

debug x25

To display information about all X.25 traffic or a specific X.25 service class, use the **debug x25** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug x25 [**only** | **cmns** | **xot**] [**events** | **all**] [**dump**]

no debug x25 [**only** | **cmns**] [**events** | **all**] [**dump**]

Syntax Description

| | |
|---------------|---|
| only | (Optional) Displays information about X.25 services only. |
| cmns | (Optional) Displays information about CMNS services only. |
| xot | (Optional) Displays information about XOT services only. |
| events | (Optional) Displays all traffic except Data and Receiver Ready (RR) packets. |
| all | (Optional) Displays all traffic. This is the default. |
| dump | (Optional) Displays the encoded packet contents in hexadecimal and ASCII formats. |

Defaults

All traffic is displayed.

Command Modes

Privileged EXEC

Command History

| Release | Modification |
|----------|--|
| 10.0 | This command was introduced. |
| 12.0(5)T | For Domain Name System (DNS)-based X.25 routing, additional functionality was added to the debug x25 events command to describe the events that occur while the X.25 address is being resolved to an IP address using a DNS server. The debug domain command can be used along with debug x25 events to observe the whole DNS-based X.25 routing data flow. |
| 12.0(7)T | For the X.25 Closed User Groups (CUGs) feature, functionality was added to the debug x25 events command to describe events that occur during CUG activity. |
| 12.2(8)T | The debug x25 events command was enhanced to display events specific to Record Boundary Preservation protocol. |
| 12.3(2)T | The dump keyword was added. |

Usage Guidelines



Caution

The X.25 debug commands can generate large amounts of debugging output. If logging of debug output to the router console is enabled (the default condition), this output may fill the console buffer, preventing the router from processing packets until the contents of the console buffer have been printed.

The **debug x25**, **debug x25 interface**, **debug x25 vc**, and **debug x25 xot** commands all generate the same basic output. The **debug x25 interface**, **debug x25 vc**, and **debug x25 xot** commands narrow the scope of the **debug x25** output to specific virtual circuits or types of traffic.

This command is particularly useful for diagnosing problems encountered when placing calls. The **debug x25 all** output includes data, control messages, and flow control packets for all virtual circuits of the router.

All **debug x25** commands can take either the **events** or the **all** keyword. The keyword **all** is the default and causes all packets meeting the other debug criteria to be reported. The keyword **events** omits reports of any Data or RR flow control packets; the normal flow of data and RR packets is commonly large and less interesting to the user, so event reporting can significantly decrease the processor load induced by debug reporting.

Use the **dump** keyword to display the entire contents, including user data, of X.25 packets. The encoded X.25 packet contents are displayed after the standard packet description. The output includes the offset into the packet and the display of the data in both hexadecimal and ASCII formats.



Caution

The X.25 packet information that is reported by using the **dump** keyword may contain sensitive data; for example, clear-text account identities and passwords. The network access policies and router configuration should be controlled appropriately to address this risk.

Examples

The following is sample output from the **debug x25** command, displaying output concerning the functions X.25 restart, call setup, data exchange, and clear:

```
Router# debug x25

Serial0: X.25 I R/Inactive Restart (5) 8 lci 0
      Cause 7, Diag 0 (Network operational/No additional information)
Serial0: X.25 O R3 Restart Confirm (3) 8 lci 0
Serial0: X.25 I P1 Call (15) 8 lci 1
From(6): 170091 To(6): 170090
      Facilities: (0)
      Call User Data (4): 0xCC000000 (ip)
Serial0: X.25 O P3 Call Confirm (3) 8 lci 1
Serial0: X.25 I D1 Data (103) 8 lci 1 PS 0 PR 0
Serial0: X.25 O D1 Data (103) 8 lci 1 PS 0 PR 1
Serial0: X.25 I P4 Clear (5) 8 lci 1
      Cause 9, Diag 122 (Out of order/Maintenance action)
Serial0: X.25 O P7 Clear Confirm (3) 8 lci 1
```

debug x25 events for DNS-Based X.25 Routing: Example

The following example of the **debug x25** command with the **events** keyword shows output related to the DNS-Based X.25 Routing feature. It shows messages concerning access to the DNS server. In the example, nine alternate addresses for one XOT path are entered into the DNS server database. All nine addresses are returned to the host cache of the router by the DNS server. However, only six addresses will be used during the XOT switch attempt because this is the limit that XOT allows.

```
Router# debug x25 events

00:18:25:Serial1:X.25 I R1 Call (11) 8 lci 1024
00:18:25: From (0): To (4):444
00:18:25: Facilities:(0)
00:18:25: Call User Data (4):0x01000000 (pad)
00:18:25:X.25 host name sent for DNS lookup is "444"
00:18:26:%3-TRUNCATE_ALT_XOT_DNS_DEST:Truncating excess XOT addresses (3)
returned by DNS
```

```

00:18:26:DNS got X.25 host mapping for "444" via network
00:18:32:[10.1.1.8 (pending)]:XOT open failed (Connection timed out; remote host not
responding)
00:18:38:[10.1.1.7 (pending)]:XOT open failed (Connection timed out; remote host not
responding)
00:18:44:[10.1.1.6 (pending)]:XOT open failed (Connection timed out; remote host not
responding)
00:18:50:[10.1.1.5 (pending)]:XOT open failed (Connection timed out; remote host not
responding)
00:18:56:[10.1.1.4 (pending)]:XOT open failed (Connection timed out; remote host not
responding)
00:20:04:[10.1.1.3,1998/10.1.1.3,11007]:XOT O P2 Call (17) 8 lci 1
00:20:04: From (0): To (4):444
00:20:04: Facilities:(6)
00:20:04: Packet sizes:128 128
00:20:04: Window sizes:2 2
00:20:04: Call User Data (4):0x01000000 (pad)
00:20:04:[10.1.1.3,1998/10.1.1.3,11007]:XOT I P2 Call Confirm (11) 8 lci 1
00:20:04: From (0): To (0):
00:20:04: Facilities:(6)
00:20:04: Packet sizes:128 128
00:20:04: Window sizes:2 2
00:20:04:Serial1:X.25 O R1 Call Confirm (5) 8 lci 1024
00:20:04: From (0): To (0):
00:20:04: Facilities:(0)

```

Record Boundary Preservation: Examples

The following examples show output for the **x25 debug** command with the **events** keyword when record boundary preservation (RBP) has been configured using the **x25 map rbp local** command.

