearing PDP contexts on the GGSN :

```
Router# clear gprs gtp pdp-context all
06:28:05: PPP-REGEN state counters: pending counter is 0
06:28:05:               State[IDLE] counter is 0
06:28:05:               State[AUTHORIZING] counter is 0
06:28:05:               State[VPDN CONNECTING] counter is 0
06:28:05:               State[PPP NEGOTIATING] counter is 0
06:28:05:               State[PPP CONNECTED] counter is 1
06:28:05:               State[PPP TERMINATING] counter is 0
06:28:05: GPRS:1011111111500001:PPP regen current state PPP CONNECTED
06:28:05: GPRS:1011111111500001:GTP disconnecting the PPP regen session
06:28:05: GPRS:Processing PPP regen reqQ
06:28:05: GPRS:1011111111500001:Processing Disconnect PPP  regen from reqQ
06:28:05: GPRS:1011111111500001:got event [CANCEL REGEN'ED PPP] in state [PPP CONNECTED]
06:28:05: PPP-REGEN state counters: pending counter is 1
06:28:05:               State[IDLE] counter is 0
06:28:05:               State[AUTHORIZING] counter is 0
06:28:05:               State[VPDN CONNECTING] counter is 0
06:28:05:               State[PPP NEGOTIATING] counter is 0
06:28:05:               State[PPP CONNECTED] counter is 0
06:28:05:               State[PPP TERMINATING] counter is 1
06:28:05: GPRS:1011111111500001:state [PPP CONNECTED->PPP TERMINATING] on event [CANCEL
REGEN'ED PPP]
06:28:05: GPRS:1011111111500001:Cancel request after VPND tunnel is up
06:28:05: PPP-REGEN state counters: pending counter is 1
06:28:05:               State[IDLE] counter is 0
06:28:05:               State[AUTHORIZING] counter is 0
06:28:05:               State[VPDN CONNECTING] counter is 0
06:28:05:               State[PPP NEGOTIATING] counter is 0
06:28:05:               State[PPP CONNECTED] counter is 0
06:28:05:               State[PPP TERMINATING] counter is 1
06:28:05: GPRS:1011111111500001:PPP down
06:28:05: GPRS:1011111111500001:got event [PPP FAILED] in state [PPP TERMINATING]
06:28:05: PPP-REGEN state counters: pending counter is 1
06:28:05:               State[IDLE] counter is 1
06:28:05:               State[AUTHORIZING] counter is 0
06:28:05:               State[VPDN CONNECTING] counter is 0
06:28:05:               State[PPP NEGOTIATING] counter is 0
06:28:05:               State[PPP CONNECTED] counter is 0
06:28:05:               State[PPP TERMINATING] counter is 0
06:28:05: GPRS:1011111111500001:state [PPP TERMINATING->IDLE] on event [PPP FAILED]
06:28:05: GPRS:1011111111500001:LCP went down
```

```
06:28:05: GPRS:1011111111500001:VPDN disconnect
06:28:05: GPRS:1011111111500001:got event [CLEANUP CONTEXT] in state [IDLE]
06:28:05: GPRS:1011111111500001:state [IDLE->IDLE] on event [CLEANUP CONTEXT]
06:28:05: GPRS:1011111111500001:Freeing context structure
06:28:05: GPRS:1011111111500001:VPDN handle invalid, no need to free it
06:28:05: GPRS:1011111111500001:remove PPP regen context from Vi2
06:28:05: GPRS:1011111111500001:Session timer stopped
06:28:05: PPP-REGEN state counters: pending counter is 0
06:28:05:               State[IDLE] counter is 0
06:28:05:               State[AUTHORIZING] counter is 0
06:28:05:               State[VPDN CONNECTING] counter is 0
06:28:05:               State[PPP NEGOTIATING] counter is 0
06:28:05:               State[PPP CONNECTED] counter is 0
06:28:05:               State[PPP TERMINATING] counter is 0
06:28:05: GPRS:1011111111500001:PPP regen context 0x633F196C released
06:28:05: %LINK-3-UPDOWN: Interface Virtual-Access2, changed state to down
06:28:06: %LINEPROTO-5-UPDOWN: Line protocol on Interface Virtual-Access2, changed state
to down
```

debug gprs gtp-director

To display information about the GTP Director Module (GDM), use the **debug gprs gtp-director**privileged EXEC command. To disable debugging output, use the **no** form of this command.

debug gprs gtp-director {**events**| **packets**}

no debug gprs gtp-director {**events**| **packets**}

Syntax Description

events	Displays events related to GDM processing.
packets	Displays packets that are sent between GDM and a GGSN.

Command Default

No default behavior or values.

Command History

Release	Modification
12.2(4)MX	This command was introduced.
12.2(8)YD	This command was incorporated in Cisco IOS Release 12.2(8)YD.
12.2(8)B	This command was incorporated in Cisco IOS Release 12.2(8)B.

Usage Guidelines

This command is useful for system operators and development engineers if problems are encountered with communication between GDM and an SGSN, or between GDM and a GGSN.

Note Because the **debug gprs gtp-director**command generates a significant amount of output, use it only when traffic on the GPRS network is low, so other activity on the system is not adversely affected.

Examples

The following debug examples provide sample output for a create PDP context request, delete PDP context request, and clear PDP context using PPP regeneration on the GGSN. The first three examples show output related to debug events messaging only. The last three examples show output while both debug events and details are enabled on the GGSN.

Examples

The following example displays events related to PPP regeneration processing for a create PDP context requested received by the GGSN :

```
Router# debug gprs gtp-director events
```

```
*Mar  1 00:02:42.787: GPRS:1111110000000000:Authen: PAP username: user@pdn.com
*Mar  1 00:02:42.787: GPRS:1111110000000000:Processing Initiate PPP regen from reqQ
*Mar  1 00:02:42.787: GPRS:1111110000000000:got event [REQUEST PPP REGEN] in state [IDLE]
*Mar  1 00:02:42.787: GPRS:1111110000000000:state [IDLE->AUTHORIZING] on event [REQUEST PPP
 REGEN]
*Mar  1 00:02:42.787: GPRS:1111110000000000:Got VPN authorization info
*Mar  1 00:02:42.787: GPRS:1111110000000000:got event [AUTHOR SUCCESS] in state [AUTHORIZING]
*Mar  1 00:02:42.787: GPRS:1111110000000000:state [AUTHORIZING->VPDN CONNECTING] on event
[AUTHOR SUCCESS]
*Mar  1 00:02:42.787: GPRS:1111110000000000:Author succeeded, establishing the tunnel
*Mar  1 00:02:42.787: GPRS:1111110000000000:Create/Clone vaccess to negotiate PPP
*Mar  1 00:02:42.791: GPRS:1111110000000000:MS no static IP addr. Get one via IPCP
*Mar  1 00:02:42.827: GPRS:1111110000000000:VPDN to inform PPP regen: CONNECTED
*Mar  1 00:02:42.827: GPRS:1111110000000000:got event [VPDN CONNECTED] in state [VPDN
CONNECTING]
*Mar  1 00:02:42.827: GPRS:1111110000000000:state [VPDN CONNECTING->PPP NEGOTIATING] on
event [VPDN CONNECTED]
*Mar  1 00:02:42.827: GPRS:1111110000000000:Start PPP negotiations on vaccess
*Mar  1 00:02:42.831: %LINK-3-UPDOWN: Interface Virtual-Access3, changed state to up
*Mar  1 00:02:42.835: GPRS:1111110000000000:IPCP is up
*Mar  1 00:02:42.835: GPRS:1111110000000000:IP addr 10.10.1.187 is negotiated for MS
*Mar  1 00:02:42.835: GPRS:1111110000000000:DNS - Primary: 10.3.0.1 Secondary: 0.0.0.0
NetBios - Primary: 0.0.0.0, Secondary: 0.0.0.0
*Mar  1 00:02:42.835: GPRS:1111110000000000:PPP connected
*Mar  1 00:02:42.835: GPRS:1111110000000000:got event [PPP NEGOTIATED] in state [PPP
NEGOTIATING]
*Mar  1 00:02:42.835: GPRS:1111110000000000:state [PPP NEGOTIATING->PPP CONNECTED] on event
 [PPP NEGOTIATED]
*Mar  1 00:02:42.835: GPRS:1111110000000000:PPP succeeded negotiation, session established
*Mar  1 00:02:43.835: %LINEPROTO-5-UPDOWN: Line protocol on Interface Virtual-Access3,
changed state to up
```

Example 2

The following example displays events related to PPP regeneration processing for a delete PDP context requested received by the GGSN :

```
Router# debug gprs gtp-director events
*Mar  1 00:03:18.331: GPRS:1111110000000000:GTP disconnecting the PPP regen session
*Mar  1 00:03:18.331: GPRS:1111110000000000:Processing Disconnect PPP regen from reqQ
*Mar  1 00:03:18.331: GPRS:1111110000000000:got event [CANCEL REGEN'ED PPP] in state [PPP
CONNECTED]
*Mar  1 00:03:18.331: GPRS:1111110000000000:state [PPP CONNECTED->PPP TERMINATING] on event
 [CANCEL REGEN'ED PPP]
*Mar  1 00:03:18.331: GPRS:1111110000000000:Cancel request after VPND tunnel is up
*Mar  1 00:03:18.335: GPRS:1111110000000000:PPP down
*Mar  1 00:03:18.335: GPRS:1111110000000000:got event [PPP FAILED] in state [PPP TERMINATING]
*Mar  1 00:03:18.339: GPRS:1111110000000000:state [PPP TERMINATING->IDLE] on event [PPP
FAILED]
*Mar  1 00:03:18.339: GPRS:1111110000000000:PPP failed negotiation
*Mar  1 00:03:18.339: GPRS:1111110000000000:got event [CLEANUP CONTEXT] in state [IDLE]
*Mar  1 00:03:18.339: GPRS:1111110000000000:VPDN to inform PPP regen: DISCONNECTED
*Mar  1 00:03:18.339: GPRS:1111110000000000:got event [VPDN DISCONNECTED] in state [IDLE]
*Mar  1 00:03:18.339: GPRS:1111110000000000:state [IDLE->IDLE] on event [CLEANUP CONTEXT]
*Mar  1 00:03:18.339: GPRS:1111110000000000:Freeing context structure
*Mar  1 00:03:18.339: %LINK-3-UPDOWN: Interface Virtual-Access3, changed state to down
*Mar  1 00:03:19.331: %LINEPROTO-5-UPDOWN: Line protocol on Interface Virtual-Access3,
changed state to down
```

Example 3

The following example displays events related to PPP regeneration processing as the GGSN clears a PDP context request :

```
Router# debug gprs gtp-director events
*Mar  1 00:04:50.083: GPRS:1111110000000000:GTP disconnecting the PPP regen session
*Mar  1 00:04:50.083: GPRS:1111110000000000:Processing Disconnect PPP regen from reqQ
*Mar  1 00:04:50.083: GPRS:1111110000000000:got event [CANCEL REGEN'ED PPP] in state [PPP
CONNECTED]
*Mar  1 00:04:50.083: GPRS:1111110000000000:state [PPP CONNECTED->PPP TERMINATING] on event
 [CANCEL REGEN'ED PPP]
```

```
*Mar  1 00:04:50.083: GPRS:1111110000000000:Cancel request after VPND tunnel is up
*Mar  1 00:04:50.087: GPRS:1111110000000000:PPP down
*Mar  1 00:04:50.087: GPRS:1111110000000000:got event [PPP FAILED] in state [PPP TERMINATING]
*Mar  1 00:04:50.091: GPRS:1111110000000000:state [PPP TERMINATING->IDLE] on event [PPP
FAILED]
*Mar  1 00:04:50.091: GPRS:1111110000000000:PPP failed negotiation
*Mar  1 00:04:50.091: GPRS:1111110000000000:got event [CLEANUP CONTEXT] in state [IDLE]
*Mar  1 00:04:50.091: GPRS:1111110000000000:VPDN to inform PPP regen: DISCONNECTED
*Mar  1 00:04:50.091: GPRS:1111110000000000:got event [VPDN DISCONNECTED] in state [IDLE]
*Mar  1 00:04:50.091: GPRS:1111110000000000:state [IDLE->IDLE] on event [CLEANUP CONTEXT]
*Mar  1 00:04:50.091: GPRS:1111110000000000:Freeing context structure
*Mar  1 00:04:50.091: %LINK-3-UPDOWN: Interface Virtual-Access4, changed state to down
*Mar  1 00:04:51.083: %LINEPROTO-5-UPDOWN: Line protocol on Interface Virtual-Access4,
changed state to down
```

Example 4

The following example displays both debug events and details related to PPP regeneration processing for a create PDP context requested received by the GGSN :

```
Router# debug gprs gtp-director events
Router# debug gprs gtp-director details
*Mar  1 00:05:21.083: PPP-REGEN state counters: pending counter is 0
*Mar  1 00:05:21.083:           State[IDLE] counter is 0
*Mar  1 00:05:21.083:           State[AUTHORIZING] counter is 0
*Mar  1 00:05:21.083:           State[VPDN CONNECTING] counter is 0
*Mar  1 00:05:21.083:           State[PPP NEGOTIATING] counter is 0
*Mar  1 00:05:21.083:           State[PPP CONNECTED] counter is 0
*Mar  1 00:05:21.083:           State[PPP TERMINATING] counter is 0
*Mar  1 00:05:21.087: PPP-REGEN state counters: pending counter is 1
*Mar  1 00:05:21.087:           State[IDLE] counter is 1
*Mar  1 00:05:21.087:           State[AUTHORIZING] counter is 0
*Mar  1 00:05:21.087:           State[VPDN CONNECTING] counter is 0
*Mar  1 00:05:21.087:           State[PPP NEGOTIATING] counter is 0
*Mar  1 00:05:21.087:           State[PPP CONNECTED] counter is 0
*Mar  1 00:05:21.087:           State[PPP TERMINATING] counter is 0
*Mar  1 00:05:21.087: GPRS:1111110000000000:Authen: PAP username: user@pdn.com
*Mar  1 00:05:21.087: GPRS:1111110000000000:Session timer started
*Mar  1 00:05:21.087: GPRS:1111110000000000:Processing Initiate PPP regen from reqQ
*Mar  1 00:05:21.087: GPRS:1111110000000000:got event [REQUEST PPP REGEN] in state [IDLE]
*Mar  1 00:05:21.087: PPP-REGEN state counters: pending counter is 1
*Mar  1 00:05:21.087:           State[IDLE] counter is 0
*Mar  1 00:05:21.087:           State[AUTHORIZING] counter is 1
*Mar  1 00:05:21.087:           State[VPDN CONNECTING] counter is 0
*Mar  1 00:05:21.087:           State[PPP NEGOTIATING] counter is 0
*Mar  1 00:05:21.087:           State[PPP CONNECTED] counter is 0
*Mar  1 00:05:21.087:           State[PPP TERMINATING] counter is 0
*Mar  1 00:05:21.087: GPRS:1111110000000000:state [IDLE->AUTHORIZING] on event [REQUEST PPP
 REGEN]
*Mar  1 00:05:21.087: GPRS:1111110000000000:Got VPN authorization info
*Mar  1 00:05:21.087: GPRS:1111110000000000:got event [AUTHOR SUCCESS] in state [AUTHORIZING]
*Mar  1 00:05:21.087: PPP-REGEN state counters: pending counter is 1
*Mar  1 00:05:21.087:           State[IDLE] counter is 0
*Mar  1 00:05:21.087:           State[AUTHORIZING] counter is 0
*Mar  1 00:05:21.087:           State[VPDN CONNECTING] counter is 1
*Mar  1 00:05:21.087:           State[PPP NEGOTIATING] counter is 0
*Mar  1 00:05:21.087:           State[PPP CONNECTED] counter is 0
*Mar  1 00:05:21.087:           State[PPP TERMINATING] counter is 0
*Mar  1 00:05:21.087: GPRS:1111110000000000:state [AUTHORIZING->VPDN CONNECTING] on event
[AUTHOR SUCCESS]
*Mar  1 00:05:21.087: GPRS:1111110000000000:Author succeeded, establishing the tunnel
*Mar  1 00:05:21.087: GPRS:1111110000000000:Create/Clone vaccess to negotiate PPP
*Mar  1 00:05:21.091: GPRS:1111110000000000:MS no static IP addr. Get one via IPCP
*Mar  1 00:05:21.127: GPRS:1111110000000000:VPDN to inform PPP regen: CONNECTED
*Mar  1 00:05:21.127: GPRS:1111110000000000:got event [VPDN CONNECTED] in state [VPDN
CONNECTING]
*Mar  1 00:05:21.127: PPP-REGEN state counters: pending counter is 1
*Mar  1 00:05:21.127:           State[IDLE] counter is 0
*Mar  1 00:05:21.127:           State[AUTHORIZING] counter is 0
*Mar  1 00:05:21.127:           State[VPDN CONNECTING] counter is 0
*Mar  1 00:05:21.127:           State[PPP NEGOTIATING] counter is 1
*Mar  1 00:05:21.127:           State[PPP CONNECTED] counter is 0
```

```
*Mar  1 00:05:21.127:           State[PPP TERMINATING] counter is 0
*Mar  1 00:05:21.127: GPRS:1111110000000000:state [VPDN CONNECTING->PPP NEGOTIATING] on
event [VPDN CONNECTED]
*Mar  1 00:05:21.127: GPRS:1111110000000000:Start PPP negotiations on vaccess
*Mar  1 00:05:21.131: %LINK-3-UPDOWN: Interface Virtual-Access5, changed state to up
*Mar  1 00:05:22.135: %LINEPROTO-5-UPDOWN: Line protocol on Interface Virtual-Access5,
changed state to up
*Mar  1 00:05:23.143: GPRS:1111110000000000:IPCP is up
*Mar  1 00:05:23.143: GPRS:1111110000000000:LNS allocates 10.10.1.187 for MS
*Mar  1 00:05:23.143: GPRS:1111110000000000:IP addr 10.10.1.187 is negotiated for MS
*Mar  1 00:05:23.143: GPRS:1111110000000000:DNS - Primary: 10.3.0.1 Secondary: 0.0.0.0
NetBios - Primary: 0.0.0.0, Secondary: 0.0.0.0
*Mar  1 00:05:23.143: GPRS:1111110000000000:PPP connected
*Mar  1 00:05:23.143: GPRS:1111110000000000:got event [PPP NEGOTIATED] in state [PPP
NEGOTIATING]
*Mar  1 00:05:23.143: PPP-REGEN state counters: pending counter is 0
*Mar  1 00:05:23.143:           State[IDLE] counter is 0
*Mar  1 00:05:23.143:           State[AUTHORIZING] counter is 0
*Mar  1 00:05:23.143:           State[VPDN CONNECTING] counter is 0
*Mar  1 00:05:23.143:           State[PPP NEGOTIATING] counter is 0
*Mar  1 00:05:23.143:           State[PPP CONNECTED] counter is 1
*Mar  1 00:05:23.143:           State[PPP TERMINATING] counter is 0
*Mar  1 00:05:23.143: GPRS:1111110000000000:state [PPP NEGOTIATING->PPP CONNECTED] on event
 [PPP NEGOTIATED]
*Mar  1 00:05:23.143: GPRS:1111110000000000:PPP succeeded negotiation, session established
*Mar  1 00:05:23.143: GPRS:1111110000000000:Session timer stopped
```

Example 5

The following example displays both debug events and details related to PPP regeneration processing for a delete PDP context requested received by the GGSN :

```
Router# debug gprs gtp-director events
Router# debug gprs gtp-director details
*Mar  1 00:05:52.399: PPP-REGEN state counters: pending counter is 0
*Mar  1 00:05:52.399:           State[IDLE] counter is 0
*Mar  1 00:05:52.399:           State[AUTHORIZING] counter is 0
*Mar  1 00:05:52.399:           State[VPDN CONNECTING] counter is 0
*Mar  1 00:05:52.399:           State[PPP NEGOTIATING] counter is 0
*Mar  1 00:05:52.399:           State[PPP CONNECTED] counter is 1
*Mar  1 00:05:52.399:           State[PPP TERMINATING] counter is 0
*Mar  1 00:05:52.399: GPRS:1111110000000000:PPP regen current state PPP CONNECTED
*Mar  1 00:05:52.399: GPRS:1111110000000000:GTP disconnecting the PPP regen session
*Mar  1 00:05:52.399: GPRS:1111110000000000:Processing Disconnect PPP regen from reqQ
*Mar  1 00:05:52.399: GPRS:1111110000000000:got event [CANCEL REGEN'ED PPP] in state [PPP
CONNECTED]
*Mar  1 00:05:52.399: PPP-REGEN state counters: pending counter is 1
*Mar  1 00:05:52.399:           State[IDLE] counter is 0
*Mar  1 00:05:52.399:           State[AUTHORIZING] counter is 0
*Mar  1 00:05:52.399:           State[VPDN CONNECTING] counter is 0
*Mar  1 00:05:52.399:           State[PPP NEGOTIATING] counter is 0
*Mar  1 00:05:52.399:           State[PPP CONNECTED] counter is 0
*Mar  1 00:05:52.399:           State[PPP TERMINATING] counter is 1
*Mar  1 00:05:52.399: GPRS:1111110000000000:state [PPP CONNECTED->PPP TERMINATING] on event
 [CANCEL REGEN'ED PPP]
*Mar  1 00:05:52.399: GPRS:1111110000000000:Cancel request after VPND tunnel is up
*Mar  1 00:05:52.403: GPRS:1111110000000000:PPP down
*Mar  1 00:05:52.403: GPRS:1111110000000000:got event [PPP FAILED] in state [PPP TERMINATING]
*Mar  1 00:05:52.407: PPP-REGEN state counters: pending counter is 1
*Mar  1 00:05:52.407:           State[IDLE] counter is 1
*Mar  1 00:05:52.407:           State[AUTHORIZING] counter is 0
*Mar  1 00:05:52.407:           State[VPDN CONNECTING] counter is 0
*Mar  1 00:05:52.407:           State[PPP NEGOTIATING] counter is 0
*Mar  1 00:05:52.407:           State[PPP CONNECTED] counter is 0
*Mar  1 00:05:52.407:           State[PPP TERMINATING] counter is 0
*Mar  1 00:05:52.407: GPRS:1111110000000000:state [PPP TERMINATING->IDLE] on event [PPP
FAILED]
*Mar  1 00:05:52.407: GPRS:1111110000000000:PPP failed negotiation
*Mar  1 00:05:52.407: GPRS:1111110000000000:got event [CLEANUP CONTEXT] in state [IDLE]
*Mar  1 00:05:52.407: GPRS:1111110000000000:VPDN to inform PPP regen: DISCONNECTED
*Mar  1 00:05:52.407: GPRS:1111110000000000:got event [VPDN DISCONNECTED] in state [IDLE]
*Mar  1 00:05:52.407: GPRS:1111110000000000:state [IDLE->IDLE] on event [CLEANUP CONTEXT]
```

```
*Mar  1 00:05:52.407: GPRS:1111110000000000:Freeing context structure
*Mar  1 00:05:52.407: GPRS:1111110000000000:Session timer stopped
*Mar  1 00:05:52.407: PPP-REGEN state counters: pending counter is 0
*Mar  1 00:05:52.407:           State[IDLE] counter is 0
*Mar  1 00:05:52.407:           State[AUTHORIZING] counter is 0
*Mar  1 00:05:52.407:           State[VPDN CONNECTING] counter is 0
*Mar  1 00:05:52.407:           State[PPP NEGOTIATING] counter is 0
*Mar  1 00:05:52.407:           State[PPP CONNECTED] counter is 0
*Mar  1 00:05:52.407:           State[PPP TERMINATING] counter is 0
*Mar  1 00:05:52.407: GPRS:1111110000000000:PPP regen context 0x6219F4BC released
*Mar  1 00:05:52.407: GPRS:GTP-PPP-REGEN context magic(0x619D4FBC) invalid
*Mar  1 00:05:52.407: %LINK-3-UPDOWN: Interface Virtual-Access5, changed state to down
*Mar  1 00:05:53.399: %LINEPROTO-5-UPDOWN: Line protocol on Interface Virtual-Access5,
changed state to down
```

Example 6

The following example displays both debug events and details related to PPP regeneration processing as the GGSN clears a PDP context request :

```
Router# debug gprs gtp-director events
Router# debug gprs gtp-director details
*Mar  1 00:06:34.907: PPP-REGEN state counters: pending counter is 0
*Mar  1 00:06:34.907:           State[IDLE] counter is 0
*Mar  1 00:06:34.907:           State[AUTHORIZING] counter is 0
*Mar  1 00:06:34.907:           State[VPDN CONNECTING] counter is 0
*Mar  1 00:06:34.907:           State[PPP NEGOTIATING] counter is 0
*Mar  1 00:06:34.907:           State[PPP CONNECTED] counter is 1
*Mar  1 00:06:34.907:           State[PPP TERMINATING] counter is 0
*Mar  1 00:06:34.907: GPRS:1111110000000000:PPP regen current state PPP CONNECTED
*Mar  1 00:06:34.907: GPRS:1111110000000000:GTP disconnecting the PPP regen session
*Mar  1 00:06:34.907: GPRS:1111110000000000:Processing Disconnect PPP regen from reqQ
*Mar  1 00:06:34.907: GPRS:1111110000000000:got event [CANCEL REGEN'ED PPP] in state [PPP
CONNECTED]
*Mar  1 00:06:34.907: PPP-REGEN state counters: pending counter is 1
*Mar  1 00:06:34.907:           State[IDLE] counter is 0
*Mar  1 00:06:34.907:           State[AUTHORIZING] counter is 0
*Mar  1 00:06:34.907:           State[VPDN CONNECTING] counter is 0
*Mar  1 00:06:34.907:           State[PPP NEGOTIATING] counter is 0
*Mar  1 00:06:34.907:           State[PPP CONNECTED] counter is 0
*Mar  1 00:06:34.907:           State[PPP TERMINATING] counter is 1
*Mar  1 00:06:34.907: GPRS:1111110000000000:state [PPP CONNECTED->PPP TERMINATING] on event
 [CANCEL REGEN'ED PPP]
*Mar  1 00:06:34.907: GPRS:1111110000000000:Cancel request after VPND tunnel is up
*Mar  1 00:06:34.911: GPRS:1111110000000000:PPP down
*Mar  1 00:06:34.911: GPRS:1111110000000000:got event [PPP FAILED] in state [PPP TERMINATING]
*Mar  1 00:06:34.915: PPP-REGEN state counters: pending counter is 1
*Mar  1 00:06:34.915:           State[IDLE] counter is 1
*Mar  1 00:06:34.915:           State[AUTHORIZING] counter is 0
*Mar  1 00:06:34.915:           State[VPDN CONNECTING] counter is 0
*Mar  1 00:06:34.915:           State[PPP NEGOTIATING] counter is 0
*Mar  1 00:06:34.915:           State[PPP CONNECTED] counter is 0
*Mar  1 00:06:34.915:           State[PPP TERMINATING] counter is 0
*Mar  1 00:06:34.915: GPRS:1111110000000000:state [PPP TERMINATING->IDLE] on event [PPP
FAILED]
*Mar  1 00:06:34.915: GPRS:1111110000000000:PPP failed negotiation
*Mar  1 00:06:34.915: GPRS:1111110000000000:got event [CLEANUP CONTEXT] in state [IDLE]
*Mar  1 00:06:34.915: GPRS:1111110000000000:VPDN to inform PPP regen: DISCONNECTED
*Mar  1 00:06:34.915: GPRS:1111110000000000:got event [VPDN DISCONNECTED] in state [IDLE]
*Mar  1 00:06:34.915: GPRS:1111110000000000:state [IDLE->IDLE] on event [CLEANUP CONTEXT]
*Mar  1 00:06:34.915: GPRS:1111110000000000:Freeing context structure
*Mar  1 00:06:34.915: GPRS:1111110000000000:Session timer stopped
*Mar  1 00:06:34.915: PPP-REGEN state counters: pending counter is 0
*Mar  1 00:06:34.915:           State[IDLE] counter is 0
*Mar  1 00:06:34.915:           State[AUTHORIZING] counter is 0
*Mar  1 00:06:34.915:           State[VPDN CONNECTING] counter is 0
*Mar  1 00:06:34.915:           State[PPP NEGOTIATING] counter is 0
*Mar  1 00:06:34.915:           State[PPP CONNECTED] counter is 0
*Mar  1 00:06:34.915:           State[PPP TERMINATING] counter is 0
*Mar  1 00:06:34.915: GPRS:1111110000000000:PPP regen context 0x62196E10 released
*Mar  1 00:06:34.915: GPRS:GTP-PPP-REGEN context magic(0x619D4FBC) invalid
*Mar  1 00:06:34.915: %LINK-3-UPDOWN: Interface Virtual-Access3, changed state to down
```

```
*Mar  1 00:06:35.907: %LINEPROTO-5-UPDOWN: Line protocol on Interface Virtual-Access3,
changed state to down
```

debug gprs radius

To display information about Remote Access Dial-In User Service (RADIUS) processing on the GGSN, use the **debug gprs radius**privileged EXEC command. To disable debugging output, use the **no** form of this command.

debug gprs radius

no debug gprs radius

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values.