The following display shows establishment of connection:

```

X25 RBP:Incoming connection for port 9999 from 10.0.155.30 port 11001
Serial0/1:X.25 O R1 Call (10) 8 lci 64
  From (5):13133 To (5):12131
  Facilities:(0)
Serial0/1:X.25 I R1 Call Confirm (3) 8 lci 64

```

The following display shows that the X.25 call was cleared by the X.25 host:

```

Serial0/1:X.25 I R1 Clear (5) 8 lci 64
  Cause 0, Diag 122 (DTE originated/Maintenance action)
X25 RBP:X.25 circuit cleared
Serial0/1:X.25 O R1 Clear Confirm (3) 8 lci 64

```

The following display shows that the TCP session has terminated:

```

[10.0.155.30,11000/10.0.155.33,9999]:TCP receive error, End of data transfer
X25 RBP:End of data transfer
Serial0/1:X.25 O R1 Clear (5) 8 lci 64
Cause 9, Diag 122 (Out of order/Maintenance action)
Serial0/1:X.25 I R1 Clear Confirm (3) 8 lci 64

```

The following examples show output of the **x25 debug** command with the **events** keyword when RBP has been configured using the **x25 pvc rbp local** command.

The following display shows data on the permanent virtual circuit (PVC) before the TCP session has been established:

```

X25 RBP:Data on unconnected PVC
Serial1/0:X.25 O D1 Reset (5) 8 lci 1
  Cause 0, Diag 113 (DTE originated/Remote network problem)
Serial1/0:X.25 I D2 Reset Confirm (3) 8 lci 1

```

The following display shows establishment of connection:

```
X25 RBP:Incoming connection for port 9998 from 2.30.0.30 port 11002
Serial1/0:X.25 O D1 Reset (5) 8 lci 1
Cause 0, Diag 0 (DTE originated/No additional information)
Serial1/0:X.25 I D2 Reset Confirm (3) 8 lci 1
```

The following display shows termination of connection when the X.25 PVC was reset:

```
Serial1/0:X.25 I D1 Reset (5) 8 lci 1
Cause 15, Diag 122 (Network operational (PVC)/Maintenance action)
X25 RBP:Reset packet received
Serial1/0:X.25 O D3 Reset Confirm (3) 8 lci 1
```

The following display shows that the TCP session has terminated:

```
[2.30.0.30,11003/2.30.0.33,9998]:TCP receive error, End of data transfer
X25 RBP:End of data transfer
Serial1/0:X.25 O D1 Reset (5) 8 lci 1
Cause 0, Diag 113 (DTE originated/Remote network problem)
Serial1/0:X.25 I D2 Reset Confirm (3) 8 lci 1
```

The following examples show output of the **x25 debug** command with the **events** keyword when RBP has been configured using the **x25 map rbp remote** command.

The following display shows that the X.25 call was cleared:

```
Serial0/1:X.25 I R1 Clear (5) 8 lci 1024
Cause 0, Diag 122 (DTE originated/Maintenance action)
X25 RBP:X.25 circuit cleared
Serial0/1:X.25 O R1 Clear Confirm (3) 8 lci 1024
```

The following display shows that the X.25 call was reset:

```
Serial0/1:X.25 I D1 Reset (5) 8 lci 1024
Cause 0, Diag 122 (DTE originated/Maintenance action)
X25 RBP:Reset packet received
Serial0/1:X.25 O R1 Clear (5) 8 lci 1024
Cause 9, Diag 122 (Out of order/Maintenance action)
Serial0/1:X.25 I R1 Clear Confirm (3) 8 lci 1024
```

The following examples show output of the **x25 debug** command with the **events** keyword when RBP has been configured using the **x25 pvc rbp remote** command.

The following display shows that the X.25 PVC has been reset:

```
Serial0/0:X.25 I D1 Reset (5) 8 lci 1
Cause 0, Diag 122 (DTE originated/Maintenance action)
X25 RBP:Reset packet received
Serial0/0:X.25 O D2 Reset Confirm (3) 8 lci 1
```

The following display shows that the connection was terminated when the X.25 interface was restarted:

```
Serial0/0:X.25 I R1 Restart (5) 8 lci 0
Cause 0, Diag 122 (DTE originated/Maintenance action)
X25 RBP:X.25 PVC inactive
Serial0/0:X.25 O R2 Restart Confirm (3) 8 lci 0
Serial0/0:X.25 O D1 Reset (5) 8 lci 1
Cause 1, Diag 113 (Out of order (PVC)/Remote network problem)
Serial0/0:X.25 I D3 Reset Confirm (3) 8 lci 1
```

debug x25 dump Example

The following is sample output for the **debug x25 dump** command. The encoded X.25 packet contents are displayed after the standard packet description. The output includes the offset into the packet and the display of the data in both hexadecimal and ASCII formats.

```
Router# debug x25 dump

Serial1: X.25 O R/Inactive Restart (5) 8 lci 0
Cause 0, Diag 0 (DTE originated/No additional information)
0: 1000FB00 00 ..{..
Serial1: X.25 I R2 Restart (5) 8 lci 0
Cause 7, Diag 0 (Network operational/No additional information)
0: 1000FB ..{
3: 0700 ..
Serial1: X.25 I R1 Call (13) 8 lci 1
From (4): 2501 To (4): 2502
Facilities: (0)
Call User Data (4): 0xCC000000 (ip)
0: 10010B 44250225 0100CC00 ...D%.%.L.
11: 0000 ..
Serial1: X.25 O R1 Call Confirm (3) 8 lci 1
0: 10010F ...
Serial1: X.25 I D1 Data (103) 8 lci 1 PS 0 PR 0
0: 100100 45000064 00000000 ...E..d....
11: FF01A764 0A190001 0A190002 0800CBFB ..'d.....K{
27: 0B1E22CA 00000000 00028464 ABCDABCD .."J.....d+M+M
43: ABCDABCD ABCDABCD ABCDABCD ABCDABCD +M+M+M+M+M+M+M+M
59: ABCDABCD ABCDABCD ABCDABCD ABCDABCD +M+M+M+M+M+M+M+M
75: ABCDABCD ABCDABCD ABCDABCD ABCDABCD +M+M+M+M+M+M+M+M
91: ABCDABCD ABCDABCD ABCDABCD +M+M+M+M+M+M
Serial1: X.25 O D1 Data (103) 8 lci 1 PS 0 PR 1
0: 100120 45000064 00000000 .. E..d....
11: FF01A764 0A190002 0A190001 0000D3FB ..'d.....S{
27: 0B1E22CA 00000000 00028464 ABCDABCD .."J.....d+M+M
43: ABCDABCD ABCDABCD ABCDABCD ABCDABCD +M+M+M+M+M+M+M+M
59: ABCDABCD ABCDABCD ABCDABCD ABCDABCD +M+M+M+M+M+M+M+M
75: ABCDABCD ABCDABCD ABCDABCD ABCDABCD +M+M+M+M+M+M+M+M
91: ABCDABCD ABCDABCD ABCDABCD +M+M+M+M+M+M
Serial1: X.25 I R1 Clear (5) 8 lci 1
Cause 9, Diag 122 (Out of order/Maintenance action)
0: 100113 097A ....z
Serial1: X.25 O R1 Clear Confirm (3) 8 lci 1
0: 100117 .
```

Table 337 describes significant fields shown in the displays.

Table 337 debug x25 Field Descriptions

| Field | Description |
|---------|--|
| Serial0 | Interface on which the X.25 event occurred. |
| X.25 | Type of event this message describes. |
| I | Letter indicating whether the X.25 packet was input (I) or output (O) through the interface. |

Table 337 *debug x25 Field Descriptions (continued)*

| Field | Description |
|--------------|--|
| R3 | <p>State of the service or virtual circuit (VC). Possible values follow:</p> <p>R/Inactive—Packet layer awaiting link layer service</p> <p>R1—Packet layer ready</p> <p>R2—Data terminal equipment (DTE) restart request</p> <p>R3—DCE restart indication</p> <p>P/Inactive—VC awaiting packet layer service</p> <p>P1—Idle</p> <p>P2—DTE waiting for DCE to connect CALL</p> <p>P3—DCE waiting for DTE to accept CALL</p> <p>P4—Data transfer</p> <p>P5—CALL collision</p> <p>P6—DTE clear request</p> <p>P7—DCE clear indication</p> <p>D/Inactive—VC awaiting setup</p> <p>D1—Flow control ready</p> <p>D2—DTE reset request</p> <p>D3—DCE reset indication</p> <p>Refer to Annex B of the <i>ITU-T Recommendation X.25</i> for more information on these states.</p> |

Table 337 debug x25 Field Descriptions (continued)

| Field | Description |
|---|--|
| Restart | <p>The type of X.25 packet. Possible values follow:</p> <p>R Events</p> <ul style="list-style-type: none"> • Restart • Restart Confirm • Diagnostic <p>P Events</p> <ul style="list-style-type: none"> • Call • Call Confirm • Clear • Clear Confirm <p>D Events</p> <ul style="list-style-type: none"> • Reset • Reset Confirm <p>D1 Events</p> <ul style="list-style-type: none"> • Data • Receiver Not Ready (RNR) • RR (Receiver Ready) • Interrupt • Interrupt Confirm <p>XOT Overhead</p> <ul style="list-style-type: none"> • PVC Setup <p>Refer to RFC 1613 <i>Cisco Systems X.25 over TCP (XOT)</i> for information about the XOT PVC Setup packet type.</p> |
| (5) | Number of bytes in the packet. |
| 8 | Modulo of the virtual circuit. Possible values are 8 and 128. |
| lci 0 | VC number. Refer to Annex A of the <i>ITU-T Recommendation X.25</i> for information on VC assignment. |
| Cause 7 | Code indicating the event that triggered the packet. The Cause field can appear only in entries for Clear, Reset, and Restart packets. Possible values for the Cause field can vary, depending on the type of packet. Refer to the appendix “X.25 Cause and Diagnostic Codes” for an explanation of these codes. |
| Diag 0 | Code providing an additional hint of what, if anything, went wrong. The Diag field can appear only in entries for Clear, Diagnostic (as “error 0”), Reset, and Restart packets. Refer to the appendix “X.25 Cause and Diagnostic Codes” for an explanation of these codes. |
| (Network operational/ No additional information) | The standard explanations of the Cause and Diagnostic codes (<i>cause/diag</i>). |

Table 337 *debug x25 Field Descriptions (continued)*

| Field | Description |
|---------------------|---|
| From (6):170091 | Source address. (6) indicates the number of digits in the address that follows. The source address is part of the address block that may be encoded in Call Setup packets. |
| To (6): 170090 | Destination address. (6) indicates the number of digits in the address that follows. The destination address is part of the address block that may be encoded in Call Setup packets. |
| Facilities:(0) | Indicates that a facilities block is encoded and that it consists of 0 bytes. A breakdown of the encoded facilities (if any) follows. |
| Call User Data (4): | Indicates that the Call User Data (CUD) field is present and consists of 4 bytes. |
| 0xCC000000 (ip) | Protocol identifier (PID). This subfield of the CUD field is presented in the output as a hexadecimal string followed by the name of the protocol (in this case, IP) that the string represents. Any bytes following the PID are designated “user data” and may be used by an application separately from the PID. |

Related Commands

| Command | Description |
|----------------------------|--|
| debug x25 interface | Displays information about a specific X.25 or CMNS context or virtual circuit. |
| debug x25 vc | Displays information about traffic for all virtual circuits that use a given number. |
| debug x25 xot | Displays information about traffic to or from a specific XOT host. |

debug x25 annexg

To display information about Annex G (X.25 over Frame Relay) events, use the **debug x25 annexg** command. To disable debugging output, use the **no** form of this command.

debug x25 annexg

no debug x25 annexg

Syntax Description This command has no arguments or keywords.

Command Modes Privileged EXEC

| Command History | Release | Modification |
|-----------------|---------|------------------------------|
| | 12.0 T | This command was introduced. |

Usage Guidelines It is generally recommended that the **debug x25 annexg** command be used only when specifically requested by Cisco TAC to obtain information about a problem with an Annex G configuration. The messages displayed by the **debug x25 annexg** command are meant to aid in the diagnosing of internal errors.



Caution

The X.25 debug commands can generate large amounts of debugging output. If logging of debug output to the router console is enabled (the default condition), this output may fill the console buffer, preventing the router from processing packets until the contents of the console buffer have been printed.