Command History

Release	Modification
12.2(4)MX	This command was introduced.
12.2(8)YD	This command was incorporated in Cisco IOS Release 12.2(8)YD.
12.2(8)YW	This command was incorporated in Cisco IOS Release 12.2(8)YW.
12.3(2)XB	This command was incorporated in Cisco IOS Release 12.3(2)XB.
12.3(8)XU	This command was incorporated in Cisco IOS Release 12.3(8)XU.
12.3(11)YJ	This command was incorporated in Cisco IOS Release 12.3(11)YJ.
12.3(14)YQ	This command was incorporated in Cisco IOS Release 12.3(14)YQ.

Usage Guidelines

This command is useful for system operators and development engineers if problems are encountered with communication between a RADIUS server and the GGSN.

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

Examples

The following example enables the display of debug messages related to RADIUS processing on the GGSN :

```
Router# debug gprs radius
```

debug gprs redundancy

To display debug messages, errors, events, or packets related to GTP session redundancy (GTP-SR), use the **debug gprs redundancy** privileged EXEC command. To disable debugging output, use the **no** form of this command.

debug gprs redundancy [**debug**| **errors**| **events**| **packets**]

no debug gprs redundancy [**debug**| **errors**| **events**| **packets**]

Syntax Description

debug	Displays debug messages related to GTP-SR.
errors	Displays errors related to GTP-SR.
events	Displays events related to GTP-SR.
packets	Displays packets related to GTP-SR packets.

Command Default

Disabled.

Command Modes

Global configuration (config)

Command History

Release	Modification
12.3(11)YJ	This command was introduced.
12.3(14)YQ	This command was incorporated in Cisco IOS Release 12.3(14)YQ.
12.4(9)T	This command was integrated into Cisco IOS Release 12.4(9)T.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.

Usage Guidelines

This command displays debug level messages, errors, events, or packets for GTP-SR. It is useful for system operators and development engineers if problems are encountered with communication between the two GGSNs configured as an redundant pair and on which GTP-SR is enabled.

Examples

Example 1

The following sample outputs is for a GGSN failover and switchover of Standby to Active. There is no PDP context involved in this debug collection.

Active GGSN:

```
Router-a#show gprs redundancy
GPRS redundancy is enabled and Unit-Status is Standby
Redundancy Transport Infrastructure status
  Redundancy Infrastructure state:              STANDBY HOT
  Peer Redundancy Infrastructure state:         ACTIVE
  GGSN Redundancy system up since:              21:29:21 EDT Aug 19 2000
  Time of last switchover:                      never
  Total Number of Switchovers:                  2
GPRS Redundancy Statistics
  Last cleared: never
  CheckPointed-From-Active Statistics
    Total Number of Messages:                   129
      Number of Context Setup messages:         19
      Number of Context Modify messages:        3
      Number of Context Remove messages:        19
      Number of Path Setup messages:            34
      Number of Path Modify messages:           5
      Number of Path Remove messages:           34
      Number of CGF Ready messages:             1
      Number of CGF Modify messages:            0
      Number of CGF Remove messages:            0
      Number of Internal State messages:        7
Router-a#debug gprs redundancy
GPRS CF packets debugging is on
GPRS CF events debugging is on
GPRS CF errors debugging is on
GPRS CF debug debugging is on
Router-a#
Router-a#
Router-a#
MWAM 10/2: 000064: Jun  1 2006 18:19:00.975 EDT: %HSRP-6-STATECHANGE: GigabitEthernet0/0.1100
 Grp 51 state Standby -> Active
MWAM 10/2: 000065: Jun  1 2006 18:19:00.975 EDT: GTP-SR:
RF_Status=403-RF_STATUS_MAINTENANCE_ENABLE  RFState=9-ACTIVE-FAST operand=0
RFPeerState=13-ACTIVE
MWAM 10/2: 000066: Jun  1 2006 18:19:00.979 EDT: GTP-SR: RF_Event=200-RF_PROG_ACTIVE_FAST
 RFState=9-ACTIVE-FAST operand=0 RFPeerState=13-ACTIVE
MWAM 10/2: 000067: Jun  1 2006 18:19:00.979 EDT: GTP-SR: Received RF Progression Active
Fast
MWAM 10/2: 000068: Jun  1 2006 18:19:00.979 EDT: GTP-SR: RF_Event=201-RF_PROG_ACTIVE_DRAIN
  RFState=10-ACTIVE-DRAIN operand=0 RFPeerState=13-ACTIVE
MWAM 10/2: 000069: Jun  1 2006 18:19:00.979 EDT: GTP-SR: Received RF Progression Active
Drain
MWAM 10/2: 000070: Jun  1 2006 18:19:00.979 EDT: GTP-SR: RF_Event=202-RF_PROG_ACTIVE_PRECONFIG
  RFState=11-ACTIVE_PRECONFIG operand=0 RFPeerState=13-ACTIVE
MWAM 10/2: 000071: Jun  1 2006 18:19:00.979 EDT: GTP-SR: Received RF Progression Active
PreConfig
MWAM 10/2: 000072: Jun  1 2006 18:19:00.979 EDT: GTP-SR:
RF_Event=203-RF_PROG_ACTIVE_POSTCONFIG  RFState=12-ACTIVE_POSTCONFIG operand=0
RFPeerState=13-ACTIVE
MWAM 10/2: 000073: Jun  1 2006 18:19:00.979 EDT: GTP-SR: Received RF Progression Active
PostConfig
MWAM 10/2: 000074: Jun  1 2006 18:19:00.979 EDT: GTP-SR: RF_Event=204-RF_PROG_ACTIVE
RFState=13-ACTIVE operand=0 RFPeerState=13-ACTIVE
MWAM 10/2: 000075: Jun  1 2006 18:19:00.979 EDT: GTP-SR: Received RF Progression Active
MWAM 10/2: 000076: Jun  1 2006 18:19:00.979 EDT: GTP-SR: Start of the Standby-to-Active
transition
MWAM 10/2: 000077: Jun  1 2006 18:19:00.979 EDT: GTP_SR: Old State Standby,Event Active
Fast Received, New State Active
MWAM 10/2: 000078: Jun  1 2006 18:19:00.979 EDT: GTP-SR:Context Type  OWN, Handler Sync,
Context Event OWN Ready, Context Sub Event No Sub Event
MWAM 10/2: 000079: Jun  1 2006 18:19:00.979 EDT: GTP-SR:State of Redundancy Context is
Initialized
MWAM 10/2: 000080: Jun  1 2006 18:19:00.979 EDT: GTP-SR: Event OWN Ready, Sub Event No Sub
 Event
MWAM 10/2: 000081: Jun  1 2006 18:19:00.979 EDT: GTP-SR: Removing element from state-list
Initialized, final count 2
MWAM 10/2: 000082: Jun  1 2006 18:19:00.979 EDT: GTP-SR: adding element in state-list Bulk
 Synch Ready, final count 2
MWAM 10/2: 000083: Jun  1 2006 18:19:00.979 EDT: GTP-SR:Context Type  CGF, Handler Sync,
```

```
Context Event CGF Ready, Context Sub Event No Sub Event
MWAM 10/2: 000084: Jun  1 2006 18:19:00.979 EDT: GTP-SR:State of Redundancy Context is
Initialized
MWAM 10/2: 000085: Jun  1 2006 18:19:00.979 EDT: GTP-SR: Event CGF Ready, Sub Event No Sub
 Event
MWAM 10/2: 000086: Jun  1 2006 18:19:00.979 EDT: GTP-SR: Removing element from state-list
Initialized, final count 1
MWAM 10/2: 000087: Jun  1 2006 18:19:00.979 EDT: GTP-SR: adding element in state-list Bulk
 Synch Ready, final count 3
MWAM 10/2: 000088: Jun  1 2006 18:19:00.979 EDT: GTP-SR: Invalid shdb 0x0
MWAM 10/2: 000089: Jun  1 2006 18:19:00.979 EDT: GTP-SR: Transition CG 10.0.250.114 to
(state 0)
MWAM 10/2: 000090: Jun  1 2006 18:19:00.979 EDT: GTP-SR: Invalid shdb 0x0
MWAM 10/2: 000091: Jun  1 2006 18:19:00.979 EDT: GTP-SR: Transition CG 10.0.250.115 to
(state 0)
MWAM 10/2: 000092: Jun  1 2006 18:19:00.983 EDT: GTP-SR: SHDB AVL tree cleanup to start in
 10 sec
MWAM 10/2: 000093: Jun  1 2006 18:19:00.983 EDT: GTP-SR: Completion of Standby-to-Active
transition
MWAM 10/2: 000094: Jun  1 2006 18:19:00.983 EDT: GTP-SR: Chkpt Status Flow Off Indication
MWAM 10/2: 000095: Jun  1 2006 18:19:00.987 EDT: %HSRP-6-STATECHANGE: GigabitEthernet0/0.301
 Grp 51 state Standby -> Active
MWAM 10/2: 000096: Jun  1 2006 18:19:00.987 EDT: GTP-SR: RF_Status=400-RF_STATUS_PEER_PRESENCE
  RFState=13-ACTIVE operand=0 RFPeerState=13-ACTIVE
MWAM 10/2: 000097: Jun  1 2006 18:19:00.987 EDT: GTP-SR: zero elements to move to other
list
MWAM 10/2: 000098: Jun  1 2006 18:19:00.987 EDT: GTP-SR: zero elements to move to other
list
MWAM 10/2: 000099: Jun  1 2006 18:19:00.987 EDT: GTP-SR: RF_Status=401-RF_STATUS_PEER_COMM
  RFState=13-ACTIVE operand=0 RFPeerState=1-DISABLED
MWAM 10/2: 000100: Jun  1 2006 18:19:01.107 EDT: %HSRP-6-STATECHANGE: GigabitEthernet0/0.1151
 Grp 51 state Standby -> Active
MWAM 10/2: 000101: Jun  1 2006 18:19:01.155 EDT: %HSRP-6-STATECHANGE: GigabitEthernet0/0.250
 Grp 51 state Standby -> Active
MWAM 10/2: 000102: Jun  1 2006 18:19:01.295 EDT: %HSRP-6-STATECHANGE: GigabitEthernet0/0.1101
 Grp 51 state Standby -> Active
MWAM 10/2: 000103: Jun  1 2006 18:19:01.355 EDT: %HSRP-6-STATECHANGE: GigabitEthernet0/0.1251
 Grp 51 state Standby -> Active
MWAM 10/2: 000104: Jun  1 2006 18:19:01.451 EDT: %HSRP-6-STATECHANGE: GigabitEthernet0/0.1201
 Grp 51 state Standby -> Active
MWAM 10/2: 000105: Jun  1 2006 18:19:01.459 EDT: %HSRP-6-STATECHANGE: GigabitEthernet0/0.220
 Grp 51 state Standby -> Active
Router-2#
MWAM 10/2: 000106: Jun  1 2006 18:19:10.983 EDT: GTP-SR: SHDB AVL tree cleanup has 3 nodes
 removed, 0 leftover
Router-a#
Router-a#
Router-a#
MWAM 10/2: 000107: Jun  1 2006 18:20:25.947 EDT: GTP-SR: Chkpt Status Flow Off Indication
MWAM 10/2: 000108: Jun  1 2006 18:20:25.947 EDT: GTP-SR: RF_Status=400-RF_STATUS_PEER_PRESENCE
  RFState=13-ACTIVE operand=1 RFPeerState=1-DISABLED
MWAM 10/2: 000109: Jun  1 2006 18:20:25.947 EDT: GTP-SR: RF_Status=401-RF_STATUS_PEER_COMM
  RFState=13-ACTIVE operand=1 RFPeerState=1-DISABLED
MWAM 10/2: 000110: Jun  1 2006 18:20:25.947 EDT: GTP-SR: RF_Event=300-RF_PROG_PLATFORM_SYNC
  RFState=13-ACTIVE operand=0 RFPeerState=0-UNKNOWN
MWAM 10/2: 000111: Jun  1 2006 18:20:25.947 EDT: GTP-SR: Received RF Progression Platform
Sync
MWAM 10/2: 000112: Jun  1 2006 18:20:53.899 EDT: GTP-SR: RF_Event=102-RF_PROG_STANDBY_CONFIG
  RFState=13-ACTIVE operand=0 RFPeerState=5-STANDBY COLD-CONFIG
MWAM 10/2: 000113: Jun  1 2006 18:20:53.899 EDT: GTP-SR: Received RF Progression Standby
Config
MWAM 10/2: 000114: Jun  1 2006 18:20:53.899 EDT: GTP-SR: RF_Event=103-RF_PROG_STANDBY_FILESYS
  RFState=13-ACTIVE operand=0 RFPeerState=6-STANDBY COLD-FILESYS
MWAM 10/2: 000115: Jun  1 2006 18:20:53.899 EDT: GTP-SR: Received RF Progression Stadnby
Filesys
MWAM 10/2: 000116: Jun  1 2006 18:20:53.899 EDT: GTP-SR: RF_Event=104-RF_PROG_STANDBY_BULK
  RFState=13-ACTIVE operand=0 RFPeerState=7-STANDBY COLD-BULK
MWAM 10/2: 000117: Jun  1 2006 18:20:53.899 EDT: GTP-SR: Received RF Progression Standby
Bulk
MWAM 10/2: 000118: Jun  1 2006 18:20:53.899 EDT: GTP-SR: Active GGSN sending Bulk Sync
finished Msg
MWAM 10/2: 000119: Jun  1 2006 18:20:53.899 EDT: GTP-SR: packing csg_path vaddr: 10.0.250.91
MWAM 10/2: 000120: Jun  1 2006 18:20:53.899 EDT: GTP-SR: packing csg_path port: 4386
```

```
MWAM 10/2: 000121: Jun  1 2006 18:20:53.899 EDT: GTP-SR: packing csg_path state: 1
MWAM 10/2: 000122: Jun  1 2006 18:20:53.899 EDT: GTP-SR: Ckpt Message was sucessfully sent
MWAM 10/2: 000123: Jun  1 2006 18:20:53.899 EDT: GTP-SR: Removing element from state-list
Bulk Synch Ready, final count 2
MWAM 10/2: 000124: Jun  1 2006 18:20:53.899 EDT: GTP-SR: adding element in state-list
Synched, final count 1
MWAM 10/2: 000125: Jun  1 2006 18:20:53.899 EDT: GTP-SR: sync next charging id 0x1C0AA436,
 local rsn 0x6B76EBDE
MWAM 10/2: 000126: Jun  1 2006 18:20:53.899 EDT: GTP-SR: Packing Pair Boot time 21:29:21
EDT Aug 19 2000
MWAM 10/2: 000127: Jun  1 2006 18:20:53.899 EDT: GTP-SR: Packing Switcover Count 3
MWAM 10/2: 000128: Jun  1 2006 18:20:53.899 EDT: GTP-SR: Packing local restart_count 21
MWAM 10/2: 000129: Jun  1 2006 18:20:53.899 EDT: GTP-SR: Ckpt Message was sucessfully sent
MWAM 10/2: 000130: Jun  1 2006 18:20:53.899 EDT: GTP-SR: Removing element from state-list
Bulk Synch Ready, final count 1
MWAM 10/2: 000131: Jun  1 2006 18:20:53.899 EDT: GTP-SR: adding element in state-list
Synched, final count 2
MWAM 10/2: 000132: Jun  1 2006 18:20:53.899 EDT: GTP-SR: sync cgf gw 10.0.250.114, operatemode
 NOT ACTIVE, nextseq 0x7530
MWAM 10/2: 000133: Jun  1 2006 18:20:53.899 EDT: GTP-SR: sync cgf gw 10.0.250.115, operatemode
 NOT ACTIVE, nextseq 0x7530
MWAM 10/2: 000134: Jun  1 2006 18:20:53.899 EDT: GTP-SR: Ckpt Message was sucessfully sent
MWAM 10/2: 000135: Jun  1 2006 18:20:53.899 EDT: GTP-SR: Removing element from state-list
Bulk Synch Ready, final count 0
MWAM 10/2: 000136: Jun  1 2006 18:20:53.899 EDT: GTP-SR: adding element in state-list
Synched, final count 3
MWAM 10/2: 000137: Jun  1 2006 18:20:53.899 EDT: GTP-SR:Active took time of 0 msec to
transfer data for bulk sync
MWAM 10/2: 000138: Jun  1 2006 18:20:53.899 EDT: GTP-SR: Empty list to sync
MWAM 10/2: 000139: Jun  1 2006 18:20:53.899 EDT: GTP-SR: Redundancy RF Event Received is
Standby Bulk Sync End
MWAM 10/2: 000140: Jun  1 2006 18:20:53.899 EDT: GTP-SR: Redundancy Event is Invalid
MWAM 10/2: 000141: Jun  1 2006 18:20:53.899 EDT: GTP-SR: RF_Event=105-RF_PROG_STANDBY_HOT
 RFState=13-ACTIVE operand=0 RFPeerState=8-STANDBY HOT
MWAM 10/2: 000142: Jun  1 2006 18:20:53.899 EDT: GTP-SR: Received RF Progression  Standby
Hot
Router-b
Router-b#show gprs redundancy

GPRS redundancy is enabled and Unit-Status is Active
Redundancy Transport Infrastructure status
  Redundancy Infrastructure state:              ACTIVE
  Peer Redundancy Infrastructure state:         STANDBY HOT
  GGSN Redundancy system up since:              21:29:21 EDT Aug 19 2000
  Time of last switchover:
  Total Number of Switchovers:                  3
GPRS Redundancy Statistics
  Last cleared: never
  CheckPointed-To-Standby Statistics
    Total Number of Messages:                   3
      Number of Context Setup messages:         0
      Number of Context Modify messages:        0
      Number of Context Remove messages:        0
      Number of Path Setup messages:            0
      Number of Path Modify messages:           0
      Number of Path Remove messages:           0
      Number of CGF Ready messages:             1
      Number of CGF Modify messages:            0
      Number of CGF Remove messages:            0
      Number of Internal State messages:        1
```

Examples

The following sample outputs is for PDP context setup, prepaid user traffic, and then PDP context teardown. The debug is given for both Active and Standby GGSN. There is no GGSN switchover.