Examples The following shows sample output from the **debug x25 annexg** command for a Frame Relay data-link connection identifier (DLCI) configured for Annex G operation:

```
Router# debug x25 annexg

Jul 31 05:23:20.316:annexg_process_events:DLCI 18 attached to interface Serial2/0:0 is
ACTIVE
Jul 31 05:23:20.316:annexg_ctxt_create:Creating X.25 context over Serial2/0:0 (DLCI:18
using X.25 profile:OMC), type 10, len 2, addr 00 12
Jul 31 05:23:20.316:annexg_create_lower_layer:Se2/0:0 DLCI 18, payload 1606, overhead 2
Jul 31 05:23:20.320:annexg_restart_tx:sending pak to Serial2/0:0
Jul 31 05:23:23.320:annexg_restart_tx:sending pak to Serial2/0:0
```

[Table 338](#) describes significant fields shown in the display.

Table 338 *debug x25 annexg Field Descriptions*

| Field | Description |
|--------------|---|
| payload | Amount of buffer space available per message before adding Frame Relay and device-specific headers. |
| overhead | The length of the Frame Relay header and any device-specific header that may be needed. |

Related Commands

| Command | Description |
|----------------------------|---|
| debug x25 | Displays information about all X.25 traffic or a specific X.25 service class. |
| debug x25 interface | Displays information about specific X.25, Annex G or CMN contexts or virtual circuits that occur on the identified interface. |
| debug x25 vc | Displays information about traffic for all virtual circuits that have a given number. |

debug x25 aodi

To display information about an interface running PPP over an X.25 session, use the **debug x25 aodi** command. To disable debugging output, use the **no** form of this command.

debug x25 aodi

no debug x25 aodi

Syntax Description This command has no arguments or keywords.

Command Modes Privileged EXEC

| Command History | Release | Modification |
|-----------------|---------|------------------------------|
| | 10.0 | This command was introduced. |

Usage Guidelines Use the **debug x25 aodi** command to display interface PPP events running over an X.25 session and to debug X.25 connections between a client and server configured for Always On/Dynamic ISDN (AO/DI).

Examples The following examples show the normal sequence of events for both the AO/DI client and the server sides:

Client Side

Router# **debug x25 aodi**

```

PPP-X25: Virtual-Access1: Initiating AODI call request
PPP-X25: Bringing UP X.25 AODI VC
PPP-X25: AODI Client Call Confirm Event Received
PPP-X25: Cloning interface for AODI is Di1
PPP-X25: Queuing AODI Client Map Event
PPP-X25: Event:AODI Client Map
PPP-X25: Created interface Vi2 for AODI service
PPP-X25: Attaching primary link Vi2 to Di1
PPP-X25: Cloning Vi2 for AODI service using Di1
PPP-X25: Vi2: Setting the PPP call direction as OUT
PPP-X25: Vi2: Setting vectors for RFC1598 operation on BRI3/0:0 VC 0
PPP-X25: Vi2: Setting the interface default bandwidth to 10 Kbps
PPP-X25: Virtual-Access2: Initiating AODI call request
PPP-X25: Bringing UP X.25 AODI VC
PPP-X25: AODI Client Call Confirm Event Received

```

Server Side

Router# **debug x25 aodi**

```

PPP-X25: AODI Call Request Event Received
PPP-X25: Event:AODI Incoming Call Request
PPP-X25: Created interface Vi1 for AODI service
PPP-X25: Attaching primary link Vi1 to Di1

```

```
PPP-X25: Cloning Vi1 for AODI service using Di1
PPP-X25: Vi1: Setting vectors for RFC1598 operation on BRI3/0:0 VC 1
PPP-X25: Vi1: Setting the interface default bandwidth to 10 Kbps
PPP-X25: Binding X.25 VC 1 on BRI3/0:0 to Vi1
```

debug x25 interface

To display information about the specific X.25, Annex G or Connection Mode Network Service (CMN) contexts or virtual circuits that occur on the identified interface, use the **debug x25 interface** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

```
debug x25 interface {serial-interface | cmns-interface [mac mac-address]} [vc number] [events | all] [dump]
```

```
no debug x25 interface {serial-interface | cmns-interface [mac mac-address]} [vc number] [events | all] [dump]
```

Syntax Description

| | |
|-------------------------------|---|
| <i>serial-interface</i> | Serial interface number that is configured for X.25 or Annex G service. |
| <i>cmns-interface</i> | Interface supporting CMNS traffic and, if specified, the MAC address of a remote host. The interface type can be Ethernet, Token Ring, or FDDI. |
| mac <i>mac-address</i> | (Optional) MAC address of the CMNS interface and remote host. |
| vc number | (Optional) Virtual circuit number. Range is from 1 to 4095. |
| events | (Optional) Displays all traffic except Data and Receiver Ready (RR) packets. |
| all | (Optional) Displays all traffic. This is the default. |
| dump | (Optional) Displays the encoded packet contents in hexadecimal and ASCII formats. |

Defaults

All traffic is displayed.

Command Modes

Privileged EXEC

Command History

| Release | Modification |
|----------|------------------------------------|
| 10.0 | This command was introduced. |
| 12.3(2)T | The dump keyword was added. |

Usage Guidelines



Caution

The X.25 debug commands can generate large amounts of debugging output. If logging of debug output to the router console is enabled (the default condition), this output may fill the console buffer, preventing the router from processing packets until the contents of the console buffer have been printed.

The **debug x25**, **debug x25 interface**, **debug x25 vc**, and **debug x25 xot** commands all generate the same basic output. The **debug x25 interface**, **debug x25 vc**, and **debug x25 xot** commands narrow the scope of the **debug x25** output to specific virtual circuits or types of traffic.

The **debug x25 interface** command is useful for diagnosing problems encountered with a single X.25 or CMNS host or virtual circuit.

The keyword **all** is the default and causes all packets meeting the other debug criteria to be reported. The keyword **events** omits reports of any Data or RR flow control packets; the normal flow of data and RR packets is commonly large and less interesting to the user, so event reporting can significantly decrease the processor load induced by debug reporting.

Use the **dump** keyword to display the entire contents, including user data, of X.25 packets. The encoded X.25 packet contents are displayed after the standard packet description. The output includes the offset into the packet and the display of the data in both hexadecimal and ASCII formats.



Caution

The X.25 packet information that is reported by using the **dump** keyword may contain sensitive data; for example, clear-text account identities and passwords. The network access policies and router configuration should be controlled appropriately to address this risk.

Examples

The following is sample output from the **debug x25 interface** command:

```
Router# debug x25 interface serial 0

X.25 packet debugging is on
X.25 packet debugging is restricted to interface serial0

Serial0: X.25 I R/Inactive Restart (5) 8 lci 0
      Cause 7, Diag 0 (Network operational/No additional information)
Serial0: X.25 O R3 Restart Confirm (3) 8 lci 0
Serial0: X.25 I P1 Call (15) 8 lci 1
From(6): 170091 To(6): 170090
      Facilities: (0)
      Call User Data (4): 0xCC000000 (ip)
Serial0: X.25 O P3 Call Confirm (3) 8 lci 1
Serial0: X.25 I D1 Data (103) 8 lci 1 PS 0 PR 0
Serial0: X.25 O D1 Data (103) 8 lci 1 PS 0 PR 1
Serial0: X.25 I P4 Clear (5) 8 lci 1
      Cause 9, Diag 122 (Out of order/Maintenance action)
Serial0: X.25 O P7 Clear Confirm (3) 8 lci 1
```

Table 339 describes the significant fields shown in the display.

Table 339 debug x25 interface Field Descriptions

| Field | Description |
|---------|--|
| Serial0 | Interface on which the X.25 event occurred. |
| X.25 | Type of event this message describes. |
| I | Letter indicating whether the X.25 packet was input (I) or output (O) through the interface. |

Table 339 debug x25 interface Field Descriptions (continued)

| Field | Description |
|-------|--|
| R3 | <p>State of the service or virtual circuit (VC). Possible values follow:</p> <p>R/Inactive—Packet layer awaiting link layer service</p> <p>R1—Packet layer ready</p> <p>R2—Data terminal equipment (DTE) restart request</p> <p>R3—DCE restart indication</p> <p>P/Inactive—VC awaiting packet layer service</p> <p>P1—Idle</p> <p>P2—DTE waiting for DCE to connect CALL</p> <p>P3—DCE waiting for DTE to accept CALL</p> <p>P4—Data transfer</p> <p>P5—CALL collision</p> <p>P6—DTE clear request</p> <p>P7—DCE clear indication</p> <p>D/Inactive—VC awaiting setup</p> <p>D1—Flow control ready</p> <p>D2—DTE reset request</p> <p>D3—DCE reset indication</p> <p>Refer to Annex B of the <i>ITU-T Recommendation X.25</i> for more information on these states.</p> |

Table 339 debug x25 interface Field Descriptions (continued)

| Field | Description |
|---|---|
| Restart | <p>The type of X.25 packet. Possible values follow:</p> <p>R Events</p> <ul style="list-style-type: none"> Restart Restart Confirm Diagnostic <p>P Events</p> <ul style="list-style-type: none"> Call Call Confirm Clear Clear Confirm <p>D Events</p> <ul style="list-style-type: none"> Reset Reset Confirm <p>D1 Events</p> <ul style="list-style-type: none"> Data Receiver Not Ready (RNR) RR (Receiver Ready) Interrupt Interrupt Confirm <p>XOT Overhead</p> <ul style="list-style-type: none"> PVC Setup |
| (5) | Number of bytes in the packet. |
| 8 | Modulo of the virtual circuit. Possible values are 8 and 128. |
| lci 0 | VC number. Refer to Annex A of the <i>ITU-T Recommendation X.25</i> for information on VC assignment. |
| Cause 7 | Code indicating the event that triggered the packet. The Cause field can appear only in entries for Clear, Reset, and Restart packets. Possible values for the Cause field can vary, depending on the type of packet. Refer to the appendix “X.25 Cause and Diagnostic Codes” for an explanation of these codes. |
| Diag 0 | Code providing an additional hint of what, if anything, went wrong. The Diag field can appear only in entries for Clear, Diagnostic (as “error 0”), Reset, and Restart packets. Refer to the appendix “X.25 Cause and Diagnostic Codes” for an explanation of these codes. |
| (Network operational/ No additional information) | The standard explanations of the Cause and Diagnostic codes (<i>cause/diag</i>). |

Table 339 *debug x25 interface Field Descriptions (continued)*

| Field | Description |
|---------------------|---|
| From (6):170091 | Source address. (6) indicates the number of digits in the address that follows. The source address is part of the address block that may be encoded in Call Setup packets. |
| To (6): 170090 | Destination address. (6) indicates the number of digits in the address that follows. The destination address is part of the address block that may be encoded in Call Setup packets. |
| Facilities:(0) | Indicates that a facilities block is encoded and that it consists of 0 bytes. A breakdown of the encoded facilities (if any) follows. |
| Call User Data (4): | Indicates that the Call User Data (CUD) field is present and consists of 4 bytes. |
| 0xCC000000 (ip) | Protocol identifier (PID). This subfield of the CUD field is presented in the output as a hexadecimal string followed by the name of the protocol (in this case, IP) that the string represents. Any bytes following the PID are designated “user data” and may be used by an application separately from the PID. |

Related Commands

| Command | Description |
|----------------------|--|
| debug x25 | Displays information about all X.25 traffic or a specific X.25 service class. |
| debug x25 vc | Displays information about traffic for all virtual circuits that use a given number. |
| debug x25 xot | Displays information about traffic to or from a specific XOT host. |

debug x25 vc

To display information about traffic for all virtual circuits that have a given number, use the **debug x25 vc** command. To disable debugging output, use the **no** form of this command.

debug x25 vc *number* [**events** | **all**] [**dump**]

no debug x25 vc *number* [**events** | **all**] [**dump**]

Syntax Description

| | |
|---------------|---|
| <i>number</i> | Virtual circuit number. Range is from 1 to 4095. |
| events | (Optional) Displays all traffic except Data and Receiver Ready (RR) packets. |
| all | (Optional) Displays all traffic. This is the default. |
| dump | (Optional) Displays the encoded packet contents in hexadecimal and ASCII formats. |

Defaults

All traffic is displayed.