Active GGSN:

```
Router-a#debug gprs redundancy
GPRS CF packets debugging is on
GPRS CF events debugging is on
GPRS CF errors debugging is on
```

```
GPRS CF debug debugging is on
Router-a#show gprs redundancy
GPRS redundancy is enabled and Unit-Status is Active
Redundancy Transport Infrastructure status
  Redundancy Infrastructure state:              ACTIVE
  Peer Redundancy Infrastructure state:         STANDBY HOT
  GGSN Redundancy system up since:              21:29:21 EDT Aug 19 2000
  Time of last switchover:
  Total Number of Switchovers:                  4
GPRS Redundancy Statistics
  Last cleared: never
  CheckPointed-To-Standby Statistics
    Total Number of Messages:                   3
      Number of Context Setup messages:         0
      Number of Context Modify messages:        0
      Number of Context Remove messages:        0
      Number of Path Setup messages:            0
      Number of Path Modify messages:           0
      Number of Path Remove messages:           0
      Number of CGF Ready messages:             1
      Number of CGF Modify messages:            0
      Number of CGF Remove messages:            0
      Number of Internal State messages:        1
Router-a#
MWAM 10/2: 000073: Aug 24 2000 23:18:55.947 EDT: GTP-SR:pdpmcb handle for pdpmcb (0x24D2FC3C)
 is (0x3A000001)
MWAM 10/2: 000074: Aug 24 2000 23:18:55.963 EDT: GTP-SR: Need to allocate redundancy context
MWAM 10/2: 000075: Aug 24 2000 23:18:55.963 EDT: GTP-SR: adding element in state-list
Initialized, final count 2
MWAM 10/2: 000076: Aug 24 2000 23:18:55.963 EDT: GTP-SR: Need to allocate redundancy context
MWAM 10/2: 000077: Aug 24 2000 23:18:55.963 EDT: GTP-SR: adding element in state-list
Initialized, final count 3
MWAM 10/2: 000078: Aug 24 2000 23:18:55.963 EDT: GTP-SR:Context Type  Path, Handler Sync,
Context Event Path Setup, Context Sub Event No Sub Event
MWAM 10/2: 000079: Aug 24 2000 23:18:55.963 EDT: GTP-SR:State of Redundancy Context is
Initialized
MWAM 10/2: 000080: Aug 24 2000 23:18:55.963 EDT: GTP-SR: Event Path Setup, Sub Event No Sub
 Event
MWAM 10/2: 000081: Aug 24 2000 23:18:55.963 EDT: GTP-SR: Removing element from state-list
Initialized, final count 2
MWAM 10/2: 000082: Aug 24 2000 23:18:55.963 EDT: GTP-SR: adding element in state-list Dynamic
 Sync Ready, final count 1
MWAM 10/2: 000083: Aug 24 2000 23:18:55.963 EDT: GTP-SR: Need to allocate redundancy context
MWAM 10/2: 000084: Aug 24 2000 23:18:55.963 EDT: GTP-SR: adding element in state-list
Initialized, final count 3
MWAM 10/2: 000085: Aug 24 2000 23:18:55.963 EDT: GTP-SR:Context Type  Path, Handler Sync,
Context Event Path Setup, Context Sub Event No Sub Event
MWAM 10/2: 000086: Aug 24 2000 23:18:55.963 EDT: GTP-SR:State of Redundancy Context is
Initialized
MWAM 10/2: 000087: Aug 24 2000 23:18:55.963 EDT: GTP-SR: Event Path Setup, Sub Event No Sub
 Event
MWAM 10/2: 000088: Aug 24 2000 23:18:55.963 EDT: GTP-SR: Removing element from state-list
Initialized, final count 2
MWAM 10/2: 000089: Aug 24 2000 23:18:55.963 EDT: GTP-SR: adding element in state-list Dynamic
 Sync Ready, final count 2
MWAM 10/2: 000090: Aug 24 2000 23:18:55.963 EDT: GTP-SR:packing pathcb->gtpv 1
MWAM 10/2: 000091: Aug 24 2000 23:18:55.963 EDT: GTP-SR:Local IP address 166.11.0.11, and
port 2123
MWAM 10/2: 000092: Aug 24 2000 23:18:55.963 EDT: GTP-SR:Remote IP address 10.10.50.3, and
port 2123
MWAM 10/2: 000093: Aug 24 2000 23:18:55.963 EDT: GTP-SR:packing pathcb->num_data_socks 0
MWAM 10/2: 000094: Aug 24 2000 23:18:55.963 EDT: GTP-SR:packing pathcb->flags 9
MWAM 10/2: 000095: Aug 24 2000 23:18:55.963 EDT: GTP-SR:packing pathcb->restart_count_remote
 1
MWAM 10/2: 000096: Aug 24 2000 23:18:55.963 EDT: GTP-SR: Different lengths during path
create: allowed: 63, packed: 23
MWAM 10/2: 000097: Aug 24 2000 23:18:55.963 EDT: GTP-SR: Ckpt Message was sucessfully sent
MWAM 10/2: 000098: Aug 24 2000 23:18:55.963 EDT: GTP-SR: Removing element from state-list
Dynamic Sync Ready, final count 1
MWAM 10/2: 000099: Aug 24 2000 23:18:55.963 EDT: GTP-SR: adding element in state-list
Synched, final count 4
MWAM 10/2: 000100: Aug 24 2000 23:18:55.963 EDT: GTP-SR:packing pathcb->gtpv 1
MWAM 10/2: 000101: Aug 24 2000 23:18:55.963 EDT: GTP-SR:Local IP address 166.11.0.11, and
```

```
port 2152
MWAM 10/2: 000102: Aug 24 2000 23:18:55.963 EDT: GTP-SR:Remote IP address 10.10.50.3, and
port 2152
MWAM 10/2: 000103: Aug 24 2000 23:18:55.967 EDT: GTP-SR:packing pathcb->num_data_socks 0
MWAM 10/2: 000104: Aug 24 2000 23:18:55.967 EDT: GTP-SR:packing pathcb->flags 8
MWAM 10/2: 000105: Aug 24 2000 23:18:55.967 EDT: GTP-SR:packing pathcb->restart_count_remote
 0
MWAM 10/2: 000106: Aug 24 2000 23:18:55.967 EDT: GTP-SR: Different lengths during path
create: allowed: 63, packed: 23
MWAM 10/2: 000107: Aug 24 2000 23:18:55.967 EDT: GTP-SR: Ckpt Message was sucessfully sent
MWAM 10/2: 000108: Aug 24 2000 23:18:55.967 EDT: GTP-SR: Removing element from state-list
Dynamic Sync Ready, final count 0
MWAM 10/2: 000109: Aug 24 2000 23:18:55.967 EDT: GTP-SR: adding element in state-list
Synched, final count 5
MWAM 10/2: 000110: Aug 24 2000 23:18:55.967 EDT: GTP-SR: Empty list to sync
MWAM 10/2: 000111: Aug 24 2000 23:18:55.967 EDT: GTP-SR: Empty list to sync
MWAM 10/2: 000112: Aug 24 2000 23:19:01.583 EDT: GTP-SR: Creating red context for category
 ID 4 username 100000000000000 on APN ms-apn
MWAM 10/2: 000113: Aug 24 2000 23:19:01.583 EDT: GTP-SR: Need to allocate redundancy context
MWAM 10/2: 000114: Aug 24 2000 23:19:01.583 EDT: GTP-SR: adding element in state-list
Initialized, final count 3
MWAM 10/2: 000115: Aug 24 2000 23:19:01.583 EDT: GTP-SR: Removing element from state-list
Initialized, final count 2
MWAM 10/2: 000116: Aug 24 2000 23:19:01.583 EDT: GTP-SR: adding element in state-list
Synched, final count 6
MWAM 10/2: 000117: Aug 24 2000 23:19:01.583 EDT: GPRS:0100000000000050:shdb 0x95000008
created for category 4 (handle 0xD0000001)
MWAM 10/2: 000118: Aug 24 2000 23:19:01.591 EDT: GTP-SR: Don't checkpoint QP4QR Clear for
Create/Update after a Quota Push Resp
MWAM 10/2: 000119: Aug 24 2000 23:19:01.591 EDT: GTP-SR:Context Type  PDP, Handler Sync,
Context Event Context Setup, Context Sub Event No Sub Event
MWAM 10/2: 000120: Aug 24 2000 23:19:01.591 EDT: GTP-SR:State of Redundancy Context is
Initialized
MWAM 10/2: 000121: Aug 24 2000 23:19:01.591 EDT: GTP-SR: Event Context Setup, Sub Event No
 Sub Event
MWAM 10/2: 000122: Aug 24 2000 23:19:01.591 EDT: GTP-SR: Removing element from state-list
Initialized, final count 1
MWAM 10/2: 000123: Aug 24 2000 23:19:01.591 EDT: GTP-SR: adding element in state-list Dynamic
 Sync Ready, final count 1
MWAM 10/2: 000124: Aug 24 2000 23:19:01.591 EDT: GTP-SR: for pdpmcb: 221 bytes to be packed
MWAM 10/2: 000125: Aug 24 2000 23:19:01.591 EDT: GTP-SR: pdpmcb bitmap = 14346
MWAM 10/2: 000126: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing pdpmcb->user-name
91100000000000000
MWAM 10/2: 000127: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing pdpmcb->msisdn
91010000000000000F000
MWAM 10/2: 000128: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing pdpmcb->selection_mode 0
MWAM 10/2: 000129: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing pdpmcb->remove_staticIP 0
MWAM 10/2: 000130: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing pdpmcb->llcframenum 0
MWAM 10/2: 000131: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing pdpmcb->idle_timeout 3600
MWAM 10/2: 000132: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing pdpmcb->session_timeout 0
MWAM 10/2: 000133: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing pdpmcb->pdpmcb_handle
973078529
MWAM 10/2: 000134: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing pdpmcb->shdb 2080374789
MWAM 10/2: 000135: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing apn_name ms-apn
MWAM 10/2: 000136: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing apnvalue ms-apn
MWAM 10/2: 000137: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing pdpmcb->teid 4194305
MWAM 10/2: 000138: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing pdpmcb->imsi 01000000000000F0
MWAM 10/2: 000139: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing
pdpmcb->pdpaddr.static_addr_allocated  0
MWAM 10/2: 000140: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing
pdpmcb->pdpaddr.dynamic_addr_allocated  1
MWAM 10/2: 000141: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing
pdpmcb->pdpaddr.dynamic_addr_requested  1
MWAM 10/2: 000142: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing pdpmcb->pdpaddr.addr_source
  3
MWAM 10/2: 000143: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing
pdpmcb->pdpaddr.allocated_prefix_len  16
MWAM 10/2: 000144: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing
pdpmcb->pdpaddr.aggregate_prefix_len  16
MWAM 10/2: 000145: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing pdpmcb->pdpaddr.pdp_type_org
  1
MWAM 10/2: 000146: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing pdpmcb->pdpaddr.pdp_type_num
  33
```

```
MWAM 10/2: 000147: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing pdpmcb->pdpaddr.addrlen  6
MWAM 10/2: 000148: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing pdpmcb-ggsn_addr_si
166.11.0.11
MWAM 10/2: 000149: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing pdpmcb-ggsn_addr_data
166.11.0.11
MWAM 10/2: 000150: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing pdpmcb->msisdn_len
9nGTP-SR:packing aaa charging profile index -1,
MWAM 10/2: 000151: Aug 24 2000 23:19:01.591 EDT: GTP-SR:pdpmcb encoded len_t 0
MWAM 10/2: 000152: Aug 24 2000 23:19:01.591 EDT: GTP-SR: pdpcb bitmap = 0
MWAM 10/2: 000153: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing pdpcb->teid_cntl_remote 1
MWAM 10/2: 000154: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing pdpcb->teid_data_local
4194306
MWAM 10/2: 000155: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing pdpcb->teid_data_remote
1000
MWAM 10/2: 000156: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing pdpcb->tid 0100000000000050
MWAM 10/2: 000157: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing naspi = 5
MWAM 10/2: 000158: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing pdpcb->internal_flags 9175041
MWAM 10/2: 000159: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing pdpcb->mnrgflag 0
MWAM 10/2: 000160: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing pdpcb->open_cdr_sent 0
MWAM 10/2: 000161: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing pdpcb->charging_reserved 0

MWAM 10/2: 000162: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing pdpcb->pri 1
MWAM 10/2: 000163: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing pdpcb->fastswitchable 0
MWAM 10/2: 000164: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing pdpcb-sgsn_addr_sig 10.10.50.3
MWAM 10/2: 000165: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing pdpcb-sgsn_addr_data
10.10.50.3
MWAM 10/2: 000166: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing pdpcb->sequence_sig 1
MWAM 10/2: 000167: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing pdpcb->fl_sig_up 0
MWAM 10/2: 000168: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing pdpcb->fl_data1_up 0
MWAM 10/2: 000169: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing pdpcb->fl_sig_down 0
MWAM 10/2: 000170: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing pdpcb->fl_data1_down 0
MWAM 10/2: 000171: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing pdpcb->fl_data2 0
MWAM 10/2: 000172: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing pdpcb->cause 128
MWAM 10/2: 000173: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing pdpcb->restart_count 0
MWAM 10/2: 000174: Aug 24 2000 23:19:01.591 EDT: GTP-SR: packing pdpcb->create_time Aug 24
 2000 23:18:56
MWAM 10/2: 000175: Aug 24 2000 23:19:01.591 EDT: GTP-SR: packing pdpcb->last_access_time
Aug 24 2000 23:18:56
MWAM 10/2: 000176: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing
pdpcb->gtpv1_qos_req.qos_profile 152109353l
MWAM 10/2: 000177: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing
pdpcb->gtpv1_qos_neg.qos_profile 152109353l
MWAM 10/2: 000178: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing pdpcb->teid_cntl_remote 1
MWAM 10/2: 000179: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing pdpcb->teid_data_local
4194306
MWAM 10/2: 000180: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing pdpcb->teid_data_remote
1000
MWAM 10/2: 000181: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing pdpcb->charging_id 471179447
MWAM 10/2: 000182: Aug 24 2000 23:19:01.591 EDT: GTP-SR:packing pdpcb->cdr_recseqnum 0
MWAM 10/2: 000183: Aug 24 2000 23:19:01.591 EDT: GTP-SR: packing of pdpcb->reorder_required
 FF
MWAM 10/2: 000184: Aug 24 2000 23:19:01.591 EDT: GPRS:0100000000000050:
GTP-SR: Successfully pack PDP
MWAM 10/2: 000185: Aug 24 2000 23:19:01.591 EDT: GTP-SR: rulebase ID MS packed
MWAM 10/2: 000186: Aug 24 2000 23:19:01.591 EDT: GTP-SR: cc_session ccfh 0 failover_supported
 1 reqnum 1 packed
MWAM 10/2: 000187: Aug 24 2000 23:19:01.591 EDT: GTP-SR: cc_session dest_host
ips-clcis1.cisco.com dest_realm cisco.com packed
MWAM 10/2: 000188: Aug 24 2000 23:19:01.591 EDT: GTP-SR: category ID 4 packed:
MWAM 10/2: 000189: Aug 24 2000 23:19:01.591 EDT: GTP-SR:    sync data len 164
MWAM 10/2: 000190: Aug 24 2000 23:19:01.591 EDT: GTP-SR:    active shdb 0x95000008
MWAM 10/2: 000191: Aug 24 2000 23:19:01.591 EDT: GTP-SR:    CSG session ID 27599459844129
MWAM 10/2: 000192: Aug 24 2000 23:19:01.591 EDT: GTP-SR:    chrg last svc rec seqnum 0
MWAM 10/2: 000193: Aug 24 2000 23:19:01.591 EDT: GTP-SR:    category state AUTHORIZED
MWAM 10/2: 000194: Aug 24 2000 23:19:01.591 EDT: GTP-SR:    category state trigger flags
0x3
MWAM 10/2: 000195: Aug 24 2000 23:19:01.591 EDT: GTP-SR:    category sub flags 0x0
MWAM 10/2: 000196: Aug 24 2000 23:19:01.591 EDT: GTP-SR:    sync flag 0x0
MWAM 10/2: 000197: Aug 24 2000 23:19:01.591 EDT: GTP-SR:    quotas  included
MWAM 10/2: 000198: Aug 24 2000 23:19:01.591 EDT: GTP-SR:    last req timestamp 0
MWAM 10/2: 000199: Aug 24 2000 23:19:01.591 EDT: GTP-SR:    last req seqnum 0
MWAM 10/2: 000200: Aug 24 2000 23:19:01.595 EDT: GTP-SR: Ckpt Message was sucessfully sent
MWAM 10/2: 000201: Aug 24 2000 23:19:01.595 EDT: GTP-SR: Removing element from state-list
```

```
Dynamic Sync Ready, final count 0
MWAM 10/2: 000202: Aug 24 2000 23:19:01.595 EDT: GTP-SR: adding element in state-list
Synched, final count 7
MWAM 10/2: 000203: Aug 24 2000 23:19:01.595 EDT: GTP-SR: Empty list to sync
MWAM 10/2: 000204: Aug 24 2000 23:19:03.939 EDT: GTP-SR:Context Type  PDP, Handler Sync,
Context Event Context Setup, Context Sub Event No Sub Event
MWAM 10/2: 000205: Aug 24 2000 23:19:03.939 EDT: GTP-SR:State of Redundancy Context is
Synched
MWAM 10/2: 000206: Aug 24 2000 23:19:03.939 EDT: GTP-SR: Event Context Setup, Sub Event No
 Sub Event
MWAM 10/2: 000207: Aug 24 2000 23:19:04.463 EDT: GTP-SR: Checkpoint SGSN init deletion via
 a category before final MCB deletion
MWAM 10/2: 000208: Aug 24 2000 23:19:04.463 EDT: GTP-SR:Context Type  Category, Handler
Update, Context Event Category update, Context Sub Event No Sub Event
MWAM 10/2: 000209: Aug 24 2000 23:19:04.463 EDT: GTP-SR:State of Redundancy Context is
Synched
MWAM 10/2: 000210: Aug 24 2000 23:19:04.463 EDT: GTP-SR: Event Category update, Sub Event
No Sub Event
MWAM 10/2: 000211: Aug 24 2000 23:19:04.463 EDT: GTP-SR: MCB internal flags 0x5802 packed
MWAM 10/2: 000212: Aug 24 2000 23:19:04.463 EDT: GTP-SR: cc_session reqnum 1 packed
MWAM 10/2: 000213: Aug 24 2000 23:19:04.463 EDT: GTP-SR: category ID 4 packed:
MWAM 10/2: 000214: Aug 24 2000 23:19:04.463 EDT: GTP-SR:    sync data len 52
MWAM 10/2: 000215: Aug 24 2000 23:19:04.463 EDT: GTP-SR:    active shdb 0x95000008
MWAM 10/2: 000216: Aug 24 2000 23:19:04.463 EDT: GTP-SR:    CSG session ID 27599459844129
MWAM 10/2: 000217: Aug 24 2000 23:19:04.463 EDT: GTP-SR:    chrg last svc rec seqnum 0
MWAM 10/2: 000218: Aug 24 2000 23:19:04.463 EDT: GTP-SR:    category state
PENDING_SERVICE_STOP
MWAM 10/2: 000219: Aug 24 2000 23:19:04.463 EDT: GTP-SR:    category state trigger flags
0x3
MWAM 10/2: 000220: Aug 24 2000 23:19:04.463 EDT: GTP-SR:    category sub flags 0x0
MWAM 10/2: 000221: Aug 24 2000 23:19:04.463 EDT: GTP-SR:    sync flag 0xA
MWAM 10/2: 000222: Aug 24 2000 23:19:04.463 EDT: GTP-SR:    quotas not included
MWAM 10/2: 000223: Aug 24 2000 23:19:04.463 EDT: GTP-SR:    last req timestamp 0
MWAM 10/2: 000224: Aug 24 2000 23:19:04.463 EDT: GTP-SR:    last req seqnum 0
MWAM 10/2: 000225: Aug 24 2000 23:19:04.463 EDT: GTP-SR: Different lengths during category
 sync: allowed 188, packed 56
MWAM 10/2: 000226: Aug 24 2000 23:19:04.463 EDT: GTP-SR: Ckpt Message was sucessfully sent
MWAM 10/2: 000227: Aug 24 2000 23:19:04.467 EDT: GTP-SR: Checkpoint final MCB deletion after
 sending a CCR Final
MWAM 10/2: 000228: Aug 24 2000 23:19:04.467 EDT: GTP-SR:Context Type  PDP, Handler Delete,
 Context Event Context Remove, Context Sub Event No Sub Event
MWAM 10/2: 000229: Aug 24 2000 23:19:04.467 EDT: GTP-SR:State of Redundancy Context is
Synched
MWAM 10/2: 000230: Aug 24 2000 23:19:04.467 EDT: GTP-SR: Event Context Remove, Sub Event
No Sub Event
MWAM 10/2: 000231: Aug 24 2000 23:19:04.467 EDT: GTP-SR: Checkpoint final MCB deletion
MWAM 10/2: 000232: Aug 24 2000 23:19:04.467 EDT: GTP-SR:Context Type  PDP, Handler Delete,
 Context Event Context Remove, Context Sub Event No Sub Event
MWAM 10/2: 000233: Aug 24 2000 23:19:04.467 EDT: GTP-SR:State of Redundancy Context is
Synched
MWAM 10/2: 000234: Aug 24 2000 23:19:04.467 EDT: GTP-SR: Event Context Remove, Sub Event
No Sub Event
MWAM 10/2: 000235: Aug 24 2000 23:19:04.467 EDT: GTP-SR: Different lengths during PDP delete:
 allowed: 40, packed: 0
MWAM 10/2: 000236: Aug 24 2000 23:19:04.467 EDT: GTP-SR: Ckpt Message was sucessfully sent
MWAM 10/2: 000237: Aug 24 2000 23:19:04.467 EDT: GTP-SR: Removing element from state-list
Synched, final count 6
MWAM 10/2: 000238: Aug 24 2000 23:19:04.467 EDT: GTP-SR: adding element in state-list Delete,
 final count 1
MWAM 10/2: 000239: Aug 24 2000 23:19:04.467 EDT: GTP-SR: Removing element from state-list
Delete, final count 0
MWAM 10/2: 000240: Aug 24 2000 23:19:04.467 EDT: GTP-SR: No redundancy context for sending
 a down event to standby
MWAM 10/2: 000241: Aug 24 2000 23:19:04.471 EDT: GTP-SR: Removing element from state-list
Synched, final count 5
Router-a#
Router-a#
```