Command Modes

Privileged EXEC

Command History

| Release | Modification |
|----------|---|
| 10.0 | This command was introduced. |
| 12.3(2)T | The dump keyword was introduced. |

Usage Guidelines



Caution

The X.25 debug commands can generate large amounts of debugging output. If logging of debug output to the router console is enabled (the default condition), this output may fill the console buffer, preventing the router from processing packets until the contents of the console buffer have been printed.

The **debug x25**, **debug x25 interface**, **debug x25 vc**, and **debug x25 xot** commands all generate the same basic output. The **debug x25 interface**, **debug x25 vc**, and **debug x25 xot** commands narrow the scope of the **debug x25** output to specific virtual circuits or types of traffic.

Because no interface is specified by the **debug x25 vc** command, traffic on any virtual circuit that has the specified number is reported.

Virtual circuit (VC) zero (vc 0) cannot be specified. It is used for X.25 service messages, such as RESTART packets, not virtual circuit traffic. Service messages can be monitored only when no virtual circuit filter is used.

Use the **dump** keyword to display the entire contents, including user data, of X.25 packets. The encoded X.25 packet contents are displayed after the standard packet description. The output includes the offset into the packet and the display of the data in both hexadecimal and ASCII formats.

**Caution**

The X.25 packet information that is reported by using the **dump** keyword may contain sensitive data; for example, clear-text account identities and passwords. The network access policies and router configuration should be controlled appropriately to address this risk.

Examples

The following shows sample output from the **debug x25 vc** command:

```
Router# debug x25 vc 1 events
```

```
X.25 special event debugging is on
X.25 debug output restricted to VC number 1
```

```
Router# show debug
```

```
X.25 (filtered for VC 1):
  X.25 special event debugging is on
*Jun 18 20:22:29.735 UTC:Serial0:X.25 O R1 Call (13) 8 lci 1
*Jun 18 20:22:29.735 UTC:  From (4):2501 To (4):2502
*Jun 18 20:22:29.735 UTC:  Facilities:(0)
*Jun 18 20:22:29.735 UTC:  Call User Data (4):0xCC000000 (ip)
*Jun 18 20:22:29.739 UTC:Serial0:X.25 I R1 Call Confirm (3) 8 lci 1
*Jun 18 20:22:36.651 UTC:Serial0:X.25 O R1 Clear (5) 8 lci 1
*Jun 18 20:22:36.651 UTC:  Cause 9, Diag 122 (Out of order/Maintenance action)
*Jun 18 20:22:36.655 UTC:Serial0:X.25 I R1 Clear Confirm (3) 8 lci 1
```

[Table 340](#) describes significant fields shown in the display.

Table 340 *debug x25 vc Field Descriptions*

| Field | Description |
|---------|--|
| Serial0 | Interface on which the X.25 event occurred. |
| X.25 | Type of event this message describes. |
| O | Letter indicating whether the X.25 packet was input (I) or output (O) through the interface. |

Table 340 *debug x25 vc Field Descriptions (continued)*

| Field | Description |
|--------------|--|
| R1 | <p>State of the service or virtual circuit (VC). Possible values follow:</p> <p>R/Inactive—Packet layer awaiting link layer service</p> <p>R1—Packet layer ready</p> <p>R2—Data terminal equipment (DTE) restart request</p> <p>R3—DCE restart indication</p> <p>P/Inactive—VC awaiting packet layer service</p> <p>P1—Idle</p> <p>P2—DTE waiting for DCE to connect CALL</p> <p>P3—DCE waiting for DTE to accept CALL</p> <p>P4—Data transfer</p> <p>P5—CALL collision</p> <p>P6—DTE clear request</p> <p>P7—DCE clear indication</p> <p>D/Inactive—VC awaiting setup</p> <p>D1—Flow control ready</p> <p>D2—DTE reset request</p> <p>D3—DCE reset indication</p> <p>Refer to Annex B of the <i>ITU-T Recommendation X.25</i> for more information on these states.</p> |

Table 340 debug x25 vc Field Descriptions (continued)

| Field | Description |
|---------------------|--|
| Call | The type of X.25 packet. Possible values follow: R Events <ul style="list-style-type: none"> • Restart • Restart Confirm • Diagnostic P Events <ul style="list-style-type: none"> • Call • Call Confirm • Clear • Clear Confirm D Events <ul style="list-style-type: none"> • Reset • Reset Confirm D1 Events <ul style="list-style-type: none"> • Data • Receiver Not Ready (RNR) • RR (Receiver Ready) • Interrupt • Interrupt Confirm XOT Overhead <ul style="list-style-type: none"> • PVC Setup |
| (5) | Number of bytes in the packet. |
| 8 | Modulo of the virtual circuit. Possible values are 8 and 128. |
| lci 0 | VC number. Refer to Annex A of the <i>ITU-T Recommendation X.25</i> for information on VC assignment. |
| From (4):2501 | Source address. (4) indicates the number of digits in the address that follows. The source address is part of the address block that may be encoded in Call Setup packets. |
| To (4): 2502 | Destination address. (4) indicates the number of digits in the address that follows. The destination address is part of the address block that may be encoded in Call Setup packets. |
| Facilities:(0) | Indicates that 0 bytes are being used to encode facilities. |
| Call User Data (4): | Indicates that the Call User Data (CUD) field is present and consists of 4 bytes. |
| 0xCC000000 (ip) | Protocol identifier (PID). This subfield of the CUD field is presented in the output as a hexadecimal string followed by the name of the protocol (in this case, IP) that the string represents. Any bytes following the PID are designated “user data” and may be used by an application separately from the PID. |

Table 340 *debug x25 vc Field Descriptions (continued)*

| Field | Description |
|---|--|
| Cause 7 | Code indicating the event that triggered the packet. The Cause field can appear only in entries for Clear, Reset, and Restart packets. Possible values for the Cause field can vary, depending on the type of packet. Refer to the appendix “X.25 Cause and Diagnostic Codes” for an explanation of these codes. |
| Diag 0 | Code providing an additional hint of what, if anything, went wrong. The Diag field can appear only in entries for Clear, Diagnostic (as “error 0”), Reset, and Restart packets. Refer to the appendix “X.25 Cause and Diagnostic Codes” for an explanation of these codes. |
| (Network operational/ No additional information) | The standard explanations of the Cause and Diagnostic codes (<i>cause/diag</i>). |