Standby GGSN:

```
Router-b#debug gprs redundancy

GPRS CF packets debugging is on
```

```
GPRS CF events debugging is on
GPRS CF errors debugging is on
GPRS CF debug debugging is on
Router-b#sh gprs redun
GPRS redundancy is enabled and Unit-Status is Standby
Redundancy Transport Infrastructure status
  Redundancy Infrastructure state:              STANDBY HOT
  Peer Redundancy Infrastructure state:         ACTIVE
  GGSN Redundancy system up since:              21:29:21 EDT Aug 19 2000
  Time of last switchover:                      never
  Total Number of Switchovers:                  4
GPRS Redundancy Statistics
  Last cleared: never
  CheckPointed-From-Active Statistics
    Total Number of Messages:                   3
      Number of Context Setup messages:         0
      Number of Context Modify messages:        0
      Number of Context Remove messages:        0
      Number of Path Setup messages:            0
      Number of Path Modify messages:           0
      Number of Path Remove messages:           0
      Number of CGF Ready messages:             1
      Number of CGF Modify messages:            0
      Number of CGF Remove messages:            0
      Number of Internal State messages:        1
Router-b#
MWAM 10/2: 000065: Jun  1 2006 18:28:06.591 EDT: GTP-SR: Redundancy RF Event Received is
Create Redundancy Context
MWAM 10/2: 000066: Jun  1 2006 18:28:06.591 EDT: GTP-SR: Redundancy Event is Path Setup
MWAM 10/2: 000067: Jun  1 2006 18:28:06.591 EDT: GTP-SR: Need to allocate redundancy context
MWAM 10/2: 000068: Jun  1 2006 18:28:06.591 EDT: GTP-SR: adding element in state-list
Initialized, final count 4
MWAM 10/2: 000069: Jun  1 2006 18:28:06.591 EDT: GTP-SR Packet Dump: Len for dump: org_len=63,
 len=63
MWAM 10/2: 000070: Jun  1 2006 18:28:06.591 EDT:  1  0  0  0  0  0  0  0  9  1  A6 B  0  B
  8  4B
MWAM 10/2: 000071: Jun  1 2006 18:28:06.591 EDT:  A  A  32 3  8  4B 1  0  0  0  0  0  0  0
  0  0
MWAM 10/2: 000072: Jun  1 2006 18:28:06.591 EDT:  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  0  0
MWAM 10/2: 000073: Jun  1 2006 18:28:06.595 EDT:  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  0
MWAM 10/2: 000074: Jun  1 2006 18:28:06.595 EDT: GTP-SR: un-packing u_path->gtpv 1
MWAM 10/2: 000075: Jun  1 2006 18:28:06.595 EDT: GTP-SR: Local IP address 166.11.0.11, and
 port 2123
MWAM 10/2: 000076: Jun  1 2006 18:28:06.595 EDT: GTP-SR: Remote IP address 10.10.50.3, and
 port 2123
MWAM 10/2: 000077: Jun  1 2006 18:28:06.595 EDT: GTP-SR: un-packing u_path->num_data_socks
 0
MWAM 10/2: 000078: Jun  1 2006 18:28:06.595 EDT: GTP-SR: un-packing u_path->flags 9
MWAM 10/2: 000079: Jun  1 2006 18:28:06.595 EDT: GTP-SR: un-packing restart_count_remote 1
MWAM 10/2: 000080: Jun  1 2006 18:28:06.595 EDT: GTP-SR:Context Type  Path, Handler Sync,
Context Event Path Setup, Context Sub Event No Sub Event
MWAM 10/2: 000081: Jun  1 2006 18:28:06.595 EDT: GTP-SR:State of Redundancy Context is
Initialized
MWAM 10/2: 000082: Jun  1 2006 18:28:06.595 EDT: GTP-SR: Event Path Setup, Sub Event No Sub
 Event
MWAM 10/2: 000083: Jun  1 2006 18:28:06.595 EDT: GTP-SR: Removing element from state-list
Initialized, final count 3
MWAM 10/2: 000084: Jun  1 2006 18:28:06.595 EDT: GTP-SR: adding element in state-list Bulk
 Synch Ready, final count 2
MWAM 10/2: 000085: Jun  1 2006 18:28:06.595 EDT: GTP-SR: Redundancy RF Event Received is
Create Redundancy Context
MWAM 10/2: 000086: Jun  1 2006 18:28:06.595 EDT: GTP-SR: Redundancy Event is Path Setup
MWAM 10/2: 000087: Jun  1 2006 18:28:06.595 EDT: GTP-SR: Need to allocate redundancy context
MWAM 10/2: 000088: Jun  1 2006 18:28:06.595 EDT: GTP-SR: adding element in state-list
Initialized, final count 4
MWAM 10/2: 000089: Jun  1 2006 18:28:06.595 EDT: GTP-SR Packet Dump: Len for dump: org_len=63,
 len=63
MWAM 10/2: 000090: Jun  1 2006 18:28:06.595 EDT:  1  0  0  0  0  0  0  0  8  1  A6 B  0  B
  8  68
MWAM 10/2: 000091: Jun  1 2006 18:28:06.595 EDT:  A  A  32 3  8  68 0  0  0  0  0  0  0  0
  0  0
```

```
MWAM 10/2: 000092: Jun  1 2006 18:28:06.595 EDT:  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  0  0
MWAM 10/2: 000093: Jun  1 2006 18:28:06.595 EDT:  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  0
MWAM 10/2: 000094: Jun  1 2006 18:28:06.595 EDT: GTP-SR: un-packing u_path->gtpv 1
MWAM 10/2: 000095: Jun  1 2006 18:28:06.595 EDT: GTP-SR: Local IP address 166.11.0.11, and
 port 2152
MWAM 10/2: 000096: Jun  1 2006 18:28:06.595 EDT: GTP-SR: Remote IP address 10.10.50.3, and
 port 2152
MWAM 10/2: 000097: Jun  1 2006 18:28:06.595 EDT: GTP-SR: un-packing u_path->num_data_socks
 0
MWAM 10/2: 000098: Jun  1 2006 18:28:06.595 EDT: GTP-SR: un-packing u_path->flags 8
MWAM 10/2: 000099: Jun  1 2006 18:28:06.595 EDT: GTP-SR: un-packing restart_count_remote 0
MWAM 10/2: 000100: Jun  1 2006 18:28:06.595 EDT: GTP-SR:Context Type  Path, Handler Sync,
Context Event Path Setup, Context Sub Event No Sub Event
MWAM 10/2: 000101: Jun  1 2006 18:28:06.595 EDT: GTP-SR:State of Redundancy Context is
Initialized
MWAM 10/2: 000102: Jun  1 2006 18:28:06.595 EDT: GTP-SR: Event Path Setup, Sub Event No Sub
 Event
MWAM 10/2: 000103: Jun  1 2006 18:28:06.595 EDT: GTP-SR: Removing element from state-list
Initialized, final count 3
MWAM 10/2: 000104: Jun  1 2006 18:28:06.595 EDT: GTP-SR: adding element in state-list Bulk
 Synch Ready, final count 3
MWAM 10/2: 000105: Jun  1 2006 18:28:12.223 EDT: GTP-SR: Redundancy RF Event Received is
Create Redundancy Context
MWAM 10/2: 000106: Jun  1 2006 18:28:12.223 EDT: GTP-SR: Redundancy Event is Context Setup
MWAM 10/2: 000107: Jun  1 2006 18:28:12.223 EDT: GTP-SR: Need to allocate redundancy context
MWAM 10/2: 000108: Jun  1 2006 18:28:12.223 EDT: GTP-SR: adding element in state-list
Initialized, final count 4
MWAM 10/2: 000109: Jun  1 2006 18:28:12.223 EDT: GTP-SR Packet Dump: Len for dump:
org_len=755, len=128
MWAM 10/2: 000110: Jun  1 2006 18:28:12.223 EDT:  1  1  39 31 31 30 30 30 30 30 30 30 30
30 30 30
MWAM 10/2: 000111: Jun  1 2006 18:28:12.223 EDT:  30 30 30 0  0  0  0  91 1  0  0  0  0  0
  0  F0
MWAM 10/2: 000112: Jun  1 2006 18:28:12.223 EDT:  0  0  0  0  0  0  E  10 0  0  0  0  0  0
  0  0
MWAM 10/2: 000113: Jun  1 2006 18:28:12.223 EDT:  C0 23 1  8  5  63 69 73 63 6F 31 31 31
63 69 73
MWAM 10/2: 000114: Jun  1 2006 18:28:12.227 EDT:  63 6F 0  7C 0  0  5  0  0  8  0  0  0  0
  0  0
MWAM 10/2: 000115: Jun  1 2006 18:28:12.227 EDT:  0  0  0  40 0  1  1  0  0  0  0  0  0
F0 B  1
MWAM 10/2: 000116: Jun  1 2006 18:28:12.227 EDT:  0  1  0  0  0  0  0  0  0  0  0  0  0  0
  0  0
MWAM 10/2: 000117: Jun  1 2006 18:28:12.227 EDT:  0  0  0  1  1  0  0  0  3  10 10 1  21 0
  6  0
MWAM 10/2: 000118: Jun  1 2006 18:28:12.227 EDT:  ...
MWAM 10/2: 000119: Jun  1 2006 18:28:12.231 EDT: GTP-SR:pdpmcb handle for pdpmcb (0x24AA0CCC)
 is (0x41000001)
MWAM 10/2: 000120: Jun  1 2006 18:28:12.231 EDT: GTP-SR: un-packing # of PDPs packed = 1
MWAM 10/2: 000121: Jun  1 2006 18:28:12.231 EDT: GTP-SR: un-packing pdpmcb->user-name
91100000000000000
MWAM 10/2: 000122: Jun  1 2006 18:28:12.231 EDT: GTP-SR: un-packing pdpmcb->msisdn
91010000000000000F000
MWAM 10/2: 000123: Jun  1 2006 18:28:12.235 EDT: GTP-SR: un-packing pdpmcb->selection_mode
 0
MWAM 10/2: 000124: Jun  1 2006 18:28:12.235 EDT: GTP-SR: un-packing pdpmcb->remove_staticIP
 0
MWAM 10/2: 000125: Jun  1 2006 18:28:12.235 EDT: GTP-SR: un-packing pdpmcb->llcframenum 0
MWAM 10/2: 000126: Jun  1 2006 18:28:12.235 EDT: GTP-SR: un-packing pdpmcb->idle_timeout
3600
MWAM 10/2: 000127: Jun  1 2006 18:28:12.235 EDT: GTP-SR: un-packing pdpmcb->session_timeout
 0
MWAM 10/2: 000128: Jun  1 2006 18:28:12.235 EDT: GTP-SR: pdpmcb bitmap = 30730
MWAM 10/2: 000129: Jun  1 2006 18:28:12.235 EDT: GTP-SR: apn name is ms-apn
MWAM 10/2: 000130: Jun  1 2006 18:28:12.235 EDT: GTP-SR: packing pdpmcb->teid 4194305
MWAM 10/2: 000131: Jun  1 2006 18:28:12.235 EDT: GTP-SR: un-packing pdpmcb->imsi
01000000000000F0
MWAM 10/2: 000132: Jun  1 2006 18:28:12.235 EDT: GTP-SR: un-packing pdpmcb->pdpaddr.pdp_addr
 11.1.0.1
MWAM 10/2: 000133: Jun  1 2006 18:28:12.235 EDT: GTP-SR: un-packing
pdpmcb->pdpaddr.static_addr_allocated 0
```

```
MWAM 10/2: 000134: Jun  1 2006 18:28:12.235 EDT: GTP-SR: un-packing
pdpmcb->pdpaddr.dynamic_addr_allocated 1
MWAM 10/2: 000135: Jun  1 2006 18:28:12.235 EDT: GTP-SR: un-packing
pdpmcb->pdpaddr.dynamic_addr_requested 1
MWAM 10/2: 000136: Jun  1 2006 18:28:12.235 EDT: GTP-SR: un-packing
pdpmcb->pdpaddr.addr_source 3
MWAM 10/2: 000137: Jun  1 2006 18:28:12.235 EDT: GTP-SR: un-packing
pdpmcb->pdpaddr.allocated_prefix_len 16
MWAM 10/2: 000138: Jun  1 2006 18:28:12.235 EDT: GTP-SR: un-packing
pdpmcb->pdpaddr.aggregate_prefix_len 16
MWAM 10/2: 000139: Jun  1 2006 18:28:12.235 EDT: GTP-SR: un-packing
pdpmcb->pdpaddr.pdp_type_org 1
MWAM 10/2: 000140: Jun  1 2006 18:28:12.235 EDT: GTP-SR: un-packing
pdpmcb->pdpaddr.pdp_type_num 33
MWAM 10/2: 000141: Jun  1 2006 18:28:12.235 EDT: GTP-SR: un-packing pdpmcb->pdpaddr.addrlen
 6
MWAM 10/2: 000142: Jun  1 2006 18:28:12.235 EDT: GTP-SR: un-packing pdpmcb->pdpaddr.dhcp_addr
 0.0.0.0
MWAM 10/2: 000143: Jun  1 2006 18:28:12.235 EDT: GTP-SR: un-packing pdpmcb-ggsn_addr_si
166.11.0.11
MWAM 10/2: 000144: Jun  1 2006 18:28:12.235 EDT: GTP-SR: un-packing pdpmcb-ggsn_addr_data
166.11.0.11
MWAM 10/2: 000145: Jun  1 2006 18:28:12.235 EDT: GTP-SR: un-packing pdpmcb->msisdn_len 9
MWAM 10/2: 000146: Jun  1 2006 18:28:12.247 EDT: GTP-SR: Got teid=4194305, as requested
MWAM 10/2: 000147: Jun  1 2006 18:28:12.247 EDT: GTP-SR: un-packing
pdpcb->gtpv1_qos_req.qos_profile 1521093531
MWAM 10/2: 000148: Jun  1 2006 18:28:12.247 EDT: GTP-SR: un-packing
pdpcb->gtpv1_qos_neg.qos_profile 1521093531
MWAM 10/2: 000149: Jun  1 2006 18:28:12.247 EDT: GTP-SR: un-packing pdpcb bitmap = 0
MWAM 10/2: 000150: Jun  1 2006 18:28:12.247 EDT: GTP-SR: un-packing pdpcb->tid0100000000000050
MWAM 10/2: 000151: Jun  1 2006 18:28:12.251 EDT: GTP-SR: un-packing nsapi = 5
MWAM 10/2: 000152: Jun  1 2006 18:28:12.251 EDT: GTP-SR: un-packing pdpcb->internal_flags
9175041
MWAM 10/2: 000153: Jun  1 2006 18:28:12.251 EDT: GTP-SR: un-packing pdpcb->mnrgflag 0
MWAM 10/2: 000154: Jun  1 2006 18:28:12.251 EDT: GTP-SR: un-packing pdpcb->open_cdr_sent
0
MWAM 10/2: 000155: Jun  1 2006 18:28:12.251 EDT: GTP-SR: un-packing pdpcb->charging_reserved
 0
MWAM 10/2: 000156: Jun  1 2006 18:28:12.251 EDT: GTP-SR: un-packing pdpcb->pri  1
MWAM 10/2: 000157: Jun  1 2006 18:28:12.251 EDT: GTP-SR: un-packing pdpcb->fastswitchable
0
MWAM 10/2: 000158: Jun  1 2006 18:28:12.251 EDT: GTP-SR: un-packing pdpcb-sgsn_addr_sig
10.10.50.3
MWAM 10/2: 000159: Jun  1 2006 18:28:12.251 EDT: GTP-SR: un-packing pdpcb-sgsn_addr_data
10.10.50.3
MWAM 10/2: 000160: Jun  1 2006 18:28:12.251 EDT: GTP-SR: un-packing of pdpcb->sequence_sig
 1
MWAM 10/2: 000161: Jun  1 2006 18:28:12.251 EDT: GTP-SR: un-packing of pdpcb->fl_sig_up 0
MWAM 10/2: 000162: Jun  1 2006 18:28:12.251 EDT: GTP-SR: un-packing of pdpcb->fl_data1_up
0
MWAM 10/2: 000163: Jun  1 2006 18:28:12.251 EDT: GTP-SR: un-packing of pdpcb->fl_sig_down
0
MWAM 10/2: 000164: Jun  1 2006 18:28:12.251 EDT: GTP-SR: un-packing of pdpcb->fl_data1_down
 0
MWAM 10/2: 000165: Jun  1 2006 18:28:12.251 EDT: GTP-SR: un-packing of pdpcb->fl_data2 0
MWAM 10/2: 000166: Jun  1 2006 18:28:12.251 EDT: GTP-SR: un-packing of pdpcb->cause 128
MWAM 10/2: 000167: Jun  1 2006 18:28:12.251 EDT: GTP-SR: un-packing of pdpcb->restart_count
 0
MWAM 10/2: 000168: Jun  1 2006 18:28:12.251 EDT: GTP-SR: un-packing of pdpcb->create_time
Apr 13 2006 01:25:25
MWAM 10/2: 000169: Jun  1 2006 18:28:12.251 EDT: GTP-SR: un-packing of pdpcb->last_access_time
 Apr 13 2006 01:25:25
MWAM 10/2: 000170: Jun  1 2006 18:28:12.251 EDT: GTP-SR: unpacking pdpcb->teid_cntl_remote
 1
MWAM 10/2: 000171: Jun  1 2006 18:28:12.251 EDT: GTP-SR: unpacking pdpcb->teid_data_local
4194306
MWAM 10/2: 000172: Jun  1 2006 18:28:12.251 EDT: GTP-SR: unpacking pdpcb->teid_data_remote
 1000
MWAM 10/2: 000173: Jun  1 2006 18:28:12.251 EDT: GTP-SR: unpacking pdpcb->charging_id
471179447
MWAM 10/2: 000174: Jun  1 2006 18:28:12.251 EDT: GTP-SR: unpacking pdpcb->cdr_recseqnum 0
MWAM 10/2: 000175: Jun  1 2006 18:28:12.251 EDT: GTP-SR: un-packing of pdpcb->reorder_required
 FF
```

```
MWAM 10/2: 000176: Jun  1 2006 18:28:12.251 EDT: GTP-SR: We wanted teid 4194306, and got
4194306
MWAM 10/2: 000177: Jun  1 2006 18:28:12.251 EDT: GTP-SR: Got teid 4194306 as requested
MWAM 10/2: 000178: Jun  1 2006 18:28:12.251 EDT: pdp_create_by_tid on standby:tid 100000050,
 pdp 24A90B24
MWAM 10/2: 000179: Jun  1 2006 18:28:12.251 EDT: GPRS:0100000000000050:
GTP-SR: Successfully unpack PDP
MWAM 10/2: 000180: Jun  1 2006 18:28:12.251 EDT: GTP-SR: rulebase ID MS unpacked
MWAM 10/2: 000181: Jun  1 2006 18:28:12.251 EDT: GTP-SR: cc_session ccfh 0 failover_supported
 1 reqnum 1 packed
MWAM 10/2: 000182: Jun  1 2006 18:28:12.251 EDT: GTP-SR: new cc_session dest_host
ips-clcis1.cisco.com unpacked
MWAM 10/2: 000183: Jun  1 2006 18:28:12.251 EDT: GTP-SR: new cc_session dest_realm cisco.com
 unpacked
MWAM 10/2: 000184: Jun  1 2006 18:28:12.251 EDT: GTP-SR: Unpacking 1 categories
MWAM 10/2: 000185: Jun  1 2006 18:28:12.251 EDT: GTP-SR: Unpacking category of ID 4
MWAM 10/2: 000186: Jun  1 2006 18:28:12.255 EDT: GTP-SR: Creating red context for category
 ID 4 username 100000000000000 on APN ms-apn
MWAM 10/2: 000187: Jun  1 2006 18:28:12.255 EDT: GTP-SR: Need to allocate redundancy context
MWAM 10/2: 000188: Jun  1 2006 18:28:12.255 EDT: GTP-SR: adding element in state-list
Initialized, final count 5
MWAM 10/2: 000189: Jun  1 2006 18:28:12.255 EDT: GTP-SR: Removing element from state-list
Initialized, final count 4
MWAM 10/2: 000190: Jun  1 2006 18:28:12.255 EDT: GTP-SR: adding element in state-list
Synched, final count 1
MWAM 10/2: 000191: Jun  1 2006 18:28:12.255 EDT: GPRS:0100000000000050:shdb 0xC6000008
created for category 4 (handle 0xDE000001)
MWAM 10/2: 000192: Jun  1 2006 18:28:12.255 EDT: GTP-SR: red context installed for the new
 category (shdb: active 0x95000008, standby 0xC6000008)
MWAM 10/2: 000193: Jun  1 2006 18:28:12.255 EDT: GTP-SR: new category ID 4 unpacked:
MWAM 10/2: 000194: Jun  1 2006 18:28:12.255 EDT: GTP-SR:    sync data len 164
MWAM 10/2: 000195: Jun  1 2006 18:28:12.255 EDT: GTP-SR:    active shdb 0x95000008
MWAM 10/2: 000196: Jun  1 2006 18:28:12.255 EDT: GTP-SR:    CSG session ID 27599459844129
MWAM 10/2: 000197: Jun  1 2006 18:28:12.255 EDT: GTP-SR:    chrg last svc rec seqnum 0
MWAM 10/2: 000198: Jun  1 2006 18:28:12.255 EDT: GTP-SR:    category state AUTHORIZED
MWAM 10/2: 000199: Jun  1 2006 18:28:12.255 EDT: GTP-SR:    category state trigger flags
0x3
MWAM 10/2: 000200: Jun  1 2006 18:28:12.255 EDT: GTP-SR:    category sub flags 0x0
MWAM 10/2: 000201: Jun  1 2006 18:28:12.255 EDT: GTP-SR:    sync flag 0x0
MWAM 10/2: 000202: Jun  1 2006 18:28:12.255 EDT: GTP-SR:    quotas included
MWAM 10/2: 000203: Jun  1 2006 18:28:12.255 EDT: GTP-SR:    last req timestamp 0
MWAM 10/2: 000204: Jun  1 2006 18:28:12.255 EDT: GTP-SR:    last req seqnum 0
MWAM 10/2: 000205: Jun  1 2006 18:28:12.255 EDT: GTP-SR: address received from active with
 radius source is
MWAM 10/2: 000206: Jun  1 2006 18:28:12.259 EDT: GTP-SR:Context Type  PDP, Handler Sync,
Context Event Context Setup, Context Sub Event No Sub Event
MWAM 10/2: 000207: Jun  1 2006 18:28:12.259 EDT: GTP-SR:State of Redundancy Context is
Initialized
MWAM 10/2: 000208: Jun  1 2006 18:28:12.259 EDT: GTP-SR: Event Context Setup, Sub Event No
 Sub Event
MWAM 10/2: 000209: Jun  1 2006 18:28:12.259 EDT: GTP-SR: Removing element from state-list
Initialized, final count 3
MWAM 10/2: 000210: Jun  1 2006 18:28:12.259 EDT: GTP-SR: adding element in state-list Bulk
 Synch Ready, final count 4
MWAM 10/2: 000211: Jun  1 2006 18:28:15.091 EDT: GTP-SR: Redundancy RF Event Received is
Update Redundancy Context
MWAM 10/2: 000212: Jun  1 2006 18:28:15.091 EDT: GTP-SR: Redundancy Event is Category update
MWAM 10/2: 000213: Jun  1 2006 18:28:15.091 EDT: GTP-SR: red context found (active shdb
0x95000008, standby shdb 0xC6000008)
MWAM 10/2: 000214: Jun  1 2006 18:28:15.091 EDT: GTP-SR Packet Dump: Len for dump:
org_len=188, len=128
MWAM 10/2: 000215: Jun  1 2006 18:28:15.091 EDT:  7C 0  0  5  0  0  58 2  0  0  0  1  0
34 34 0
MWAM 10/2: 000216: Jun  1 2006 18:28:15.091 EDT:  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  0  95
MWAM 10/2: 000217: Jun  1 2006 18:28:15.091 EDT:  0  0  8  0  0  19 1A 0  0  0  21 0  0  0
  0  0
MWAM 10/2: 000218: Jun  1 2006 18:28:15.091 EDT:  0  0  9  0  0  0  3  0  0  A  0  0  0  0
  0  0
MWAM 10/2: 000219: Jun  1 2006 18:28:15.091 EDT:  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  0  0
MWAM 10/2: 000220: Jun  1 2006 18:28:15.095 EDT:  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  0  0
```

```
MWAM 10/2: 000221: Jun  1 2006 18:28:15.095 EDT:  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  0  0
MWAM 10/2: 000222: Jun  1 2006 18:28:15.095 EDT:  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  0  0
MWAM 10/2: 000223: Jun  1 2006 18:28:15.095 EDT:  ...
MWAM 10/2: 000224: Jun  1 2006 18:28:15.095 EDT: GTP-SR: category found with handle 0xDE000001
 shdbs: active 0x95000008 standby 0xC6000008 (MCB shdbs: active 0x7C000005, standby
0xC6000008)
MWAM 10/2: 000225: Jun  1 2006 18:28:15.095 EDT: GTP-SR: MCB internal flags 0x5802 unpacked
MWAM 10/2: 000226: Jun  1 2006 18:28:15.095 EDT: GTP-SR: cc_session reqnum 1 unpacked and
installed
MWAM 10/2: 000227: Jun  1 2006 18:28:15.095 EDT: GTP-SR: Unpacking category of ID 4
MWAM 10/2: 000228: Jun  1 2006 18:28:15.095 EDT: GTP-SR: sync obj created in prep for MCB
deletion
MWAM 10/2: 000229: Jun  1 2006 18:28:15.095 EDT: GTP-SR:  category ID 4 unpacked:
MWAM 10/2: 000230: Jun  1 2006 18:28:15.095 EDT: GTP-SR:    sync data len 52
MWAM 10/2: 000231: Jun  1 2006 18:28:15.095 EDT: GTP-SR:    active shdb 0x95000008
MWAM 10/2: 000232: Jun  1 2006 18:28:15.095 EDT: GTP-SR:    CSG session ID 27599459844129
MWAM 10/2: 000233: Jun  1 2006 18:28:15.095 EDT: GTP-SR:    chrg last svc rec seqnum 0
MWAM 10/2: 000234: Jun  1 2006 18:28:15.095 EDT: GTP-SR:    category state
PENDING_SERVICE_STOP
MWAM 10/2: 000235: Jun  1 2006 18:28:15.095 EDT: GTP-SR:    category state trigger flags
0x3
MWAM 10/2: 000236: Jun  1 2006 18:28:15.095 EDT: GTP-SR:    category sub flags 0x0
MWAM 10/2: 000237: Jun  1 2006 18:28:15.095 EDT: GTP-SR:    sync flag 0xA
MWAM 10/2: 000238: Jun  1 2006 18:28:15.095 EDT: GTP-SR:    quotas not included
MWAM 10/2: 000239: Jun  1 2006 18:28:15.095 EDT: GTP-SR:    last req timestamp 0
MWAM 10/2: 000240: Jun  1 2006 18:28:15.095 EDT: GTP-SR:    last req seqnum 0
MWAM 10/2: 000241: Jun  1 2006 18:28:15.095 EDT: GTP-SR: Redundancy RF Event Received is
Redundancy Context Delete
MWAM 10/2: 000242: Jun  1 2006 18:28:15.095 EDT: GTP-SR: Redundancy Event is Context Remove
MWAM 10/2: 000243: Jun  1 2006 18:28:15.095 EDT: GTP-SR Packet Dump: Len for dump: org_len=40,
 len=40
MWAM 10/2: 000244: Jun  1 2006 18:28:15.095 EDT:  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  0  0
MWAM 10/2: 000245: Jun  1 2006 18:28:15.095 EDT:  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  0  0
MWAM 10/2: 000246: Jun  1 2006 18:28:15.095 EDT:  0  0  0  0  0  0  0  0
MWAM 10/2: 000247: Jun  1 2006 18:28:15.095 EDT: GPRS:GTP-SR: Deleting v1 MCB on the standby
MWAM 10/2: 000248: Jun  1 2006 18:28:15.095 EDT: GPRS:0100000000000050:GTP-SR: Deleting v1
 PDP on the standby
MWAM 10/2: 000249: Jun  1 2006 18:28:15.095 EDT: GTP-SR: MCB deletion sync obj deleted
MWAM 10/2: 000250: Jun  1 2006 18:28:15.095 EDT: GTP-SR: Removing element from state-list
Synched, final count 0
MWAM 10/2: 000251: Jun  1 2006 18:28:15.095 EDT: GTP-SR: Removing element from state-list
Bulk Synch Ready, final count 3
MWAM 10/2: 000252: Jun  1 2006 18:29:15.103 EDT: GTP-SR: Redundancy RF Event Received is
Redundancy Context Delete
MWAM 10/2: 000253: Jun  1 2006 18:29:15.103 EDT: GTP-SR: Redundancy Event is Path Remove
MWAM 10/2: 000254: Jun  1 2006 18:29:15.103 EDT: GTP-SR:Context Type  Path, Handler Delete,
 Context Event Path Remove, Context Sub Event No Sub Event
MWAM 10/2: 000255: Jun  1 2006 18:29:15.103 EDT: GTP-SR:State of Redundancy Context is Bulk
 Synch Ready
MWAM 10/2: 000256: Jun  1 2006 18:29:15.103 EDT: GTP-SR: Event Path Remove, Sub Event No
Sub Event
MWAM 10/2: 000257: Jun  1 2006 18:29:15.103 EDT: GTP-SR: Removing element from state-list
Bulk Synch Ready, final count 2
MWAM 10/2: 000258: Jun  1 2006 18:29:15.103 EDT: GTP-SR: Redundancy RF Event Received is
Redundancy Context Delete
MWAM 10/2: 000259: Jun  1 2006 18:29:15.103 EDT: GTP-SR: Redundancy Event is Path Remove
MWAM 10/2: 000260: Jun  1 2006 18:29:15.103 EDT: GTP-SR:Context Type  Path, Handler Delete,
 Context Event Path Remove, Context Sub Event No Sub Event
MWAM 10/2: 000261: Jun  1 2006 18:29:15.103 EDT: GTP-SR:State of Redundancy Context is Bulk
 Synch Ready
MWAM 10/2: 000262: Jun  1 2006 18:29:15.103 EDT: GTP-SR: Event Path Remove, Sub Event No
Sub Event
MWAM 10/2: 000263: Jun  1 2006 18:29:15.103 EDT: GTP-SR: Removing element from state-list
Bulk Synch Ready, final count 1
```

Related Commands

Command	Description
clear gprs redundancy statistics	Clears statistics related to GTP-SR.
gprs redundancy	Enables GTP-SR on a GGSN.
gprs redundancy charging sync-window cdr rec-seqnum	Configures the window size used to determine when the CDR record sequence number needs to be synchronized to the Standby GGSN.
gprs redundancy charging sync-window gtpp seqnum	Configures the window size used to determine when the GTP' sequence number needs to be synchronized to the Standby GGSN.
show gprs redundancy	Displays statistics related to GTP-SR.

debug gvrp

To display GVRP debugging information, use the **debug gvrp**command in privileged EXEC mode. To disable debugging outpu, use the **no** form of this command.

debug gvrp {**all**| **config**| **error**| **event**| **ha**| **packets**| **switch**}

no debug gvrp

Syntax Description

all	(Optional) Enables all levels of debugging
config	(Optional) Displays user configuration.
error	(Optional) Enables error level debugging.
event	(Optional) Enables event level debugging.
ha	(Optional) Enables ha level debugging.
packets	(Optional) Enables packet level debugging.
switch	(Optional) Enables switch level debugging.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(33)SRB	This command was introduced.

Usage Guidelines

Conditional interface debugging can be used to limit the debugging output messages related to an interface.

Examples

The following example shows how to enable all levels of debugging:

```
debug gvrp all
```

Related Commands

Command	Description
show gvrp interface	Displays details of the adininstrative and operational GVRP states of all or one particular .1Q trunk port in the device.

Command	Description
show gvrp summary	Displays the GVRP configuration at the device leve.

debug h225

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

debug h225 {**asn1**| **events**}

no debug h225

Syntax Description

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

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

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

Usage Guidelines

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

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

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

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

Examples

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

```
Router# debug h225 events
H.225 Event Messages debugging is on
Router#
*Mar  2 02:47:14.689:      H225Lib::h225TConn:connect in progress on socket [2]
*Mar  2 02:47:14.689:      H225Lib::h225TConn:Q.931 Call State is initialized to be
[Null].
*Mar  2 02:47:14.697:Hex representation of the SETUP TPKT to
send.0300004D080200DC05040380C0A36C09913133233133333030700991313423313433303 07E00260500800
60008914A000102004B1F5E5D8990006C0000000005BF7454000C0700000000000000
*Mar  2 02:47:14.701:
*Mar  2 02:47:14.701:      H225Lib::h225SetupRequest:Q.931 SETUP sent from socket [2]
*Mar  2 02:47:14.701:      H225Lib::h225SetupRequest:Q.931 Call State changed to
[Call Initiated].
*Mar  2 02:47:14.729:Hex representation of the received
TPKT03000021080280DC013401017E00120503400600089 14A000100000109350E2B28
*Mar  2 02:47:14.729:
*Mar  2 02:47:14.729:      H225Lib::h225RecvData:Q.931 ALERTING received from socket [2]
*Mar  2 02:47:14.729:      H225Lib::h225RecvData:Q.931 Call State changed to
[Call Delivered].
*Mar  2 02:47:17.565:Hex representation of the received
TPKT03000034080280DC07040380C0A37E00230502400600089 14A0001000109350E2B2802004B1F5E5D899000
6C0000000005BF7454
*Mar  2 02:47:17.569:
*Mar  2 02:47:17.569:      H225Lib::h225RecvData:Q.931 CONNECT received from socket [2]
*Mar  2 02:47:17.569:      H225Lib::h225RecvData:Q.931 Call State changed to [Active].
*Mar  2 02:47:23.273:Hex representation of the received
TPKT0300001A080280DC5A080280107E000A050500060008914A0001
*Mar  2 02:47:23.273:
*Mar  2 02:47:23.273:     H225Lib::h225RecvData:Q.931 RELEASE COMPLETE received from socket
 [2]
*Mar  2 02:47:23.273:      H225Lib::h225RecvData:Q.931 Call State changed to [Null].
*Mar  2 02:47:23.293:Hex representation of the RELEASE COMPLETE TPKT
to send.0300001A080200DC5A080280107E000A050500060008914A0001
*Mar  2 02:47:23.293:
*Mar  2 02:47:23.293:      H225Lib::h225TerminateRequest:Q.931 RELEASE COMPLETE sent from
socket [2]. Call state changed to [Null].
*Mar  2 02:47:23.293:      H225Lib::h225TClose:TCP connection from socket [2] closed
```

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

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

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

```
Router# debug h225 asn1
H.225 ASN1 Messages debugging is on
Router#
value RasMessage ::= admissionRequest :
*Mar  2 02:48:18.445:  {
*Mar  2 02:48:18.445:    requestSeqNum 03320,
*Mar  2 02:48:18.445:    callType pointToPoint :NULL,
*Mar  2 02:48:18.445:    callModel direct :NULL,
*Mar  2 02:48:18.445:    endpointIdentifier "60D6BA4C00000001",
*Mar  2 02:48:18.445:    destinationInfo
*Mar  2 02:48:18.445:    {
*Mar  2 02:48:18.445:      e164 :"14#14300"
*Mar  2 02:48:18.445:    },
*Mar  2 02:48:18.449:    srcInfo
*Mar  2 02:48:18.449:    {
*Mar  2 02:48:18.449:      e164 :"13#13300"
*Mar  2 02:48:18.449:    },
*Mar  2 02:48:18.449:    bandWidth 0640,
*Mar  2 02:48:18.449:    callReferenceValue 0224,
*Mar  2 02:48:18.449:    conferenceID '4B1F5E5D899000720000000005C067A4'H,
*Mar  2 02:48:18.449:    activeMC FALSE,
*Mar  2 02:48:18.449:    answerCall FALSE
*Mar  2 02:48:18.449:  }
*Mar  2 02:48:18.449:25800CF7 00F00036 00300044 00360042 00410034 00430030 00300030
00300030
00300030 00310103 80470476 33010380 46046633 40028000 E04B1F5E 5D899000
72000000 0005C067 A400
29000CF7 40028000 0109350E 06B80077
value RasMessage ::= admissionConfirm :
*Mar  2 02:48:18.469:  {
*Mar  2 02:48:18.469:    requestSeqNum 03320,
*Mar  2 02:48:18.469:    bandWidth 0640,
*Mar  2 02:48:18.469:    callModel direct :NULL,
*Mar  2 02:48:18.469:    destCallSignalAddress ipAddress :
*Mar  2 02:48:18.469:      {
*Mar  2 02:48:18.469:        ip '0109350E'H,
*Mar  2 02:48:18.469:        port 01720
*Mar  2 02:48:18.469:      },
*Mar  2 02:48:18.469:    irrFrequency 0120
*Mar  2 02:48:18.473:  }
*Mar  2 02:48:18.473:value H323-UserInformation ::=
*Mar  2 02:48:18.481:{
*Mar  2 02:48:18.481:  h323-uu-pdu
*Mar  2 02:48:18.481:  {
*Mar  2 02:48:18.481:    h323-message-body setup :
*Mar  2 02:48:18.481:      {
*Mar  2 02:48:18.481:        protocolIdentifier { 0 0 8 2250 0 1 },
*Mar  2 02:48:18.481:        sourceInfo
*Mar  2 02:48:18.481:        {
*Mar  2 02:48:18.481:          terminal
*Mar  2 02:48:18.481:          {
*Mar  2 02:48:18.481:          },
*Mar  2 02:48:18.481:          mc FALSE,
*Mar  2 02:48:18.481:          undefinedNode FALSE
*Mar  2 02:48:18.481:        },
*Mar  2 02:48:18.481:        activeMC FALSE,
*Mar  2 02:48:18.481:        conferenceID '4B1F5E5D899000720000000005C067A4'H,
*Mar  2 02:48:18.481:        conferenceGoal create :NULL,
*Mar  2 02:48:18.485:        callType pointToPoint :NULL,
*Mar  2 02:48:18.485:        sourceCallSignalAddress ipAddress :
```

```
*Mar  2 02:48:18.485:          {
*Mar  2 02:48:18.485:            ip '00000000'H,
*Mar  2 02:48:18.485:            port 00
*Mar  2 02:48:18.485:          }
*Mar  2 02:48:18.485:      }
*Mar  2 02:48:18.485:  }
*Mar  2 02:48:18.485:}
*Mar  2 02:48:18.485:00800600 08914A00 0102004B 1F5E5D89 90007200 00000005 C067A400
0C070000
00000000 00
value H323-UserInformation ::=
*Mar  2 02:48:18.525:{
*Mar  2 02:48:18.525:  h323-uu-pdu
*Mar  2 02:48:18.525:  {
*Mar  2 02:48:18.525:    h323-message-body alerting :
*Mar  2 02:48:18.525:      {
*Mar  2 02:48:18.525:        protocolIdentifier { 0 0 8 2250 0 1 },
*Mar  2 02:48:18.525:        destinationInfo
*Mar  2 02:48:18.525:        {
*Mar  2 02:48:18.525:          mc FALSE,
*Mar  2 02:48:18.525:          undefinedNode FALSE
*Mar  2 02:48:18.525:        },
*Mar  2 02:48:18.525:        h245Address ipAddress :
*Mar  2 02:48:18.525:          {
*Mar  2 02:48:18.525:            ip '0109350E'H,
*Mar  2 02:48:18.525:            port 011050
*Mar  2 02:48:18.525:          }
*Mar  2 02:48:18.525:      }
*Mar  2 02:48:18.525:  }
*Mar  2 02:48:18.525:}
*Mar  2 02:48:18.525:value H323-UserInformation ::=
*Mar  2 02:48:22.753:{
*Mar  2 02:48:22.753:  h323-uu-pdu
*Mar  2 02:48:22.753:  {
*Mar  2 02:48:22.753:    h323-message-body connect :
*Mar  2 02:48:22.753:      {
*Mar  2 02:48:22.753:        protocolIdentifier { 0 0 8 2250 0 1 },
*Mar  2 02:48:22.753:        h245Address ipAddress :
*Mar  2 02:48:22.753:          {
*Mar  2 02:48:22.753:            ip '0109350E'H,
*Mar  2 02:48:22.753:            port 011050
*Mar  2 02:48:22.753:          },
*Mar  2 02:48:22.753:        destinationInfo
*Mar  2 02:48:22.753:        {
*Mar  2 02:48:22.753:          terminal
*Mar  2 02:48:22.753:          {
*Mar  2 02:48:22.753:          },
*Mar  2 02:48:22.757:          mc FALSE,
*Mar  2 02:48:22.757:          undefinedNode FALSE
*Mar  2 02:48:22.757:        },
*Mar  2 02:48:22.757:        conferenceID '4B1F5E5D8990007200000000005C067A4'H
*Mar  2 02:48:22.757:      }
*Mar  2 02:48:22.757:  }
*Mar  2 02:48:22.757:}
*Mar  2 02:48:22.757:value H323-UserInformation ::=
*Mar  2 02:48:27.109:{
*Mar  2 02:48:27.109:  h323-uu-pdu
*Mar  2 02:48:27.109:  {
*Mar  2 02:48:27.109:    h323-message-body releaseComplete :
*Mar  2 02:48:27.109:      {
*Mar  2 02:48:27.109:        protocolIdentifier { 0 0 8 2250 0 1 }
*Mar  2 02:48:27.109:      }
*Mar  2 02:48:27.109:  }
*Mar  2 02:48:27.109:}
*Mar  2 02:48:27.109:value RasMessage ::= disengageRequest :
*Mar  2 02:48:27.117:  {
*Mar  2 02:48:27.117:    requestSeqNum 03321,
*Mar  2 02:48:27.117:    endpointIdentifier "60D6BA4C00000001",
*Mar  2 02:48:27.117:    conferenceID '4B1F5E5D8990007200000000005C067A4'H,
*Mar  2 02:48:27.121:    callReferenceValue 0224,
*Mar  2 02:48:27.121:    disengageReason normalDrop :NULL
*Mar  2 02:48:27.121:  }
*Mar  2 02:48:27.121:3C0CF81E 00360030 00440036 00420041 00340043 00300030 00300030
```

```
00300030
00300031 4B1F5E5D 89900072 00000000 05C067A4 00E020
400CF8
value RasMessage ::= disengageConfirm :
*Mar  2 02:48:27.133:  {
*Mar  2 02:48:27.133:    requestSeqNum 03321
*Mar  2 02:48:27.133:  }
*Mar  2 02:48:27.133:value H323-UserInformation ::=
*Mar  2 02:48:27.133:{
*Mar  2 02:48:27.133:  h323-uu-pdu
*Mar  2 02:48:27.133:  {
*Mar  2 02:48:27.133:    h323-message-body releaseComplete :
*Mar  2 02:48:27.133:      {
*Mar  2 02:48:27.133:        protocolIdentifier { 0 0 8 2250 0 1 }
*Mar  2 02:48:27.133:      }
*Mar  2 02:48:27.133:  }
*Mar  2 02:48:27.133:}
*Mar  2 02:48:27.133:05000600 08914A00 01
```

debug h225 asn1

To display ASN1 contents of RAS an d Q.931 messages, use the **debug h255 asn1** privileged EXEC command. The **no** form of this command disables debugging output.

debug h255 asn1

no debug h255 asn1

Syntax Description

This command has no arguments or keywords.

Command History

Release	Modification
11.3(2)NA	This command was introduced.
12.0(3)T	This command was modified.
12.2(15)T	This command is no longer supported in Cisco IOS Mainline or Technology-based (T) releases. It may continue to appear in Cisco IOS 12.2S-family releases.

Usage Guidelines

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

Examples

Examples

The following output shows two proxy call scenarios. A trace is collected on the gatekeeper with ASN1 turned on. The call is being established.

```
Router# debug h225 asn1
H.225 ASN1 Messages debugging is on
Router#24800006 03C00030 00300036 00380041 00450037 00430030 00300030 00300030
00300030 00310140 0F007000 74006500 6C003200 33004000 7A006F00 6E006500
32002E00 63006F00 6D020180 AAAA4006 00700074 0065006C 00320031 0033401E
0000015F C8490FB4 B9D111BF AF0060B0 00E94500
value RasMessage ::= admissionRequest :
  {
    requestSeqNum 7,
    callType pointToPoint : NULL,
    endpointIdentifier "0068AE7C00000001",
    destinationInfo
    {
      h323-ID : "ptel23@zone2.com"
    },
    srcInfo
    {
      e164 : "7777",
      h323-ID : "ptel213"
    },
```

```
    bandWidth 7680,
    callReferenceValue 1,
    conferenceID '5FC8490FB4B9D111BFAF0060B000E945'H,
    activeMC FALSE,
    answerCall FALSE
  }
value RasMessage ::= admissionConfirm :
  {
    requestSeqNum 7,
    bandWidth 7680,
    callModel direct : NULL,
    destCallSignalAddress ipAddress :
      {
        ip '65000001'H,
        port 1720
      },
    irrFrequency 30
  }
29000006 401E0000 65000001 06B8001D
2480001D 03C00030 00300036 00380041 00390036 00300030 00300030 00300030
00300030 00320140 0F007000 74006500 6C003200 33004000 7A006F00 6E006500
32002E00 63006F00 6D014006 00700074 0065006C 00320031 00334002 8000015F
C8490FB4 B9D111BF AF0060B0 00E94540
value RasMessage ::= admissionRequest :
  {
    requestSeqNum 30,
    callType pointToPoint : NULL,
    endpointIdentifier "0068A960000000002",
    destinationInfo
    {
      h323-ID : "ptel23@zone2.com"
    },
    srcInfo
    {
      h323-ID : "ptel213"
    },
    bandWidth 640,
    callReferenceValue 1,
    conferenceID '5FC8490FB4B9D111BFAF0060B000E945'H,
    activeMC FALSE,
    answerCall TRUE
  }
value ACFnonStandardInfo ::=
{
  srcTerminalAlias
  {
    e164 : "7777",
    h323-ID : "ptel213"
  },
  dstTerminalAlias
  {
    h323-ID : "ptel23@zone2.com"
  },
  dstProxyAlias
  {
    h323-ID : "px2"
  },
  dstProxySignalAddress
  {
    ip '66000001'H,
    port 1720
  }
}
C00203AA AA800600 70007400 65006C00 32003100 3301800F 00700074 0065006C
00320033 0040007A 006F006E 00650032 002E0063 006F006D 01800200 70007800
32660000 0106B8
value RasMessage ::= admissionConfirm :
  {
    requestSeqNum 30,
    bandWidth 7680,
    callModel direct : NULL,
    destCallSignalAddress ipAddress :
      {
```

```
        ip '66000001'H,
        port 1720
      },
    irrFrequency 30,
    nonStandardData
    {
      nonStandardIdentifier h221NonStandard :
        {
          t35CountryCode 181,
          t35Extension 0,
          manufacturerCode 18
        },
      data
'C00203AAAA800600700074006500 6C00320031003301800F007000740065006C003200 ...'H
    }
  }
2980001D 401E0000 66000001 06B8001D 40B50000 1247C002 03AAAA80 06007000
74006500 6C003200 31003301 800F0070 00740065 006C0032 00330040 007A006F
006E0065 0032002E 0063006F 006D0180 02007000 78003266 00000106 B8
24C0001E 03C00030 00300036 00380041 00390036 00300030 00300030 00300030
00300030 00320140 0F007000 74006500 6C003200 33004000 7A006F00 6E006500
32002E00 63006F00 6D006600 000106B8 020180AA AA400600 70007400 65006C00
32003100 33401E00 00435FC8 490FB4B9 D111BFAF 0060B000 E94500
value RasMessage ::= admissionRequest :
  {
    requestSeqNum 31,
    callType pointToPoint : NULL,
    endpointIdentifier "0068A96000000002",
    destinationInfo
    {
      h323-ID : "ptel23@zone2.com"
    },
    destCallSignalAddress ipAddress :
      {
        ip '66000001'H,
        port 1720
      },
    srcInfo
    {
      e164 : "7777",
      h323-ID : "ptel213"
    },
    bandWidth 7680,
    callReferenceValue 67,
    conferenceID '5FC8490FB4B9D111BFAF0060B000E945'H,
    activeMC FALSE,
    answerCall FALSE
  }
value RasMessage ::= admissionConfirm :
  {
    requestSeqNum 31,
    bandWidth 7680,
    callModel direct : NULL,
    destCallSignalAddress ipAddress :
      {
        ip '66000001'H,
        port 1720
      },
    irrFrequency 30
  }
```

Examples

The following output shows two proxy call scenarios. A trace is collected on the source proxy with ASN1 turned on. The call is being torn down

```
Router# debug h225 asn1
H.225 ASN1 Messages debugging is on
Router#
value H323-UserInformation ::=
{
  h323-uu-pdu
```

```
  {
    h323-message-body setup :
      {
        protocolIdentifier { 0 0 8 2250 0 1 },
        sourceAddress
        {
          h323-ID : "ptel213"
        },
        sourceInfo
        {
          terminal
          {
          },
          mc FALSE,
          undefinedNode FALSE
        },
        destinationAddress
        {
          h323-ID : "ptel23@zone2.com"
        },
        activeMC FALSE,
        conferenceID '5FC8490FB4B9D111BFAF0060B000E945'H,
        conferenceGoal create : NULL,
        callType pointToPoint : NULL,
        sourceCallSignalAddress ipAddress :
          {
            ip '3200000C'H,
            port 1720
          }
      }
  }
}
value RasMessage ::= admissionRequest :
  {
    requestSeqNum 30,
    callType pointToPoint : NULL,
    endpointIdentifier "0068A96000000002",
    destinationInfo
    {
      h323-ID : "ptel23@zone2.com"
    },
    srcInfo
    {
      h323-ID : "ptel213"
    },
    bandWidth 640,
    callReferenceValue 1,
    conferenceID '5FC8490FB4B9D111BFAF0060B000E945'H,
    activeMC FALSE,
    answerCall TRUE
  }
2480001D 03C00030 00300036 00380041 00390036 00300030 00300030 00300030
00300030 00320140 0F007000 74006500 6C003200 33004000 7A006F00 6E006500
32002E00 63006F00 6D014006 00700074 0065006C 00320031 00334002 8000015F
C8490FB4 B9D111BF AF0060B0 00E94540
2980001D 401E0000 66000001 06B8001D 40B50000 1247C002 03AAAA80 06007000
74006500 6C003200 31003301 800F0070 00740065 006C0032 00330040 007A006F
006E0065 0032002E 0063006F 006D0180 02007000 78003266 00000106 B8
value RasMessage ::= admissionConfirm :
  {
    requestSeqNum 30,
    bandWidth 7680,
    callModel direct : NULL,
    destCallSignalAddress ipAddress :
      {
        ip '66000001'H,
        port 1720
      },
    irrFrequency 30,
    nonStandardData
    {
      nonStandardIdentifier h221NonStandard :
        {
```

```
          t35CountryCode 181,
          t35Extension 0,
          manufacturerCode 18
        },
      data
'C00203AAAA800600700074006500 6C003200 3100330 1800F00700074006500 6C003200 ...'H
    }
  }
C00203AA AA800600 70007400 65006C00 32003100 3301800F 00700074 0065006C
00320033 0040007A 006F006E 00650032 002E0063 006F006D 01800200 70007800
32660000 0106B8
value ACFnonStandardInfo ::=
{
  srcTerminalAlias
  {
    e164 : "7777",
    h323-ID : "ptel213"
  },
  dstTerminalAlias
  {
    h323-ID : "ptel23@zone2.com"
  },
  dstProxyAlias
  {
    h323-ID : "px2"
  },
  dstProxySignalAddress
  {
    ip '66000001'H,
    port 1720
  }
}
value RasMessage ::= admissionRequest :
  {
    requestSeqNum 31,
    callType pointToPoint : NULL,
    endpointIdentifier "0068A96000000002",
    destinationInfo
    {
      h323-ID : "ptel23@zone2.com"
    },
    destCallSignalAddress ipAddress :
      {
        ip '66000001'H,
        port 1720
      },
    srcInfo
    {
      e164 : "7777",
      h323-ID : "ptel213"
    },
    bandWidth 7680,
    callReferenceValue 67,
    conferenceID '5FC8490FB4B9D111BFAF0060B000E945'H,
    activeMC FALSE,
    answerCall FALSE
  }
24C0001E 03C00030 00300036 00380041 00390036 00300030 00300030 00300030
00300030 00320140 0F007000 74006500 6C003200 33004000 7A006F00 6E006500
32002E00 63006F00 6D006600 000106B8 020180AA AA400600 70007400 65006C00
32003100 33401E00 00435FC8 490FB4B9 D111BFAF 0060B000 E94500
2900001E 401E0000 66000001 06B8001D
value RasMessage ::= admissionConfirm :
  {
    requestSeqNum 31,
    bandWidth 7680,
    callModel direct : NULL,
    destCallSignalAddress ipAddress :
      {
        ip '66000001'H,
        port 1720
      },
    irrFrequency 30
```

```
  }
value H323-UserInformation ::=
{
  h323-uu-pdu
  {
    h323-message-body callProceeding :
      {
        protocolIdentifier { 0 0 8 2250 0 1 },
        destinationInfo
        {
          gateway
          {
            protocol
            {
              h323 :
                {
                }
            }
          },
          mc FALSE,
          undefinedNode FALSE
        }
      }
  }
}
01000600 08914A00 01088001 2800
value H323-UserInformation ::=
{
  h323-uu-pdu
  {
    h323-message-body setup :
      {
        protocolIdentifier { 0 0 8 2250 0 1 },
        sourceAddress
        {
          h323-ID : "ptel213"
        },
        sourceInfo
        {
          vendor
          {
            vendor
            {
              t35CountryCode 181,
              t35Extension 0,
              manufacturerCode 18
            }
          },
          gateway
          {
            protocol
            {
              h323 :
                {
                }
            }
          },
          mc FALSE,
          undefinedNode FALSE
        },
        destinationAddress
        {
          h323-ID : "ptel23@zone2.com"
        },
        destCallSignalAddress ipAddress :
          {
            ip '66000001'H,
            port 1720
          },
        activeMC FALSE,
        conferenceID '5FC8490FB4B9D111BFAF0060B000E945'H,
        conferenceGoal create : NULL,
        callType pointToPoint : NULL,
```

```
        sourceCallSignalAddress ipAddress :
          {
            ip '65000001'H,
            port 1720
          },
        remoteExtensionAddress h323-ID : "ptel23@zone2.com"
      }
  }
}
00B80600 08914A00 01014006 00700074 0065006C 00320031 00332800 B5000012
40012800 01400F00 70007400 65006C00 32003300 40007A00 6F006E00 65003200
2E006300 6F006D00 66000001 06B8005F C8490FB4 B9D111BF AF0060B0 00E94500
0E070065 00000106 B822400F 00700074 0065006C 00320033 0040007A 006F006E
00650032 002E0063 006F006D
value H323-UserInformation ::=
{
  h323-uu-pdu
  {
    h323-message-body callProceeding :
      {
        protocolIdentifier { 0 0 8 2250 0 1 },
        destinationInfo
        {
          gateway
          {
            protocol
            {
              h323 :
                {
                }
            }
          },
          mc FALSE,
          undefinedNode FALSE
        }
      }
  }
}
value H323-UserInformation ::=
{
  h323-uu-pdu
  {
    h323-message-body alerting :
      {
        protocolIdentifier { 0 0 8 2250 0 1 },
        destinationInfo
        {
          mc FALSE,
          undefinedNode FALSE
        }
      }
  }
}
value H323-UserInformation ::=
{
  h323-uu-pdu
  {
    h323-message-body alerting :
      {
        protocolIdentifier { 0 0 8 2250 0 1 },
        destinationInfo
        {
          mc FALSE,
          undefinedNode FALSE
        }
      }
  }
}
03000600 08914A00 010000
value H323-UserInformation ::=
{
  h323-uu-pdu
  {
```

```
    h323-message-body connect :
      {
        protocolIdentifier { 0 0 8 2250 0 1 },
        h245Address ipAddress :
          {
            ip '66000001'H,
            port 11011
          },
        destinationInfo
        {
          gateway
          {
            protocol
            {
              h323 :
                {
                }
            }
          },
          mc FALSE,
          undefinedNode FALSE
        },
        conferenceID '5FC8490FB4B9D111BFAF0060B000E945'H
      }
  }
}
value H323-UserInformation ::=
{
  h323-uu-pdu
  {
    h323-message-body connect :
      {
        protocolIdentifier { 0 0 8 2250 0 1 },
        h245Address ipAddress :
          {
            ip '65000001'H,
            port 11007
          },
        destinationInfo
        {
          gateway
          {
            protocol
            {
              h323 :
                {
                }
            }
          },
          mc FALSE,
          undefinedNode FALSE
        },
        conferenceID '5FC8490FB4B9D111BFAF0060B000E945'H
      }
  }
}
02400600 08914A00 01006500 00012AFF 08800128 005FC849 0FB4B9D1 11BFAF00
60B000E9 45
```