Related Commands

| Command | Description |
|----------------------------|--|
| debug x25 | Displays information about all X.25 traffic or a specific X.25 service class. |
| debug x25 interface | Displays information about a specific X.25 or CMNS context or virtual circuit. |
| debug x25 xot | Displays information about traffic to or from a specific XOT host. |

debug x25 xot

To display information about traffic to or from a specific X.25 over TCP (XOT) host, use the **debug x25 xot** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

```
debug x25 xot [remote ip-address [port number]] [local ip-address [port number]] [events | all] [dump]
```

```
no debug x25 xot [remote ip-address [port number]] [local ip-address [port number]] [events | all] [dump]
```

Syntax Description

| | |
|--|--|
| remote <i>ip-address</i> [port number] | (Optional) Remote IP address and, optionally, a port number. Range is from 1 to 65535. |
| local <i>ip-address</i> [port number] | (Optional) Local host IP address and, optionally, a port number. Range is from 1 to 65535. |
| events | (Optional) Displays all traffic except Data and Receiver Ready (RR) packets. |
| all | (Optional) Displays all traffic. This is the default. |
| dump | (Optional) Displays the encoded packet contents in hexadecimal and ASCII formats. |

Defaults

All traffic is displayed.

Command Modes

Privileged EXEC

Command History

| Release | Modification |
|----------|------------------------------------|
| 10.0 | This command was introduced. |
| 12.3(2)T | The dump keyword was added. |

Usage Guidelines



Caution

The X.25 debug commands can generate large amounts of debugging output. If logging of debug output to the router console is enabled (the default condition), this output may fill the console buffer, preventing the router from processing packets until the contents of the console buffer have been printed.

The **debug x25**, **debug x25 interface**, **debug x25 vc**, and **debug x25 xot** commands all generate the same basic output. The **debug x25 interface**, **debug x25 vc**, and **debug x25 xot** commands narrow the scope of the **debug x25** output to specific virtual circuits or types of traffic.

The **debug x25 xot** output allows you to restrict the debug output reporting to XOT traffic for one or both hosts or host/port combinations. Because each XOT virtual circuit uses a unique TCP connection, an XOT debug request that specifies both host addresses and ports will report traffic only for that virtual circuit. Also, you can restrict reporting to sessions initiated by the local or remote router by specifying 1998 for the remote or local port. (XOT connections are received on port 1998.)

Use the **dump** keyword to display the entire contents, including user data, of X.25 packets. The encoded X.25 packet contents are displayed after the standard packet description. The output includes the offset into the packet and the display of the data in both hexadecimal and ASCII formats.



Caution

The X.25 packet information that is reported by using the **dump** keyword may contain sensitive data; for example, clear-text account identities and passwords. The network access policies and router configuration should be controlled appropriately to address this risk.

Examples

The following shows sample output from the **debug x25 xot** command:

```
Router# debug x25 xot
```

```
X.25 packet debugging is on
X.25 debug output restricted to protocol XOT
```

```
Router# show debug
```

```
X.25 (filtered for XOT):
  X.25 packet debugging is on
*Jun 18 20:32:34.699 UTC: [10.0.155.71,11001/10.0.155.70,1998]:XOT I P/Inactive Call (19) 8
lci 1
*Jun 18 20:32:34.699 UTC:   From (4):2501 To (4):2502
*Jun 18 20:32:34.699 UTC:   Facilities:(6)
*Jun 18 20:32:34.699 UTC:   Packet sizes:128 128
*Jun 18 20:32:34.699 UTC:   Window sizes:2 2
*Jun 18 20:32:34.699 UTC:   Call User Data (4):0xCC000000 (ip)
*Jun 18 20:32:34.707 UTC: [10.0.155.71,11001/10.0.155.70,1998]:XOT O P3 Call Confirm (11) 8
lci 1
*Jun 18 20:32:34.707 UTC:   From (0): To (0):
*Jun 18 20:32:34.707 UTC:   Facilities:(6)
*Jun 18 20:32:34.707 UTC:   Packet sizes:128 128
*Jun 18 20:32:34.707 UTC:   Window sizes:2 2
*Jun 18 20:32:34.715 UTC: [10.0.155.71,11001/10.0.155.70,1998]:XOT I D1 Data (103) 8 lci 1
PS 0 PR 0
*Jun 18 20:32:34.723 UTC: [10.0.155.71,11001/10.0.155.70,1998]:XOT O D1 Data (103) 8 lci 1
PS 0 PR 1
*Jun 18 20:32:34.731 UTC: [10.0.155.71,11001/10.0.155.70,1998]:XOT I D1 Data (103) 8 lci 1
PS 1 PR 1
*Jun 18 20:32:34.739 UTC: [10.0.155.71,11001/10.0.155.70,1998]:XOT O D1 Data (103) 8 lci 1
PS 1 PR 2
*Jun 18 20:32:34.747 UTC: [10.0.155.71,11001/10.0.155.70,1998]:XOT I D1 Data (103) 8 lci 1
PS 2 PR 2
*Jun 18 20:32:34.755 UTC: [10.0.155.71,11001/10.0.155.70,1998]:XOT O D1 Data (103) 8 lci 1
PS 2 PR 3
*Jun 18 20:32:34.763 UTC: [10.0.155.71,11001/10.0.155.70,1998]:XOT I D1 Data (103) 8 lci 1
PS 3 PR 3
*Jun 18 20:32:34.771 UTC: [10.0.155.71,11001/10.0.155.70,1998]:XOT O D1 Data (103) 8 lci 1
PS 3 PR 4
*Jun 18 20:32:34.779 UTC: [10.0.155.71,11001/10.0.155.70,1998]:XOT I D1 Data (103) 8 lci 1
PS 4 PR 4
*Jun 18 20:32:34.787 UTC: [10.0.155.71,11001/10.0.155.70,1998]:XOT O D1 Data (103) 8 lci 1
PS 4 PR 5
```

Table 341 describes the significant fields shown in the display.