Example 3

The following output shows two proxy call scenarios. A trace is collected on a destination router where both destination proxy and destination Gatekeeper coexist. Both RAS and H.225 traces are enabled for one complete call.

```
px2#
      RASLib::RASRecvData: successfully rcvd message of length 80 from 40.0.0.33:1585
      RASLib::RASRecvData: LRQ rcvd from [40.0.0.33:1585] on sock [6880372]
      RASlib::ras_sendto: msg length 111 sent to 40.0.0.33
      RASLib::RASSendLCF: LCF sent to 40.0.0.33
      H225Lib::h225TAccept: TCP connection accepted from 101.0.0.1:11002 on
```

```
socket [2]
      H225Lib::h225TAccept: Q.931 Call State is initialized to be [Null].
Hex representation of the received TPKT
030000A60802008005040488988CA56C0591373737377E008D0500B8060008914A000101400
60070007400650060C0032003100332800B5000012400128000140 0F007000740065006C00320
0330040007A006F006E00650032002E0063006F006D006600000106B8003DC8490FB4B9D111B
FAF0060B000E945000E07006500000106B822400F007000740065006C003200330040007A006
F006E00650032002E0063006F006D
      H225Lib::h225RecvData: Q.931 SETUP received from socket [2]
      H225Lib::h225RecvData: State changed to [Call Present].
      RASlib::ras_sendto: msg length 119 sent to 102.0.0.1
      RASLib::RASSendARQ: ARQ sent to 102.0.0.1
      RASLib::RASRecvData: successfully rcvd message of length 119 from 102.0.0.1:24999
      RASLib::RASRecvData: ARQ rcvd from [102.0.0.1:24999] on sock [0x68FC74]
      RASlib::ras_sendto: msg length 16 sent to 70.0.0.31
      RASLib::RASSendACF: ACF sent to 70.0.0.31
      RASLib::RASRecvData: successfully rcvd message of length 16 from 102.0.0.1:1719
      RASLib::RASRecvData: ACF rcvd from [102.0.0.1:1719] on sock [0x67E6A4]
      RASlib::ras_sendto: msg length 119 sent to 102.0.0.1
      RASLib::RASSendARQ: ARQ sent to 102.0.0.1
      RASLib::RASRecvData: successfully rcvd message of length 119 from 102.0.0.1:24999
      RASLib::RASRecvData: ARQ rcvd from [102.0.0.1:24999] on sock [0x68FC74]
      RASlib::ras_sendto: msg length 16 sent to 70.0.0.31
      RASLib::RASSendACF: ACF sent to 70.0.0.31
      RASLib::RASRecvData: successfully rcvd message of length 16 from 102.0.0.1:1719
      RASLib::RASRecvData: ACF rcvd from [102.0.0.1:1719] on sock [0x67E6A4]
Hex representation of the CALL PROCEEDING TPKT to send.
0300001B08028080027E000F050100060008914A00010880012800
      H225Lib::h225CallProcRequest: Q.931 CALL PROCEEDING sent from socket
[2]. Call state remains unchanged (Q.931 FSM simplified for H.225.0)
      H225Lib::h225TConn: connect in progress on socket [4]
      H225Lib::h225TConn: Q.931 Call State is initialized to be [Null].
Hex representation of the SETUP TPKT to send.
030000A50802008005040388C0A56C0591373737377E008D0500B8060008914A00010140060
07000740065006C0032003100332800B5000012400128000140 0F007000740065006C0032003
30040007A006F006E00650032002E0063006F006D005A00000D06B8003DC8490FB4B9D111BFA
F0060B000E945000E07006600000106B822400F007000740065006C003200330040007A006F0
06E00650032002E0063006F006D
      H225Lib::h225SetupRequest: Q.931 SETUP sent from socket [4]
      H225Lib::h225SetupRequest: Q.931 Call State changed to [Call Initiated].
      RASLib::RASRecvData: successfully rcvd message of length 123 from 90.0.0.13:1700
      RASLib::RASRecvData: ARQ rcvd from [90.0.0.13:1700] on sock [0x68FC74]
      RASlib::ras_sendto: msg length 16 sent to 90.0.0.13
      RASLib::RASSendACF: ACF sent to 90.0.0.13
Hex representation of the received TPKT
0300001808028080027E000C050100060008914A00010200
      H225Lib::h225RecvData: Q.931 CALL PROCEEDING received from socket [4]
Hex representation of the received TPKT
0300001808028080017E000C050300060008914A00010200
      H225Lib::h225RecvData: Q.931 ALERTING received from socket [4]
      H225Lib::h225RecvData: Q.931 Call State changed to [Call Delivered].
Hex representation of the ALERTING TPKT to send.
0300001808028080017E000C050300060008914A00010000
      H225Lib::h225AlertRequest: Q.931 ALERTING sent from socket [2]. Call
state changed to [Call Received].
Hex representation of the received TPKT
0300003508028080070404889886A57E00230502400600008914A0001005A00000D06A402003
DC8490FB4B9D111BFAF0060B000E945
      H225Lib::h225RecvData: Q.931 CONNECT received from socket [4]
      H225Lib::h225RecvData: Q.931 Call State changed to [Active].
Hex representation of the CONNECT TPKT to send.
0300003708028080070403 88C0A57E00260502400600008914A000100660000012AFC0880012
8003DC8490FB4B9D111BFAF0060B000E945
      H225Lib::h225SetupResponse: Q.931 CONNECT sent from socket [2]
      H225Lib::h225SetupResponse: Q.931 Call State changed to [Active].
      RASlib::ras_sendto: msg length 108 sent to 102.0.0.1
      RASLib::RASSendIRR: IRR sent to 102.0.0.1
      RASLib::RASRecvData: successfully rcvd message of length 108 from 102.0.0.1:24999
      RASLib::RASRecvData: IRR rcvd from [102.0.0.1:24999] on sock [0x68FC74]
      RASLib::RASRecvData: successfully rcvd message of length 101 from 90.0.0.13:1700
      RASLib::RASRecvData: IRR rcvd from [90.0.0.13:1700] on sock [0x68FC74]
Hex representation of the received TPKT
0300001A080280805A080280107E000A050500060008914A0001
```

```
      H225Lib::h225RecvData: Q.931 RELEASE COMPLETE received from socket [2]
      H225Lib::h225RecvData: Q.931 Call State changed to [Null].
      RASlib::ras_sendto: msg length 55 sent to 102.0.0.1
      RASLib::RASSendDRQ: DRQ sent to 102.0.0.1
      H225Lib::h225RecvData: no connection on socket [2]
      RASLib::RASRecvData: successfully rcvd message of length 55 from 102.0.0.1:24999
      RASLib::RASRecvData: DRQ rcvd from [102.0.0.1:24999] on sock [0x68FC74]
      RASlib::ras_sendto: msg length 3 sent to 70.0.0.31
      RASLib::RASSendDCF: DCF sent to 70.0.0.31
Hex representation of the RELEASE COMPLETE TPKT to send.
0300001A080280805A080280107E000A050500060008914A0001
     H225Lib::h225TerminateRequest: Q.931 RELEASE COMPLETE sent from socket [2]. Call state
 changed to [Null].
      H225Lib::h225TClose: TCP connection from socket [2] closed
      RASlib::ras_sendto: msg length 55 sent to 102.0.0.1
      RASLib::RASSendDRQ: DRQ sent to 102.0.0.1
      RASLib::RASRecvData: successfully rcvd message of length 3 from 102.0.0.1:1719
      RASLib::RASRecvData: DCF rcvd from [102.0.0.1:1719] on sock [0x67E6A4]
      RASLib::RASRecvData: successfully rcvd message of length 55 from 102.0.0.1:24999
      RASLib::RASRecvData: DRQ rcvd from [102.0.0.1:24999] on sock [0x68FC74]
      RASlib::ras_sendto: msg length 3 sent to 70.0.0.31
      RASLib::RASSendDCF: DCF sent to 70.0.0.31
      RASLib::RASRecvData: successfully rcvd message of length 3 from 102.0.0.1:1719
      RASLib::RASRecvData: DCF rcvd from [102.0.0.1:1719] on sock [0x67E6A4]
Hex representation of the RELEASE COMPLETE TPKT to send.
0300001A080280805A080280107E000A050500060008914A0001
     H225Lib::h225TerminateRequest: Q.931 RELEASE COMPLETE sent from socket [4]. Call state
 changed to [Null].
      H225Lib::h225TClose: TCP connection from socket [4] closed
      RASLib::RASRecvData: successfully rcvd message of length 55 from 90.0.0.13:1700
      RASLib::RASRecvData: DRQ rcvd from [90.0.0.13:1700] on sock [0x68FC74]
      RASlib::ras_sendto: msg length 3 sent to 90.0.0.13
      RASLib::RASSendDCF: DCF sent to 90.0.0.13
```

debug h225 events

To display Q.931 events, use the **debug h225 events** privileged EXEC command. The **no** form of this command disables debugging output.

debug h225 events

no debug h255 events

Syntax Description

This command has no arguments or keywords.

Command History

Release	Modification
11.3(2)NA	This command was introduced.
12.0(3)T	This command was modified.
12.2(15)T	This command is no longer supported in Cisco IOS Mainline or Technology-based (T) releases. It may continue to appear in Cisco IOS 12.2S-family releases.

Examples

The following are sample output from the **debug h225 events** command.

Examples

The following output shows two proxy call scenarios. A trace is collected on the source proxy with H.225 turned on. The call is being established.

```
Router# debug h225 events
H.225 Event Messages debugging is on
Router#  H225Lib::h225TAccept: TCP connection accepted from 50.0.0.12:1701 on
socket [2]
      H225Lib::h225TAccept: Q.931 Call State is initialized to be [Null].
Hex representation of the received TPKT
0300007408020001050404889886A56C0580373737377E005B0500B0060008914A000101400
6007000740065006C00320031003302000140 0F007000740065006C003200330040007A006F0
06E00650032002E0063006F006D004EC8490FB4B9D111BFAF0060B000E945000C07003200000
C06B8
      H225Lib::h225RecvData: Q.931 SETUP received from socket [2]
      H225Lib::h225RecvData: State changed to [Call Present].
Hex representation of the CALL PROCEEDING TPKT to send.
0300001B08028001027E000F050100060008914A00010880012800
      H225Lib::h225CallProcRequest: Q.931 CALL PROCEEDING sent from socket
[2]. Call state remains unchanged (Q.931 FSM simplified for H.225.0)
      H225Lib::h225TConn: connect in progress on socket [4]
      H225Lib::h225TConn: Q.931 Call State is initialized to be [Null].
Hex representation of the SETUP TPKT to send.
030000A6080200840504048898 8CA56C0591373737377E008D0500B8060008914A000101400
6007000740065006C0032003100332800B500001240012800014 00F007000740065006C00320
0330040007A006F006E00650032002E0063006F006D006600000106B8004EC8490FB4B9D111B
FAF0060B000E945000E07006500000106B822400F007000740065006C003200330040007A006
F006E00650032002E0063006F006D
      H225Lib::h225SetupRequest: Q.931 SETUP sent from socket [4]
      H225Lib::h225SetupRequest: Q.931 Call State changed to [Call Initiated].
Hex representation of the received TPKT
```

```
0300001B08028084027E000F050100060008914A00010880012800
      H225Lib::h225RecvData: Q.931 CALL PROCEEDING received from socket [4]
Hex representation of the received TPKT
0300001808028084017E000C050300060008914A00010000
      H225Lib::h225RecvData: Q.931 ALERTING received from socket [4]
      H225Lib::h225RecvData: Q.931 Call State changed to [Call Delivered].
Hex representation of the ALERTING TPKT to send.
0300001808028001017E000C050300060008914A00010000
      H225Lib::h225AlertRequest: Q.931 ALERTING sent from socket [2]. Call
state changed to [Call Received].
Hex representation of the received TPKT
0300003708028084070403 88C0A57E00260502400600 08914A000100660000012AFF0880012
8004EC8490FB4B9D111BFAF0060B000E945
      H225Lib::h225RecvData: Q.931 CONNECT received from socket [4]
      H225Lib::h225RecvData: Q.931 Call State changed to [Active].
Hex representation of the CONNECT TPKT to send.
0300003808028001070404889886A57E00260502400600 08914A000100650000012AFC08800
128004EC8490FB4B9D111BFAF0060B000E945
      H225Lib::h225SetupResponse: Q.931 CONNECT sent from socket [2]
      H225Lib::h225SetupResponse: Q.931 Call State changed to [Active].
```

Examples

The following output shows two proxy call scenarios. A trace is collected on the source proxy with H.225 turned on. The call is being torn down.

```
Router# debug h225 events
H.225 Event Messages debugging is on
Router#
Hex representation of the received TPKT
0300001A080200015A080200907E000A050500060008914A0001
      H225Lib::h225RecvData: Q.931 RELEASE COMPLETE received from socket [2]
      H225Lib::h225RecvData: Q.931 Call State changed to [Null].
      H225Lib::h225RecvData: no connection on socket [2]
Hex representation of the RELEASE COMPLETE TPKT to send.
0300001A080280015A080280107E000A050500060008914A0001
     H225Lib::h225TerminateRequest: Q.931 RELEASE COMPLETE sent from socket [2]. Call state
 changed to [Null].
      H225Lib::h225TClose: TCP connection from socket [2] closed
Hex representation of the RELEASE COMPLETE TPKT to send.
0300001A080280845A080280107E000A050500060008914A0001
     H225Lib::h225TerminateRequest: Q.931 RELEASE COMPLETE sent from socket [4]. Call state
 changed to [Null].
      H225Lib::h225TClose: TCP connection from socket [4] closed
```

debug h245 asn1

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

debug h245 asn1

no debug h245 asn1

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

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

Usage Guidelines

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

debug h245 events

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

debug h245 events

no debug h245 events

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

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

debug h245 srtp

To display H.245 Secure Real-Time Transport Protocol (SRTP) messages, use the **debug h245 srtp**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug h245 srtp

no debug h245 srtp

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.4(6)T1	This command was introduced.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.

Usage Guidelines

To reduce the system impact of the output that this command generates, use the **debug h245 srtp** command during times of minimal system traffic. To reduce system overhead and redirect logging to an internal buffer, use the **logging buffered** command.

Use the **debug h225 asn1** command to display Abstract Syntax Notation One (ASN.1) contents of H.225 messages.

Use the **debug h245 asn1** command to display ASN.1 contents of H.245 messages.

The output from the **debug h245 srtp** command is primarily used by Cisco technical personnel. See the “Examples” section for an explanation of selected fields.

Examples

The following example displays SRTP messages exchanged during H.225 and H.245 signaling:

```
Router# debug h245 srtp
H.245 SRTP Messages debugging is on
SY3725_1#
000072: Mar 16 16:46:38.237: //-1/xxxxxxxxxxxx/H323/cch323_post_call_setup_request:
cch323_post_call_setup_request:1015: SRTP added to ev for stream:SRTP_TX_STREAM.
000073: Mar 16 16:46:38.237: //-1/xxxxxxxxxxxx/H323/cch323_dump_srtp_caps:
crypto_bm=0x1,
codec_bm=0xB.
000074: Mar 16 16:46:38.237: //-1/xxxxxxxxxxxx/H323/cch323_dump_srtp_media_params:
```

The following lines show SRTP media parameters:

```
ssrc_sel=0x0,
srtp_ssrc=0x0,
crypto_services=0x0,
```

```
crypto_suite=0x0,
master_key_len=0x0,
master_salt_len=0x0,
master_key=0xxxxxxxxxxxxxxxx,
master_salt=0xxxxxxxxxxxxxx
000075: Mar 16 16:46:38.237: //-1/xxxxxxxxxxx/H323/cch323_dump_srtp_sess_params:
```

The following lines show SRTP session parameters:

```
lifetime=0x0,
mki=0x,
mki_length=0x0,
kdr=0x0,
encryptedsrtp=0x0,
encryptedsrtcp=0x0,
fecorder=0x0,
windowsizehint=0x0
000076: Mar 16 16:46:38.237: cch323_post_call_setup_request:993:
callID = 0x4PeercallID = 0x3params->incomingCallid = 0x3srtp_params_caps = 0x0i=1
000077: Mar 16 16:46:38.237: //-1/xxxxxxxxxxx/H323/cch323_post_call_setup_request:
cch323_post_call_setup_request:1021: SRTP NOT added to ev
000078: Mar 16 16:46:38.241: //4/CCB23DEA8003/H323/cch323_set_srtp_call:
../voip/cch323/gw/os/src/h323_gw_dialpeer.c:cch323_set_srtp_call:3420: SRTP configuration:
```

The following lines show SRTP configuration. Fields having a value set to 1 show that SRTP is enabled.

```
peer->voice_peer_tag = 101, ccb->srtp_call = 1, ccb->srtp_fallback = 1, ccb->srtp_transparent
 = 1
000079: Mar 16 16:46:38.241: //4/CCB23DEA8003/H323/cch323_get_caps_chn_info: Local[TX] SRTP
 Info:../voip/cch323/gw/src/cch323_h245_iwf_util.c:cch323_get_caps_chn_info:148
000080: Mar 16 16:46:38.241: //4/CCB23DEA8003/H323/cch323_dump_srtp_caps:
```

The following lines show SRTP capabilities:

```
crypto_bm=0x1,
codec_bm=0xB.
000081: Mar 16 16:46:38.241: //4/CCB23DEA8003/H323/cch323_dump_srtp_media_params:
ssrc_sel=0x0,
srtp_ssrc=0x0,
crypto_services=0x0,
crypto_suite=0x0,
master_key_len=0x0,
master_salt_len=0x0,
master_key=0xxxxxxxxxxxxxxxx,
master_salt=0xxxxxxxxxxxxxx
000082: Mar 16 16:46:38.241: //4/CCB23DEA8003/H323/cch323_dump_srtp_sess_params:
lifetime=0x0,
mki=0x,
mki_length=0x0,
kdr=0x0,
encryptedsrtp=0x0,
encryptedsrtcp=0x0,
fecorder=0x0,
windowsizehint=0x0
000083: Mar 16 16:46:38.241: //4/CCB23DEA8003/H323/cch323_get_caps_chn_info: Remote[RX]
SRTP Info:../voip/cch323/gw/src/cch323_h245_iwf_util.c:cch323_get_caps_chn_info:151
000084: Mar 16 16:46:38.241: //4/CCB23DEA8003/H323/cch323_dump_srtp_caps:
```

The following line shows SRTP capability for the DSP, indicated by the crypto_bm field. A value of 0x0 indicates the DSP is not SRTP capable.

```
crypto_bm=0x0,
codec_bm=0x0.
000085: Mar 16 16:46:38.241: //4/CCB23DEA8003/H323/cch323_dump_srtp_media_params:
ssrc_sel=0x0,
srtp_ssrc=0x0,
crypto_services=0x0,
crypto_suite=0x0,
master_key_len=0x0,
master_salt_len=0x0,
master_key=0xxxxxxxxxxxxxxxx,
master_salt=0xxxxxxxxxxxxxx
```

```
000086: Mar 16 16:46:38.241: //4/CCB23DEA8003/H323/cch323_dump_srtp_sess_params:
lifetime=0x0,
mki=0x,
mki_length=0x0,
kdr=0x0,
encryptedsrtp=0x0,
encryptedsrtcp=0x0,
fecorder=0x0,
windowsizehint=0x0
000087: Mar 16 16:46:38.245: //4/CCB23DEA8003/H323/cch323_build_local_encoded_fastStartOLCs:

../voip/cch323/gw/src/cch323_h245_util.c:cch323_build_local_encoded_fastStartOLCs:1518:
OGW: generating Keys.
000088: Mar 16 16:46:38.245: //4/CCB23DEA8003/H323/cch323_generate_srtp_info:
../voip/cch323/gw/os/src/h323_gw_srtpapi.c:cch323_generate_srtp_info:31: Entry
000089: Mar 16 16:46:38.245: //4/CCB23DEA8003/H323/cch323_generate_srtp_info: Generated
SRTP info:../voip/cch323/gw/os/src/h323_gw_srtpapi.c:cch323_generate_srtp_info:83
000090: Mar 16 16:46:38.245: //4/CCB23DEA8003/H323/cch323_dump_srtp_caps:
crypto_bm=0x1,
codec_bm=0xB.
000091: Mar 16 16:46:38.245: //4/CCB23DEA8003/H323/cch323_dump_srtp_media_params:
ssrc_sel=0x0,
srtp_ssrc=0x0,
crypto_services=0x3,
crypto_suite=0x1,
master_key_len=0x10,
master_salt_len=0xE,
master_key=0xxxxxxxxxxxxxxxx,
master_salt=0xxxxxxxxxxxxxx
000092: Mar 16 16:46:38.245: //4/CCB23DEA8003/H323/cch323_dump_srtp_sess_params:
lifetime=0x0,
mki=0x,
mki_length=0x0,
kdr=0x18,
encryptedsrtp=0x1,
encryptedsrtcp=0x1,
fecorder=0x0,
windowsizehint=0x40
000093: Mar 16 16:46:38.245: //4/CCB23DEA8003/H323/cch323_generate_srtp_info:
../voip/cch323/gw/os/src/h323_gw_srtpapi.c:cch323_generate_srtp_info:86: Exit
000094: Mar 16 16:46:38.249: //4/CCB23DEA8003/H323/cch323_generate_srtp_info:
../voip/cch323/gw/os/src/h323_gw_srtpapi.c:cch323_generate_srtp_info:31: Entry
000095: Mar 16 16:46:38.249: //4/CCB23DEA8003/H323/cch323_generate_srtp_info: Generated
SRTP info:../voip/cch323/gw/os/src/h323_gw_srtpapi.c:cch323_generate_srtp_info:83
000096: Mar 16 16:46:38.249: //4/CCB23DEA8003/H323/cch323_dump_srtp_caps:
crypto_bm=0x0,
codec_bm=0x0.
000097: Mar 16 16:46:38.249: //4/CCB23DEA8003/H323/cch323_dump_srtp_media_params:
ssrc_sel=0x0,
srtp_ssrc=0x0,
crypto_services=0x3,
crypto_suite=0x0,
master_key_len=0x10,
master_salt_len=0xE,
master_key=0xxxxxxxxxxxxxxxx,
master_salt=0xxxxxxxxxxxxxx
000098: Mar 16 16:46:38.249: //4/CCB23DEA8003/H323/cch323_dump_srtp_sess_params:
lifetime=0x0,
mki=0x,
mki_length=0x0,
kdr=0x18,
encryptedsrtp=0x1,
encryptedsrtcp=0x1,
fecorder=0x0,
windowsizehint=0x40
000099: Mar 16 16:46:38.249: //4/CCB23DEA8003/H323/cch323_generate_srtp_info:
../voip/cch323/gw/os/src/h323_gw_srtpapi.c:cch323_generate_srtp_info:86: Exit
000100: Mar 16 16:46:38.249: //4/CCB23DEA8003/H323/build_fastStart_OLCs: FWD OLC SRTP
params:../voip/cch323/gw/src/cch323_h245_util.c:build_fastStart_OLCs:1403
000101: Mar 16 16:46:38.249: //4/CCB23DEA8003/H323/build_fastStart_OLCs: RVR OLC SRTP
Params:../voip/cch323/gw/src/cch323_h245_util.c:build_fastStart_OLCs:1417
000102: Mar 16 16:46:38.301: //4/CCB23DEA8003/H323/cch323_build_olc_for_ccapi:
../voip/cch323/gw/src/cch323_h245_util.c:cch323_build_olc_for_ccapi:1690: WE ARE OGW.
```

The following lines show the outgoing gateway sending SRTP capabilities:

```
000103: Mar 16 16:46:38.301: //4/CCB23DEA8003/H323/cch323_build_olc_for_ccapi: Local SRTP
Info:../voip/cch323/gw/src/cch323_h245_util.c:cch323_build_olc_for_ccapi:1779
000104: Mar 16 16:46:38.301: //4/CCB23DEA8003/H323/cch323_dump_srtp_caps:
crypto_bm=0x1,
codec_bm=0xB.
000105: Mar 16 16:46:38.301: //4/CCB23DEA8003/H323/cch323_dump_srtp_media_params:
ssrc_sel=0x0,
srtp_ssrc=0x0,
crypto_services=0x3,
crypto_suite=0x1,
master_key_len=0x10,
master_salt_len=0xE,
master_key=0xxxxxxxxxxxxxxxx,
master_salt=0xxxxxxxxxxxxxx
000106: Mar 16 16:46:38.301: //4/CCB23DEA8003/H323/cch323_dump_srtp_sess_params:
lifetime=0x0,
mki=0x,
mki_length=0x0,
kdr=0x18,
encryptedsrtp=0x1,
encryptedsrtcp=0x1,
fecorder=0x0,
windowsizehint=0x40
```

The following lines show that the gateway has received SRTP capabilities from the remote end:

```
000107: Mar 16 16:46:38.301: //4/CCB23DEA8003/H323/cch323_build_olc_for_ccapi: Remote SRTP
 Info:../voip/cch323/gw/src/cch323_h245_util.c:cch323_build_olc_for_ccapi:1783
000108: Mar 16 16:46:38.301: //4/CCB23DEA8003/H323/cch323_dump_srtp_caps:
crypto_bm=0x1,
codec_bm=0x1.
000109: Mar 16 16:46:38.301: //4/CCB23DEA8003/H323/cch323_dump_srtp_media_params:
ssrc_sel=0x0,
srtp_ssrc=0x0,
crypto_services=0x3,
crypto_suite=0x1,
master_key_len=0x10,
master_salt_len=0xE,
master_key=0xxxxxxxxxxxxxxxx,
master_salt=0xxxxxxxxxxxxxx
000110: Mar 16 16:46:38.301: //4/CCB23DEA8003/H323/cch323_dump_srtp_sess_params:
lifetime=0x0,
mki=0x,
mki_length=0x0,
kdr=0x18,
encryptedsrtp=0x1,
encryptedsrtcp=0x1,
fecorder=0x0,
windowsizehint=0x40
000111: Mar 16 16:46:38.305: //4/CCB23DEA8003/H323/h323_common_setup_rtcp_parameters:
../voip/cch323/gw/os/src/h323_gw_rtpapi.c:h323_common_setup_rtcp_parameters:378:olc->rtcp_session.srtp_services:3
000112: Mar 16 16:46:38.305: vtsp_call_ssrc_update: updated ssrc=0x5227F02
000113: Mar 16 16:46:38.305: //4/CCB23DEA8003/H323/cch323_setup_srtp_session:
../voip/cch323/gw/os/src/h323_gw_srtpapi.c:cch323_setup_srtp_session:369: Entry
000114: Mar 16 16:46:38.305: //4/CCB23DEA8003/H323/cch323_setup_srtp_session:
../voip/cch323/gw/os/src/h323_gw_srtpapi.c:cch323_setup_srtp_session:416: TX [Local] SRTP
Info .
000115: Mar 16 16:46:38.305: //4/CCB23DEA8003/H323/cch323_dump_srtp_media_params:
ssrc_sel=0x0,
srtp_ssrc=0x0,
crypto_services=0x3,
crypto_suite=0x1,
master_key_len=0x10,
master_salt_len=0xE,
master_key=0xxxxxxxxxxxxxxxx,
master_salt=0xxxxxxxxxxxxxx
000116: Mar 16 16:46:38.305: //4/CCB23DEA8003/H323/cch323_setup_srtp_session:
../voip/cch323/gw/os/src/h323_gw_srtpapi.c:cch323_setup_srtp_session:448: RX [Remote] SRTP
 Info .
000117: Mar 16 16:46:38.305: //4/CCB23DEA8003/H323/cch323_dump_srtp_media_params:
```

```
ssrc_sel=0x0,
srtp_ssrc=0x0,
crypto_services=0x3,
crypto_suite=0x1,
master_key_len=0x10,
master_salt_len=0xE,
master_key=0xxxxxxxxxxxxxxxx,
master_salt=0xxxxxxxxxxxxxx
000118: Mar 16 16:46:38.305: //4/CCB23DEA8003/H323/cch323_setup_srtp_session:
% SRTP Library session creation, id:0x80000002, context:0x653E0CF8, num_context=2,
rtp:0x653D2CF4, rtp_session:0x66D79B00
000119: Mar 16 16:46:38.305: //4/CCB23DEA8003/H323/cch323_setup_srtp_session:
../voip/cch323/gw/os/src/h323_gw_srtpapi.c:cch323_setup_srtp_session:514: Exit
000120: Mar 16 16:46:38.313: //4/CCB23DEA8003/H323/h323_common_setup_rtcp_parameters:
../voip/cch323/gw/os/src/h323_gw_rtpapi.c:h323_common_setup_rtcp_parameters:378:olc->rtcp_session.srtp_services:3
000121: Mar 16 16:46:38.313: //4/CCB23DEA8003/H323/cch323_setup_srtp_session:
../voip/cch323/gw/os/src/h323_gw_srtpapi.c:cch323_setup_srtp_session:369: Entry
000122: Mar 16 16:46:38.313: //4/CCB23DEA8003/H323/cch323_setup_srtp_session:
../voip/cch323/gw/os/src/h323_gw_srtpapi.c:cch323_setup_srtp_session:416: TX [Local] SRTP
Info .
000123: Mar 16 16:46:38.313: //4/CCB23DEA8003/H323/cch323_dump_srtp_media_params:
ssrc_sel=0x0,
srtp_ssrc=0x0,
crypto_services=0x3,
crypto_suite=0x1,
master_key_len=0x10,
master_salt_len=0xE,
master_key=0xxxxxxxxxxxxxxxx,
master_salt=0xxxxxxxxxxxxxx
000124: Mar 16 16:46:38.313: //4/CCB23DEA8003/H323/cch323_setup_srtp_session:
../voip/cch323/gw/os/src/h323_gw_srtpapi.c:cch323_setup_srtp_session:448: RX [Remote] SRTP
 Info .
000125: Mar 16 16:46:38.313: //4/CCB23DEA8003/H323/cch323_dump_srtp_media_params:
ssrc_sel=0x0,
srtp_ssrc=0x0,
crypto_services=0x3,
crypto_suite=0x1,
master_key_len=0x10,
master_salt_len=0xE,
master_key=0xxxxxxxxxxxxxxxx,
master_salt=0xxxxxxxxxxxxxx
000126: Mar 16 16:46:38.313: //4/CCB23DEA8003/H323/cch323_setup_srtp_session:
% SRTP Library session update with 2 keysid:0x80000002 context:0x653E0CF8
000127: Mar 16 16:46:38.313: //4/CCB23DEA8003/H323/cch323_setup_srtp_session:
../voip/cch323/gw/os/src/h323_gw_srtpapi.c:cch323_setup_srtp_session:514: Exit
000128: Mar 16 16:46:38.317: //4/CCB23DEA8003/H323/h323_common_setup_rtcp_parameters:
../voip/cch323/gw/os/src/h323_gw_rtpapi.c:h323_common_setup_rtcp_parameters:378:olc->rtcp_session.srtp_services:3
000129: Mar 16 16:46:38.317: //4/CCB23DEA8003/H323/cch323_setup_srtp_session:
../voip/cch323/gw/os/src/h323_gw_srtpapi.c:cch323_setup_srtp_session:369: Entry
000130: Mar 16 16:46:38.317: //4/CCB23DEA8003/H323/cch323_setup_srtp_session:
../voip/cch323/gw/os/src/h323_gw_srtpapi.c:cch323_setup_srtp_session:416: TX [Local] SRTP
Info .
000131: Mar 16 16:46:38.317: //4/CCB23DEA8003/H323/cch323_dump_srtp_media_params:
ssrc_sel=0x0,
srtp_ssrc=0x0,
crypto_services=0x3,
crypto_suite=0x1,
master_key_len=0x10,
master_salt_len=0xE,
master_key=0xxxxxxxxxxxxxxxx,
master_salt=0xxxxxxxxxxxxxx
000132: Mar 16 16:46:38.317: //4/CCB23DEA8003/H323/cch323_setup_srtp_session:
../voip/cch323/gw/os/src/h323_gw_srtpapi.c:cch323_setup_srtp_session:448: RX [Remote] SRTP
 Info .
000133: Mar 16 16:46:38.317: //4/CCB23DEA8003/H323/cch323_dump_srtp_media_params:
ssrc_sel=0x0,
srtp_ssrc=0x0,
crypto_services=0x3,
crypto_suite=0x1,
master_key_len=0x10,
master_salt_len=0xE,
master_key=0xxxxxxxxxxxxxxxx,
master_salt=0xxxxxxxxxxxxxx
```

```
000134: Mar 16 16:46:38.317: //4/CCB23DEA8003/H323/cch323_setup_srtp_session:
% SRTP Library session update with 2 keysid:0x80000002 context:0x653E0CF8
000135: Mar 16 16:46:38.321: //4/CCB23DEA8003/H323/cch323_setup_srtp_session:
../voip/cch323/gw/os/src/h323_gw_srtpapi.c:cch323_setup_srtp_session:514: Exit
000136: Mar 16 16:46:38.321: //4/CCB23DEA8003/H323/cch323_h245_cap_ind: Updated
CCB(0x66D8D2D4) with TCS Remote SRTP Info:
000137: Mar 16 16:46:38.321: //4/CCB23DEA8003/H323/cch323_dump_srtp_caps:
crypto_bm=0x1,
codec_bm=0xB.
000138: Mar 16 16:46:38.321: //4/CCB23DEA8003/H323/cch323_dump_srtp_media_params:
ssrc_sel=0x0,
srtp_ssrc=0x0,
crypto_services=0x3,
crypto_suite=0x1,
master_key_len=0x10,
master_salt_len=0xE,
master_key=0xxxxxxxxxxxxxxxx,
master_salt=0xxxxxxxxxxxxxx
000139: Mar 16 16:46:38.321: //4/CCB23DEA8003/H323/cch323_dump_srtp_sess_params:
lifetime=0x0,
mki=0x,
mki_length=0x0,
kdr=0x18,
encryptedsrtp=0x1,
encryptedsrtcp=0x1,
fecorder=0x0,
windowsizehint=0x40
000140: Mar 16 16:46:38.321: //4/CCB23DEA8003/H323/cch323_update_tcs_nonstd_info:
cch323_update_tcs_nonstd_info:5800: Posting TCS SRTP caps to other callleg.
000141: Mar 16 16:46:38.321: //4/CCB23DEA8003/H323/cch323_dump_srtp_caps:
crypto_bm=0x1,
codec_bm=0xB.
000142: Mar 16 16:46:38.321: //4/CCB23DEA8003/H323/cch323_dump_srtp_media_params:
ssrc_sel=0x0,
srtp_ssrc=0x0,
crypto_services=0x3,
crypto_suite=0x1,
master_key_len=0x10,
master_salt_len=0xE,
master_key=0xxxxxxxxxxxxxxxx,
master_salt=0xxxxxxxxxxxxxx
000143: Mar 16 16:46:38.321: //4/CCB23DEA8003/H323/cch323_dump_srtp_sess_params:
lifetime=0x0,
mki=0x,
mki_length=0x0,
kdr=0x18,
encryptedsrtp=0x1,
encryptedsrtcp=0x1,
fecorder=0x0,
windowsizehint=0x40
000144: Mar 16 16:46:38.325: //4/CCB23DEA8003/H323/cch323_h245_cap_ind:
cch323_h245_cap_ind:360 cch323_update_tcs_nonstd_info failed
```

Related Commands

Command	Description
debug h225 asn1	Displays ASN.1 contents of H.225 messages.
debug h245 asn1	Displays ASN.1 contents of H.245 messages.
logging buffered	Enables system message logging to a local buffer.
srtp (voice)	Enables secure calls globally.
srtp (dial-peer)	Enables secure calls for a specific dial peer.

debug h323-annexg

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

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

no debug h323-annexg

Syntax Description

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

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

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

Examples

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

```
Router# debug h323-annexg events
Aug 16 14:03:40.983:be_process:BE QUEUE_EVENT (minor 73) wakeup
Aug 16 14:03:40.983:be_sm:Received event BE_EV_DO_QUERY
Aug 16 14:03:40.983:<- query_neighbor:Sent descriptorIDRequest to
172.18.195.46:2099 [320]
Aug 16 14:03:40.983:be_sm:Started query-timer of 1 minutes for
neighbor at 172.18.195.46
Aug 16 14:03:40.991:-> nxg_recv_msg:Rcvd dscrptrIDCnfrmtn from
172.18.195.46:2099 [320]
Aug 16 14:03:41.531:<- send_descriptor_request:Sent descriptorRequest
```

```
to 172.18.195.46:2099 [321]
Aug 16 14:03:41.539:-> nxg_recv_msg:Rcvd descriptorConfirmation from
172.18.195.46:2099 [321]
Aug 16 14:03:41.539:handle_descriptor_cfm:Descriptor from neighbor
172.18.195.46 unchanged, TTL is 60 Seconds
```

Related Commands

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

debug hccp timing

To display debug messages for the timing of HCCP events, use the **debug hccp timing**command in privileged EXEC mode. To disable the debug message output, use the **no**form of this command.

debug hccp timing [if-config]

no debug hccp timing [if-config]

Syntax Description

if-config	(Optional) Displays debugging messages showing the timing of the reconfiguration of cable interfaces during HCCP redundancy operations.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.1(3a)EC	This command was introduced.
12.2(4)XF1, 12.2(4)BC1	Support was added for the N+1 (1:n) RF Switch with the Cisco uBR10012 router.
12.2(11)BC1	Support was added for the N+1 (1:n) RF Switch with the Cisco uBR7246VXR router.
12.2(15)BC1	The **if-config** option was added.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.

Usage Guidelines

You must activate the **debug hccp events** command before the **debug hccp timing**command will generate any debug message output.

Examples

The following example shows typical output for the **debug** hccp timing command:

```
Router# debug hccp events
Router# debug hccp timing
HCCP timing measurement debugging is on
May 31 10:21:07.609 HCCP P is busy. Deactivating 1 6
May 31 10:21:07.609 HCCP P is busy. Deactivating 2 6
May 31 10:21:08.705 HCCP hwif_goingdown for Cable8/1/0. Deactivate 1 6
May 31 10:21:08.705 HCCP hwif_goingdown for Cable8/1/1. Deactivate 2 6
May 31 10:21:08.773 HCCP 2 6 Working: become standby - 68 msec
May 31 10:21:08.793 HCCP 1 6 Working: become standby - 20 msec
```

```
May 31 10:21:10.730 HCCP 1 1 Working: turn on "uc" - 8 msec
May 31 10:21:10.730 HCCP 1 1 Working: turn on "nru" - 0 msec
May 31 10:21:10.734 HCCP 1 1 Working: become active - 4 msec
May 31 10:21:10.774 HCCP 2 1 Working: turn on "uc" - 52 msec
May 31 10:21:10.774 HCCP 2 1 Working: turn on "nru" - 0 msec
May 31 10:21:10.774 HCCP 2 1 Working: become active - 0 msec
May 31 10:21:12.350 HCCP hwif_goingdown for Cable5/1/0. Deactivate 1 1
May 31 10:21:12.350 HCCP hwif_goingdown for Cable5/1/0. Deactivate 1 6
May 31 10:21:12.350 HCCP hwif_goingdown for Cable5/1/0. Deactivate 1 3
May 31 10:21:12.350 HCCP hwif_goingdown for Cable5/1/0. Deactivate 1 2
May 31 10:21:12.350 HCCP hwif_goingdown for Cable5/1/0. Deactivate 1 5
May 31 10:21:12.350 HCCP hwif_goingdown for Cable5/1/0. Deactivate 1 4
May 31 10:21:12.350 HCCP hwif_goingdown for Cable5/1/1. Deactivate 2 1
May 31 10:21:12.350 HCCP hwif_goingdown for Cable5/1/1. Deactivate 2 3
May 31 10:21:12.350 HCCP hwif_goingdown for Cable5/1/1. Deactivate 2 6
May 31 10:21:12.350 HCCP hwif_goingdown for Cable5/1/1. Deactivate 2 2
May 31 10:21:12.350 HCCP hwif_goingdown for Cable5/1/1. Deactivate 2 4
May 31 10:21:12.350 HCCP hwif_goingdown for Cable5/1/1. Deactivate 2 5
May 31 10:21:13.726 HCCP 1 1 Protect: turn off "uc" - 1972 msec
May 31 10:21:13.790 HCCP 2 1 Protect: turn off "uc" - 2036 msec
May 31 10:21:14.422 HCCP 1 1 Protect: turn off "nru" - 696 msec
May 31 10:21:14.422 HCCP 1 1 Protect: unload config (if) - 0 msec
May 31 10:21:14.438 HCCP 1 1 Protect: unload config (subif) - 16 msec
May 31 10:21:14.702 HCCP 1 1 Protect: unload config (ds) - 264 msec
May 31 10:21:14.702 HCCP 1 1 Protect: become standby - 0 msec
May 31 10:21:16.078 HCCP 2 1 Protect: turn off "nru" - 2288 msec
May 31 10:21:16.078 HCCP 2 1 Protect: unload config (if) - 0 msec
May 31 10:21:16.078 HCCP 2 1 Protect: unload config (subif) - 0 msec
May 31 10:21:16.599 HCCP 2 1 Protect: unload config (ds) - 520 msec
May 31 10:21:16.599 HCCP 2 1 Protect: become standby - 0 msec
May 31 10:21:17.014 HCCP: P missed hello ack in LEARN state and is locked. Deactivate 4 1
May 31 10:21:17.014 HCCP 4 1 Protect: turn off "rfswitch" - 52 msec
May 31 10:21:17.593 HCCP 3 1 Working: turn on "rfswitch" - 0 msec
May 31 10:21:17.593 HCCP 3 1 Working: become active - 0 msec
May 31 10:21:18.112 HCCP 1 1 Protect: load config (if) - 0 msec
May 31 10:21:18.112 HCCP 1 1 Protect: load config (subif) - 4 msec
May 31 10:21:18.331 HCCP 1 1 Protect: load config (ds) - 100 msec
May 31 10:21:18.331 HCCP 2 1 Working: turn off "rfswitch" - 0 msec
May 31 10:21:18.331 HCCP 2 Cable5/0/1 Protect: resolve conflict Learn->Teach
May 31 10:21:18.331 HCCP 2 1 Protect: load config (if) - 0 msec
May 31 10:21:18.331 HCCP 2 1 Protect: load config (subif) - 0 msec
May 31 10:21:19.691 HCCP 2 1 Protect: load config (ds) - 76 msec
May 31 10:21:20.112 HCCP 2 1 Protect: turn on "rfswitch" - 48 msec
May 31 10:21:20.112 HCCP 2 1 Protect: become active - 0 msec
May 31 10:21:20.112 HCCP 2 1 Protect: load config (ds) - 76 msec
May 31 10:21:20.112 HCCP 2 1 Protect: turn on "rfswitch" - 48 msec
May 31 10:21:20.112 HCCP 2 1 Protect: become active - 0 msec
```

The following example shows typical output for the **debug** hccp timing if-config command:

```
Router# debug hccp events
Router# debug hccp timing if-config
HCCP Timing measurements messages of (UN)LOAD IF config CLI is on
HCCP 1 1 Working: unload config (ds) - 112 msec
HCCP 1 1 Protect: load config (ds) - 123 msec
HCCP 1 1 Protect: load config (chnl set freq) - 35 msec
```

Related Commands

Command	Description
debug hccp authentication	Displays authentication debug messages for HCCP groups.
debug hccp channel-switch	Displays debug messages related to an RF or channel switch that is being used for HCCP N+1 (1:n) redundancy.

Command	Description
debug hccp events	Displays debug messages for all HCCP group interaction.
debug hccp inter-db	Displays debug messages for the inter-database events during HCCP operations.
debug hccp plane	Displays debug messages for HCCP-related messages sent between the router's control plane and data backplane.
debug hccp sync	Displays debug messages for HCCP synchronization messages.

debug hpi

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

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

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

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

Syntax Description

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

Command Default This command is not enabled.

Command Modes Privileged EXEC

Command History

Release	Modification
12.1(5)XM	This command was introduced on the Cisco AS5300 and Cisco AS5800.
12.2(2)T	This command was implemented on the Cisco 1700, Cisco 2600 series, Cisco 3600 series, and the Cisco MC3810. The **stats** keyword was added.
12.2(10), 12.2(11)T	This command was implemented on the Cisco 827, Cisco 2400, Cisco 7200 series, and Cisco CVA 120. The following keywords were added: **buffer**, **capture**, **and destination**.
12.3(8)T	This command was replaced by the **debug voip hpi** command.

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

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

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

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

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

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

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

Examples

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

```
Router# debug hpi capture
HPI Capture/Logger debugging is on
```

Related Commands

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

debug http client

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

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

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

Syntax Description

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

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

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

Usage Guidelines

The output of this command is effected by the **debug condition application voice** command. If the **debug condition application voice** command is configured and the <cisco-debug> element is enabled in the

VoiceXML document, debugging output is limited to the VoiceXML application named in the **debug condition application voice** command.

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

Examples

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

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

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

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

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

```
Router# debug http client main
HTTP Client Main Process debugging is on
*Jan  3 11:56:05.885: httpc_get, url: http://serverX.com/vxml/test/root.vxml
*Jan  3 11:56:05.889: httpc_msg_send, sid: 0, method: 83951618
*Jan  3 11:56:05.889: httpc_enqueue_wmsg, sid: 0, method: 83951618
*Jan  3 11:56:05.893: httpc_process_write_queue, socket writeble fd: 0, process enqueued
msg, sid: 0, method: 83951618
*Jan  3 11:56:05.893: httpc_msg_write, sid: 0, method: 83951618
*Jan  3 11:56:05.901: HTTPC_MSG_COMPLETE:
```

```
rsp_code(304),msg(62C9C25C)URL:http://serverX.com/vxml/test/root.vxml, fd(0)
*Jan  3 11:56:05.901: httpc_process_redirect_rsp:
msg(62C9C25C)URL:http://serverX.com/vxml/test/root.vxml, response code
HTTPC_NOT_MODIFIED_304
*Jan  3 11:56:05.913: httpc_get, url:
http://serverX.com/vxml/test/getPhoneInfo.vxml?ani=1234567&dnis=7654321
*Jan  3 11:56:05.917: httpc_msg_send, sid: 0, method: 65538
*Jan  3 11:56:05.917: httpc_enqueue_wmsg, sid: 0, method: 65538
*Jan  3 11:56:05.917: httpc_process_write_queue, socket writeble fd: 1, process enqueued
msg,
                          sid: 0, method: 65538
*Jan  3 11:56:05.917: httpc_msg_write, sid: 0, method: 65538
*Jan  3 11:56:05.925: HTTPC_MSG_COMPLETE:
rsp_code(200),msg(62CB5824)URL:http://serverX.com/vxml/test/getPhoneInfo.vxml?ani=1234567&
dnis=7654321, fd(1)
*Jan  3 11:56:05.929: httpc_get, url: http://1.7.100.1/vxml/test/engine.vxml?flow=iso
*Jan  3 11:56:05.929: httpc_msg_send, sid: 0, method: 65538
*Jan  3 11:56:05.929: httpc_enqueue_wmsg, sid: 0, method: 65538
*Jan  3 11:56:05.929: httpc_process_free_rsp: User returns noncache response (626F9670)
*Jan  3 11:56:05.929: httpc_process_write_queue, socket writeble fd: 1, process enqueued
msg,
                          sid: 0, method: 65538
*Jan  3 11:56:05.929: httpc_msg_write, sid: 0, method: 65538
*Jan  3 11:56:05.937: HTTPC_MSG_COMPLETE:
rsp_code(200),msg(62CB03AC)URL:http://1.7.100.1/vxml/test/engine.vxml?flow=iso, fd(1)
```

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

```
Router# debug http client msg
HTTP Client:
  HTTP Client Messages debugging is on
*Jan  1 05:07:30.534: HTTP Client write buffer fd(0):
GET /vxml/abcdefg/test/init.vxml HTTP/1.1
Host: c5300-2
Content-Type: application/x-www-form-urlencoded
Connection: Keep-Alive
Accept: text/vxml; level = 1, text/plain, text/html, audio/basic
User-Agent: Cisco-IOS-C5300/12.2(20010829:180555) VoiceXML/1.0
*Jan  1 05:07:30.538: about to send data to socket 0 :
first 263 bytes of data:
62397130:                           47455420             GET
62397140: 2F76786D 6C2F6162 63646566 672F7465  /vxml/abcdefg/te
62397150: 73742F69 6E69742E 76786D6C 20485454  st/init.vxml HTT
62397160: 502F312E 310D0A48 6F73743A 20633533  P/1.1..Host: c53
62397170: 30302D32 0D0A436F 6E74656E 742D5479  00-2..Content-Ty
62397180: 70653A20 6170706C 69636174 696F6E2F  pe: application/
62397190: 782D7777 772D666F 726D2D75 726C656E  x-www-form-urlen
623971A0: 636F6465 640D0A43 6F6E6E65 6374696F  coded..Connectio
623971B0: 6E3A204B 6565702D 416C6976 650D0A41  n: Keep-Alive..A
623971C0: 63636570 743A2074 6578742F 76786D6C  ccept: text/vxml
623971D0: 3B206C65 76656C20 3D20312C 20746578  ; level = 1, tex
623971E0: 742F706C 61696E2C 20746578 742F6874  t/plain, text/ht
623971F0: 6D6C2C20 61756469 6F2F6261 7369630D  ml, audio/basic.
62397200: 0A557365 722D4167 656E743A 20436973  .User-Agent: Cis
62397210: 636F2D49 4F532D43 35333030 2F31322E  co-IOS-C5300/12.
62397220: 32283230 30313038 32393A31 38303535  2(20010829:18055
62397230: 35292056 6F696365 584D4C2F 312E300D  5) VoiceXML/1.0.
62397240: 0A0D0A00                             ....
*Jan  1 05:07:30.546: read data from socket 0 :
first 400 bytes of data:
628DE8F0:           48545450 2F312E31 20323030      HTTP/1.1 200
628DE900: 204F4B0D 0A446174 653A2046 72692C20   OK..Date: Fri,
628DE910: 33312041 75672032 30303120 30373A30  31 Aug 2001 07:0
628DE920: 363A3335 20474D54 0D0A5365 72766572  6:35 GMT..Server
628DE930: 3A204170 61636865 2F312E33 2E313120  : Apache/1.3.11
628DE940: 28556E69 78292041 70616368 654A5365  (Unix) ApacheJSe
628DE950: 72762F31 2E300D0A 4C617374 2D4D6F64  rv/1.0..Last-Mod
628DE960: 69666965 643A2057 65642C20 3233204D  ified: Wed, 23 M
628DE970: 61792032 30303120 31353A35 333A3233  ay 2001 15:53:23
628DE980: 20474D54 0D0A4554 61673A20 22323430   GMT..ETag: "240
628DE990: 37642D31 39322D33 62306264 63663322  7d-192-3b0bdcf3"
628DE9A0: 0D0A4163 63657074 2D52616E 6765733A  ..Accept-Ranges:
628DE9B0: 20627974 65730D0A 436F6E74 656E742D   bytes..Content-
```

```
628DE9C0: 4C656E67 74683A20 3430320D 0A4B6565  Length: 402..Kee
628DE9D0: 702D416C 6976653A 2074696D 656F7574  p-Alive: timeout
628DE9E0: 3D352C20 6D61783D 31300D0A 436F6E6E  =5, max=10..Conn
628DE9F0: 65637469 6F6E3A20 4B656570 2D416C69  ection: Keep-Ali
628DEA00: 76650D0A 436F6E74 656E742D 54797065  ve..Content-Type
628DEA10: 3A207465 78742F76 786D6C0D 0A0D0A3C  : text/vxml....<
628DEA20: 3F786D6C 20766572 73696F6E 3D22312E  ?xml version="1.
628DEA30: 30223F3E 0A3C7678 6D6C2076 65727369  0"?>.<vxml versi
628DEA40: 6F6E3D22 312E3022 20617070 6C696361  on="1.0" applica
628DEA50: 74696F6E 3D22726F 6F742E76 786D6C22  tion="root.vxml"
628DEA60: 3E0A2020 3C666F72 6D3E0A20 2020203C  >.  <form>.    <
628DEA70: 626C6F63 6B3E0A20 20202020 203C212D  block>.      <!-
628DEA80: 2D0A2020 20                          -.
*Jan  1 05:07:30.550: httpc_decode_msgheader: Client ignores this header: Server:
Apache/1.3.11 (Unix) ApacheJServ/1.0
*Jan  1 05:07:30.554: httpc_decode_msgheader: Client ignores this header: Accept-Ranges:
bytes
*Jan  1 05:07:30.554: processing server rsp msg:
msg(62C711C4)URL:http://vvv.com/vxml/abcdefg/test/init.vxml, fd(0):
*Jan  1 05:07:30.554: Request msg: GET /vxml/abcdefg/test/init.vxml HTTP/1.1
*Jan  1 05:07:30.554: Message Response Code: 200
*Jan  1 05:07:30.554: Message Rsp Decoded Headers:
*Jan  1 05:07:30.554: Date:Fri, 31 Aug 2001 07:06:35 GMT
*Jan  1 05:07:30.554: Content-Length:402
*Jan  1 05:07:30.554: Content-Type:text/vxml
*Jan  1 05:07:30.554: ETag:"2407d-192-3b0bdcf3"
*Jan  1 05:07:30.554: Last-Modified:Wed, 23 May 2001 15:53:23 GMT
*Jan  1 05:07:30.554: Connection:Keep-Alive
*Jan  1 05:07:30.554: Keep-Alive:timeout=5, max=10
*Jan  1 05:07:30.554: httpc_dump_msg: headers:
*Jan  1 05:07:30.554: HTTP/1.1 200 OK
Date: Fri, 31 Aug 2001 07:06:35 GMT
Server: Apache/1.3.11 (Unix) ApacheJServ/1.0
Last-Modified: Wed, 23 May 2001 15:53:23 GMT
ETag: "2407d-192-3b0bdcf3"
Accept-Ranges: bytes
Content-Length: 402
Keep-Alive: timeout=5, max=10
Connection: Keep-Alive
Content-Type: text/vxml
*Jan  1 05:07:30.558: httpc_dump_msg: body:
*Jan  1 05:07:30.558: <?xml version="1.0"?>
<vxml version="1.0" application="root.vxml">
  <form>
    <block>
      <!--
      <var name="ani" expr="session.telephone.ani"/>
      <var name="dnis" expr="session.telephone.dnis"/>
      -->
      <var name="ani" expr="1234567"/>
      <var name="dnis" expr="7654321"/>
      <submit next="getPhoneInfo.vxml" method="get" namelist="ani dnis"/>
    </block>
  </form>
</vxml>
```

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

```
Router# debug http client socket
HTTP Client Sockets debugging is on
*Jan  3 11:32:38.353: httpc_process_read_ev: HTTPC SOCK_PENDING --> SOCK_CONNECTED fd(0)
port(80)
*Jan  3 11:32:38.377: httpc_process_read_ev: HTTPC SOCK_PENDING --> SOCK_CONNECTED fd(1)
port(80)
*Jan  3 11:32:38.381: httpc_socket_cleanup: fd(1)
*Jan  3 11:32:38.389: httpc_process_read_ev: HTTPC SOCK_PENDING --> SOCK_CONNECTED fd(1)
port(80)
*Jan  3 11:32:38.393: httpc_socket_cleanup: fd(1)
*Jan  3 11:32:38.397: httpc_process_read_ev: HTTPC SOCK_PENDING --> SOCK_CONNECTED fd(1)
port(80)
*Jan  3 11:32:40.361: httpc_socket_cleanup: fd(0)
*Jan  3 11:32:40.413: httpc_socket_cleanup: fd(1)
*Jan  3 11:40:43.557: httpc_socket_connect failed fd(2)
```

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

```
Router# debug http client error
HTTP Client Error debugging is on

*Jan  3 12:07:40.209: HTTPC URL:http://serverX.com/vxml/test.vxml, Server rsp_code(404),
fd(0)
*Jan  3 12:08:01.677: HTTPC SOCK_FAIL() - msg(62CA5FB4)URL:http://serverX/vxml/test.vxml
*Jan  3 12:08:01.677: httpc_free: NULL pointer argument
*Jan  3 12:08:01.677: HTTPC URL:http://serverX/vxml/test.vxml, MSG_XMIT_ERROR, fd(-1)
Jan   3 23:44:06 PDT: HTTPC
URL:http://serverY.com:9000/ivr/sid-351/dropoffDeposit?pri=0&disconnect=1, TIMEOUT(60000
msec), fd(-1)
Jan   3 23:44:07 PDT: HTTPC msg timeout waiting for rsp - fd(21)
Jan   3 23:44:07 PDT: httpc_free: NULL pointer argument
Jan   4 02:45:07 PDT: HTTPC msg timeout waiting for rsp - fd(0)
Jan   4 02:45:07 PDT: HTTPC URL:http://rtsp-ws/dropoffAppend.php?append=&disconnect=1,
TIMEOUT(10000 msec), fd(-1)
Jan   4 02:46:07 PDT: httpc_msg_read: URL(http://1.7.100.1/vxml/root.vxml) - msg length not
 available.Failed to parse message body.
Jan   4 02:46:07 PDT: httpc_msg_read: ERROR - DECODE
Jan   4 02:46:08 PDT: HTTPC bad message read - fd(6), conp(632A93B4),
msg(63280794)URL:http://1.7.100.1/vxml/test/root.vxml, len(1611)
Jan   4 02:46:08 PDT: First 400 bytes read from socket:
6241D9C0:                             3C3F78              <?x
6241D9D0: 6D6C2076 65727369 6F6E3D22 312E3022  ml version="1.0"
6241D9E0: 3F3E0A3C 76786D6C 20766572 73696F6E  ?>.<vxml version
6241D9F0: 3D22312E 30223E0A 0A20203C 70726F70  ="1.0">..  <prop
6241DA00: 65727479 206E616D 653D2263 61636869  erty name="cachi
6241DA10: 6E672220 76616C75 653D2273 61666522  ng" value="safe"
6241DA20: 2F3E0A20 203C7072 6F706572 7479206E  />.  <property n
6241DA30: 616D653D 2274696D 656F7574 22207661  ame="timeout" va
6241DA40: 6C75653D 22333073 222F3E0A 20203C70  lue="30s"/>.  <p
6241DA50: 726F7065 72747920 6E616D65 3D226665  roperty name="fe
6241DA60: 74636874 696D656F 75742220 76616C75  tchtimeout" valu
6241DA70: 653D2236 3073222F 3E0A2020 3C70726F  e="60s"/>.  <pro
6241DA80: 70657274 79206E61 6D653D22 696E7075  perty name="inpu
6241DA90: 746D6F64 65732220 76616C75 653D2264  tmodes" value="d
6241DAA0: 746D6622 2F3E0A0A 20203C76 6172206E  tmf"/>..  <var n
6241DAB0: 616D653D 22736964 22206578 70723D22  ame="sid" expr="
6241DAC0: 27313131 31312722 2F3E0A20 203C7661  '11111'"/>.  <va
6241DAD0: 72206E61 6D653D22 6C632220 65787072  r name="lc" expr
6241DAE0: 3D222765 6E2D7573 27222F3E 0A20203C  ="'en-us'"/>.  <
6241DAF0: 76617220 6E616D65 3D226861 6E646C65  var name="handle
6241DB00: 22206578 70723D22 74727565 222F3E0A  " expr="true"/>.
6241DB10: 0A20203C 63617463 68206576 656E743D  .  <catch event=
6241DB20: 2274656C 6570686F 6E652E64 6973636F  "telephone.disco
6241DB30: 6E6E6563 74222063 6F6E643D 2268616E  nnect" cond="han
6241DB40: 646C6522 3E0A2020 20203C61 73736967  dle">.    <assig
6241DB50: 6E206E61 6D653D22 68616E64 6C65      n name="handle
Jan   4 02:47:08 PDT: httpc_free: NULL pointer argument
Jan   4 02:47:09 PDT: HTTPC URL:http://1.7.100.1/vxml/test/root.vxml, MSG_DECODE_ERROR,
fd(6)
Jan   4 03:47:09 PDT: WARNING:httpc_msg_retry: msg(6325CDD4):http://vvv.com/vxml/prompts/5.au
```

Related Commands

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

Command	Description
debug vxml	Displays debugging messages for VoIP VoiceXML interactions.

debug http client cookie

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

debug http client cookie

no debug http client cookie

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

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

Examples

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

```
Router# debug http client cookie

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

Related Commands

Command	Description
http client cache memory	Configures the memory limits for the HTTP client cache.
http client cache refresh	Configures the refresh time for the HTTP client cache.

Command	Description
http client cookie	Enables the HTTP client to send and receive cookies.
show http client cookie	Displays cookies that are being stored by the HTTP client.

debug hw-module all upgrade

To enable debug messages for field-programmable devices (FPDs), use the **debug hw-module all upgrade**commandin privileged EXEC configuration mode. To disable debug messages, use the **no** form of the command.

debug hw-module all upgrade [**error**| **event**]

no debug hw-module all upgrade [**error**| **event**]

Syntax Description

all	Enable debug messaging for all supported modules in the system.
error	(Optional) Enables display of FPD upgrade error messages.
event	(Optional) Enables display of FPD upgrade event messages.

Command Default

No default behavior or values

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(18)SXE	Thiscommand was introduced.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SCB	This command was integrated into Cisco IOS Release 12.2(33)SCB. The FPD image upgrade is supported only for the SPAs inserted in the Cisco SIP 600 on a Cisco uBR100012 router.

Usage Guidelines

The **debug hw-module all upgrade**command is intended for use by Cisco Systems technical support personnel.

If you attempt to use this command without a SPA installed, or with an incompatible SPA installed, the keyword options are not provided.

Caution Because debugging output is assigned high priority in the CPU process, it can render the system unusable. For this reason, use **debug** commands only to troubleshoot specific problems or during troubleshooting sessions with Cisco Systems technical support personnel. Moreover, it is best to use **debug** commands during periods of lower network traffic and fewer users. Debugging during these periods decreases the likelihood that increased **debug** command processing overhead will affect system use.

For more information about FPD upgrades on SPA interface processors (SIPs) and shared port adapters (SPAs), refer to the Cisco 7600 Series Router SIP, SSC, and SPA Software Configuration Guide.

Examples

The following example enables FPD upgrade debug messages for all supported card types on the Cisco 7600 series router:

```
Router# debug hw-module all upgrade
```

debug hw-module subslot

To debug a shared port adapter SPA and all of its interfaces, use the **debug hw-module subslot**commandin privileged EXEC configuration mode.

debug hw-module subslot *slotsubslot* {**all**| **driver**| **fpga**| **if**| **mac**| **phy**| **tcam**| **upgrade** [**error**| **event**] **intr**| **force-intr**}

no debug hw-module subslot *slotsubslot* {**all**| **driver**| **fpga**| **if**| **mac**| **phy**| **tcam**| **upgrade** [**error**| **event**] **intr**| **force-intr**}

Syntax Description

slot	Chassis slot number. Refer to the appropriate hardware manual for slot information. For SIPs, refer to the platform-specific SPA hardware installation guide or the corresponding "Identifying Slots and Subslots for SIPs and SPAs" topic in the platform-specific SPA software configuration guide.
/ *subslot*	Secondary slot number on a SIP where a SPA is installed. Refer to the platform-specific SPA hardware installation guide and the corresponding "Specifying the Interface Address on a SPA" topic in the platform-specific SPA software configuration guide for subslot information.
all	Enables all SPA debug messages.
driver	Enables debug messages for SPA drivers.
fpga	Enables debug messages related to SPA field programmable gate array (FPGA) processing.
if	Enables debug messages related to SPA interface processing
mac	Enables debug messages related to SPA MAC driver processing.
phy	Enables debug messages related to SPA PHY driver processing.
tcam	Enables debug messages related to SPA ternary content addressable memory (TCAM) processing.

upgrade [**error** | **event**	Enables debug messages related to Field-Programmable Device (FPD) upgrade information. • **error** --Specifies that upgrade error messages are displayed. • **event** --Specifies that upgrade event messages are displayed.
intr	Enables debug messages related to SPA interrupts. **Caution** The **intr**option should be used only under the supervision of Cisco Systems technical support personnel and is not intended for production networks.
force-intr	Enables debug messages related to manually forced SPA interrupts. **Caution** The **force-intr**option should be used only under the supervision of Cisco Systems technical support personnel and is not intended for production networks.

Command Default No default behavior or values.

Command Modes Privileged EXEC (#)

Command History

Release	Modification
12.2(20)S2	Thiscommand was introduced.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.

Usage Guidelines The **debug hw-module subslot** command is intended for use by Cisco Systems technical support personnel.

If you attempt to use this command without a SPA installed, or with an incompatible SPA installed, the keyword options are not provided.

Caution Because debugging output is assigned high priority in the CPU process, it can render the system unusable. For this reason, use **debug** commands only to troubleshoot specific problems or during troubleshooting sessions with Cisco Systems technical support personnel. Moreover, it is best to use **debug** commands during periods of lower network traffic and fewer users. Debugging during these periods decreases the likelihood that increased **debug** command processing overhead will affect system use.

Examples

The following example enables interface debug messages for the 4-Port 10/100 Fast Ethernet located in the top subslot (0) of the MSC that is installed in slot 4 of the Cisco 7304 router and shows an interface being shut down and restarted:

```
Router# debug hw-module subslot 4/0 if
SPA 4xFE/2xGE interface debugging is on
Router# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# int fast 4/0/0
Router(config-if)# shut
Router(config-if)#
4d01h: Interface FastEthernet4/0/0, stopping the devices
4d01h: Interface FastEthernet4/0/0, Turning off the port LED
Router(config-if)#
4d01h: %LINK-5-CHANGED: Interface FastEthernet4/0/0, changed state to administratively down
4d01h: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet4/0/0, changed state to
 down
Router(config-if)#
Router(config-if)# no shut
Router(config-if)#
4d01h: Interface FastEthernet4/0/0, stopping the devices
4d01h: Interface FastEthernet4/0/0, clearing the MAC address filter table
4d01h: Interface FastEthernet4/0/0, Disabling promiscuous mode
4d01h: Interface FastEthernet4/0/0, setting the MAC address to 00b0.64ff.4480
4d01h: Interface FastEthernet4/0/0, Disabling promiscuous mode
4d01h: Interface FastEthernet4/0/0, configuring media_type = RJ45, speed = Auto Speed,
duplex = Auto Duplex, mode = auto-negotiation
4d01h: Interface FastEthernet4/0/0, starting the devices
4d01h: Interface FastEthernet4/0/0, clearing the hardware counters
4d01h: %LINK-3-UPDOWN: Interface FastEthernet4/0/0, changed state to up
4d01h: Interface FastEthernet4/0/0, Setting port LED to green
4d01h: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet4/0/0, changed state to
 up
```

Related Commands

Command	Description
show controllers fastethernet	Displays Fast Ethernet interface information, transmission statistics and errors, and applicable MAC destination address and VLAN filtering tables.
show controllers gigabitethernet	Displays Gigabit Ethernet interface information, transmission statistics and errors, and applicable MAC destination address and VLAN filtering tables.
show tcam-mgr subslot	Displays TCAM manager information for SPAs.
test hw-module subslot mac	Tests the MAC device on a SPA.
test hw-module subslot phy	Tests the PHY device on a SPA.

Command	Description
test hw-module subslot tcam	Tests the TCAM device on a SPA.
test tcam-mgr subslot	Tests the TCAM manager for a SPA.

debug hw-module subslot commands

To enable debug messages for control plane configuration and commands on a shared port adapter (SPA), use the **debug hw-module subslot commands**commandin privileged EXEC configuration mode. To disable debug messages, use the **no** form of the command.

debug hw-module subslot {*slotsubslot*| **all**} **commands**

no debug hw-module subslot {*slotsubslot*| **all**} **commands**

Syntax Description

slot	Chassis slot number. Refer to the appropriate hardware manual for slot information. For SIPs, refer to the platform-specific SPA hardware installation guide or the corresponding “Identifying Slots and Subslots for SIPs and SPAs” topic in the platform-specific SPA software configuration guide.
/ *subslot*	Secondary slot number on a SPA interface processor (SIP) where a SPA is installed. Refer to the platform-specific SPA hardware installation guide and the corresponding “Specifying the Interface Address on a SPA” topic in the platform-specific SPA software configuration guide for subslot information.
all	Enable debug messaging for all supported modules in the system.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(18)SXE	Thiscommand was introduced.
12.0(31)S	This command was integrated into Cisco IOS Release 12.0(31)S.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

The **debug hw-module subslot**commands are intended for use by Cisco Systems technical support personnel.

If you attempt to use a **debug hw-module subslot** command without a SPA installed, or with an incompatible SPA installed, the keyword options are not provided.

Caution Because debugging output is assigned high priority in the CPU process, it can render the system unusable. For this reason, use **debug** commands only to troubleshoot specific problems or during troubleshooting sessions with Cisco Systems technical support personnel. Moreover, it is best to use **debug** commands during periods of lower network traffic and fewer users. Debugging during these periods decreases the likelihood that increased **debug** command processing overhead will affect system use.

Examples

The following example enables control plane debug messages for the SPA located in the top subslot (0) of the SIP that is installed in slot 4 of a router:

```
Router# debug hw-module subslot 4/0 commands
```

debug hw-module subslot errors

To enable debug messages for error handling and race conditions on a shared port adapter (SPA), use the **debug hw-module subslot errors**commandin privileged EXEC configuration mode. To disable debug messages, use the **no** form of the command.

debug hw-module subslot {*slotsubslot*| **all**} **errors**

no debug hw-module subslot {*slotsubslot*| **all**} **errors**

Syntax Description

slot	Chassis slot number. Refer to the appropriate hardware manual for slot information. For SIPs, refer to the platform-specific SPA hardware installation guide or the corresponding “Identifying Slots and Subslots for SIPs and SPAs” topic in the platform-specific SPA software configuration guide.
/ *subslot*	Secondary slot number on a SPA interface processor (SIP) where a SPA is installed. Refer to the platform-specific SPA hardware installation guide and the corresponding “Specifying the Interface Address on a SPA” topic in the platform-specific SPA software configuration guide for subslot information.
all	Enable debug messaging for all supported modules in the system.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(18)SXE	Thiscommand was introduced.
12.0(31)S	This command was integrated into Cisco IOS Release 12.0(31)S.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

The **debug hw-module subslot**commands are intended for use by Cisco Systems technical support personnel.

If you attempt to use a **debug hw-module subslot** command without a SPA installed, or with an incompatible SPA installed, the keyword options are not provided.

Caution Because debugging output is assigned high priority in the CPU process, it can render the system unusable. For this reason, use **debug** commands only to troubleshoot specific problems or during troubleshooting sessions with Cisco Systems technical support personnel. Moreover, it is best to use **debug** commands during periods of lower network traffic and fewer users. Debugging during these periods decreases the likelihood that increased **debug** command processing overhead will affect system use.

Examples

The following example enables error handling debug messages for the SPA located in the top subslot (0) of the SPA that is installed in slot 4 of a router:

```
Router# debug hw-module subslot 4/0 errors
```

debug hw-module subslot events

To enable debug messages for control plane event notifications on a shared port adapter (SPA), use the **debug hw-module subslot events**commandin privileged EXEC configuration mode. To disable debug messages, use the **no** form of the command.

debug hw-module subslot {**slotsubslot| all**} **events**

no debug hw-module subslot {**slotsubslot| all**} **events**

Syntax Description

slot	Chassis slot number. Refer to the appropriate hardware manual for slot information. For SIPs, refer to the platform-specific SPA hardware installation guide or the corresponding “Identifying Slots and Subslots for SIPs and SPAs” topic in the platform-specific SPA software configuration guide.
/ *subslot*	Secondary slot number on a SPA interface processor (SIP) where a SPA is installed. Refer to the platform-specific SPA hardware installation guide and the corresponding “Specifying the Interface Address on a SPA” topic in the platform-specific SPA software configuration guide for subslot information.
all	Enable debug messaging for all supported modules in the system.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(18)SXE	Thiscommand was introduced.
12.0(31)S	This command was integrated into Cisco IOS Release 12.0(31)S.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

The **debug hw-module subslot**commands are intended for use by Cisco Systems technical support personnel.

If you attempt to use a **debug hw-module subslot** command without a SPA installed, or with an incompatible SPA installed, the keyword options are not provided.

Caution Because debugging output is assigned high priority in the CPU process, it can render the system unusable. For this reason, use **debug** commands only to troubleshoot specific problems or during troubleshooting sessions with Cisco Systems technical support personnel. Moreover, it is best to use **debug** commands during periods of lower network traffic and fewer users. Debugging during these periods decreases the likelihood that increased **debug** command processing overhead will affect system use.

Examples

The following example enables control plane event messages for the SPA located in the top subslot (0) of the SIP that is installed in slot 4 of a router:

```
Router# debug hw-module subslot 4/0 events
```

debug hw-module subslot interrupts

To enable debug messages for interrupt handling on a shared port adapter (SPA), use the **debug hw-module subslot interrupts**commandin privileged EXEC configuration mode. To disable debug messages, use the **no** form of the command.

debug hw-module subslot interruptsslot/subslot | allinterface

no debug hw-module subslot interruptsslot /subslot | allinterface

Syntax Description

slot	Chassis slot number. Refer to the appropriate hardware manual for slot information. For SIPs, refer to the platform-specific SPA hardware installation guide or the corresponding “Identifying Slots and Subslots for SIPs and SPAs” topic in the platform-specific SPA software configuration guide.
/ *subslot*	Secondary slot number on a SPA interface processor (SIP) where a SPA is installed. Refer to the platform-specific SPA hardware installation guide and the corresponding “Specifying the Interface Address on a SPA” topic in the platform-specific SPA software configuration guide for subslot information.
all	Enable debug messaging for all supported modules in the system.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(18)SXE	Thiscommand was introduced.
12.0(31)S	This command was integrated into Cisco IOS Release 12.0(31)S.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

The **debug hw-module subslot**commands are intended for use by Cisco Systems technical support personnel.

If you attempt to use a **debug hw-module subslot** command without a SPA installed, or with an incompatible SPA installed, the keyword options are not provided.

Caution Because debugging output is assigned high priority in the CPU process, it can render the system unusable. For this reason, use **debug** commands only to troubleshoot specific problems or during troubleshooting sessions with Cisco Systems technical support personnel. Moreover, it is best to use **debug** commands during periods of lower network traffic and fewer users. Debugging during these periods decreases the likelihood that increased **debug** command processing overhead will affect system use.

Examples

The following example enables interrupt handling debug messages for the SPA located in the top subslot (0) of the SIP that is installed in slot 4 of a router :

```
Router# debug hw-module subslot 4/0 interrupts
```

debug hw-module subslot ipcshim

To enable debug messages for Inter-Process Communication (IPC) shim application processing for all supported modules in the system, use the **debug hw-module subslot ipcshim**commandin privileged EXEC configuration mode. To disable debug messages, use the **no** form of the command.

debug hw-module subslot all ipcshim

no debug hw-module subslot all ipcshim

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(18)SXE	Thiscommand was introduced.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

The **debug hw-module subslot**commands are intended for use by Cisco Systems technical support personnel.

The **debug hw-module subslot ipcshim**command is only supported by certain shared port adapters (SPAs).

Caution Because debugging output is assigned high priority in the CPU process, it can render the system unusable. For this reason, use **debug** commands only to troubleshoot specific problems or during troubleshooting sessions with Cisco Systems technical support personnel. Moreover, it is best to use **debug** commands during periods of lower network traffic and fewer users. Debugging during these periods decreases the likelihood that increased **debug** command processing overhead will affect system use.

Examples

The following example enables IPC SHIM application debug messages for all supported modules in the router:

```
Router# debug hw-module subslot all ipcshim
```

debug hw-module subslot oir

To enable debug messages for online insertion and removal (OIR) processing on a shared port adapter (SPA), use the **debug hw-module subslot oir**commandin privileged EXEC configuration mode. To disable debug messages, use the **no** form of the command.

debug hw-module subslot {*slotsubslot*| **all**} **oir** {**plugin**| **state-machine**}

no debug hw-module subslot {*slotsubslot*| **all**} **oir** {**plugin**| **state-machine**}

Syntax Description

slot	Chassis slot number. Refer to the appropriate hardware manual for slot information. For SIPs, refer to the platform-specific SPA hardware installation guide or the corresponding "Identifying Slots and Subslots for SIPs and SPAs" topic in the platform-specific SPA software configuration guide.
/ *subslot*	Secondary slot number on a SPA interface processor (SIP) where a SPA is installed. Refer to the platform-specific SPA hardware installation guide and the corresponding "Specifying the Interface Address on a SPA" topic in the platform-specific SPA software configuration guide for subslot information.
all	Enable debug messaging for all supported modules in the system.
plugin	Enable debug messaging for platform-provided plugin routines.
state-machine	Enable debug messaging for SPA OIR state machines.

Command Default

No default behavior or values

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(18)SXE	Thiscommand was introduced.
12.0(31)S	This command was integrated into Cisco IOS Release 12.0(31)S.

Release	Modification
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.

Usage Guidelines

The **debug hw-module subslot**commands are intended for use by Cisco Systems technical support personnel.

If you attempt to use a **debug hw-module subslot** command without a SPA installed, or with an incompatible SPA installed, the keyword options are not provided.

Caution Because debugging output is assigned high priority in the CPU process, it can render the system unusable. For this reason, use **debug** commands only to troubleshoot specific problems or during troubleshooting sessions with Cisco Systems technical support personnel. Moreover, it is best to use **debug** commands during periods of lower network traffic and fewer users. Debugging during these periods decreases the likelihood that increased **debug** command processing overhead will affect system use.

Examples

The following example shows enabling of OIR plugin debug messages for the SPA located in subslot 1 of the SIP that is installed in slot 4 of the router , and the corresponding messages during a SPA reload:

```
Router# debug hw-module subslot 4/1 oir plugin
WARNING: This command is not intended for production use
and should only be used under the supervision of
Cisco Systems technical support personnel.
SPA subslot 4/1:
   SPA specific oir handling debugging is on
Router# hw-module subslot 4/1 reload
Router#
Mar 26 01:35:04: cwrp_handle_spa_oir_tsm_event: subslot 4/1 event=9
Mar 26 01:35:04: cwrp_handle_spa_oir_tsm_event: subslot 4/1 event=1
Router#
Mar 26 01:35:09: cwrp_handle_spa_oir_tsm_event: subslot 4/1 event=0
Mar 26 01:35:10: cwrp_handle_spa_oir_tsm_event: subslot 4/1 event=2
```

debug hw-module subslot periodic

To enable debug messages for periodic processing on a shared port adapter (SPA), use the **debug hw-module subslot periodic**commandin privileged EXEC configuration mode. To disable debug messages, use the **no** form of the command.

debug hw-module subslot {*slotsubslot*| **all**} **periodic**

no debug hw-module subslot {*slotsubslot*| **all**} **periodic**

Syntax Description

slot	Chassis slot number. Refer to the appropriate hardware manual for slot information. For SIPs, refer to the platform-specific SPA hardware installation guide or the corresponding "Identifying Slots and Subslots for SIPs and SPAs" topic in the platform-specific SPA software configuration guide.
/ *subslot*	Secondary slot number on a SPA interface processor (SIP) where a SPA is installed. Refer to the platform-specific SPA hardware installation guide and the corresponding "Specifying the Interface Address on a SPA" topic in the platform-specific SPA software configuration guide for subslot information.
all	Enable debug messaging for all supported modules in the system.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(18)SXE	Thiscommand was introduced.
12.0(31)S	This command was integrated into Cisco IOS Release 12.0(31)S.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

The **debug hw-module subslot**commands are intended for use by Cisco Systems technical support personnel.

If you attempt to use a **debug hw-module subslot** command without a SPA installed, or with an incompatible SPA installed, the keyword options are not provided.

Caution Because debugging output is assigned high priority in the CPU process, it can render the system unusable. For this reason, use **debug** commands only to troubleshoot specific problems or during troubleshooting sessions with Cisco Systems technical support personnel. Moreover, it is best to use **debug** commands during periods of lower network traffic and fewer users. Debugging during these periods decreases the likelihood that increased **debug** command processing overhead will affect system use.

Examples

The following example enables periodic processing debug messages for the SPA located in the top subslot (0) of the SIP that is installed in slot 4 of a router:

```
Router# debug hw-module subslot 4/0 periodic
```