Table 341 debug x25 xot Field Descriptions

| Field | Description |
|--------------------------------------|---|
| [10.0.155.71,11001/10.0.155.70,1998] | TCP connection identified by the remote IP address, remote TCP port/local IP address, local TCP port. An XOT connection is always placed to port ID 1998, so a remote port ID of 1998 implies that the router initiated the TCP connection, whereas a local port ID of 1998 implies that the router received the TCP connection. |
| XOT | Type of event this message describes. |
| I | Letter indicating whether the X.25 packet was input (I) or output (O) through the interface. |
| P/Inactive | State of the service or virtual circuit (VC). Possible values follow: R/Inactive—Packet layer awaiting link layer service R1—Packet layer ready R2—Data terminal equipment (DTE) restart request R3—DCE restart indication P/Inactive—VC awaiting packet layer service P1—Idle P2—DTE waiting for DCE to connect CALL P3—DCE waiting for DTE to accept CALL P4—Data transfer P5—CALL collision P6—DTE clear request P7—DCE clear indication D/Inactive—VC awaiting setup D1—Flow control ready D2—DTE reset request D3—DCE reset indication Refer to Annex B of the <i>ITU-T Recommendation X.25</i> for more information on these states. |

Table 341 debug x25 xot Field Descriptions (continued)

| Field | Description |
|----------------|--|
| Call | The type of X.25 packet. Possible values follow: R Events <ul style="list-style-type: none"> • Restart • Restart Confirm • Diagnostic P Events <ul style="list-style-type: none"> • Call • Call Confirm • Clear • Clear Confirm D Events <ul style="list-style-type: none"> • Reset • Reset Confirm D1 Events <ul style="list-style-type: none"> • Data • Receiver Not Ready (RNR) • RR (Receiver Ready) • Interrupt • Interrupt Confirm XOT Overhead <ul style="list-style-type: none"> • PVC Setup |
| (19) | Number of bytes in the packet. |
| 8 | Modulo of the virtual circuit. Possible values are 8 and 128. |
| lci 1 | VC number. Refer to Annex A of the <i>ITU-T Recommendation X.25</i> for information on VC assignment. |
| From (4):2501 | Source address. (4) indicates the number of digits in the address that follows. The source address is part of the address block that may be encoded in Call Setup packets. |
| To (4): 2502 | Destination address. (4) indicates the number of digits in the address that follows. The destination address is part of the address block that may be encoded in Call Setup packets. |
| Facilities:(6) | Indicates that a facilities block is encoded and that it consists of 6 bytes. A breakdown of the encoded facilities follows. |

Table 341 debug x25 xot Field Descriptions (continued)

| Field | Description |
|---|---|
| Packet sizes | Encoded packet size facility settings. |
| Window sizes | Encoded window size facility settings. |
| Call User Data (4): | Indicates that the Call User Data (CUD) field is present and consists of 4 bytes. |
| 0xCC000000 (ip) | Protocol identifier (PID). This subfield of the CUD field is presented in the output as a hexadecimal string followed by the name of the protocol (in this case, IP) that the string represents. Any bytes following the PID are designated “user data” and may be used by an application separately from the PID. |
| Cause 7 | Code indicating the event that triggered the packet. The Cause field can appear only in entries for Clear, Reset, and Restart packets. Possible values for the Cause field can vary, depending on the type of packet. Refer to the appendix “X.25 Cause and Diagnostic Codes” for an explanation of these codes. |
| Diag 0 | Code providing an additional hint of what, if anything, went wrong. The Diag field can appear only in entries for Clear, Diagnostic (as “error 0”), Reset, and Restart packets. Refer to the appendix “X.25 Cause and Diagnostic Codes” for an explanation of these codes. |
| (Network operational/ No additional information) | The standard explanations of the Cause and Diagnostic codes (<i>cause/diag</i>). |

Related Commands

| Command | Description |
|----------------------------|--|
| debug x25 | Displays information about all X.25 traffic or a specific X.25 service class. |
| debug x25 interface | Displays information about a specific X.25 or CMNS context or virtual circuit. |
| debug x25 vc | Displays information about traffic for all virtual circuits that use a given number. |

debug x28

To monitor error information and X.28 connection activity, use the **debug x28** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug x28

no debug x28

Syntax Description This command has no arguments or keywords.

Command Modes Privileged EXEC

Examples The following is sample output while the packet assembler/disassembler (PAD) initiates an X.28 outgoing call:

```
Router# debug x28

X28 MODE debugging is on
Router# x28

*
03:30:43: X.28 mode session started
03:30:43: X28 escape is exit
03:30:43: Speed for console & vty lines :9600
*call 123456
COM
03:39:04: address ="123456", cud="[none]" 03:39:04: Setting X.3 Parameters for this
call...1:1 2:1 3:126 4:0 5:1 6:2 7:2 8:0 9:0 10:0 11:14 12:1 13:0 14:0 15:0 16:127 17:24
18:18 19:2 20:0 21:0 22:0

Router> exit
CLR CONF

*
*03:40:50: Session ended
* exit

Router#
*03:40:51: Exiting X.28 mode
```

debug xcctsp all

To debug External Call Control Telephony Service Provider (TSP) information, use the **debug xcctsp all** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug xcctsp all

no debug xcctsp all

Syntax Description This command has no arguments or keywords.

Command Modes Privileged EXEC

| Command History | Release | Modification |
|-----------------|----------|--|
| | 12.0(5)T | This command was introduced. |
| | 12.0(7)T | Support for this command was extended to the Cisco uBR924 cable modem. |

Examples See the following examples to turn on and off external call control debugging:

```
AS5300-TGW# debug xcctsp all
```

```
External call control all debugging is on
```

```
AS5300-TGW# no debug xcctsp all
```

```
External call control all debugging is off
```

```
AS5300-TGW#
```

| Related Commands | Command | Description |
|------------------|-----------------------------|--|
| | debug xcctsp error | Enables debugging on external call control errors. |
| | debug xcctsp session | Enables debugging on external call control sessions. |

debug xcctsp error

To debug External Call Control Telephony Service Provider (TSP) error information, use the **debug xcctsp error** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug xcctsp error

no debug xcctsp error

Syntax Description This command has no arguments or keywords.

Command Modes Privileged EXEC

| Command History | Release | Modification |
|-----------------|----------|--|
| | 12.0(5)T | This command was introduced. |
| | 12.0(7)T | Support for this command was integrated on the Cisco uBR924 cable modem. |

Examples See the following examples to turn on and off error-level debugging:

```
AS5300-TGW# debug xcctsp error
External call control error debugging is on
AS5300-TGW# no debug xcctsp error
External call control error debugging is off
```

| Related Commands | Command | Description |
|------------------|-----------------------------|--|
| | debug xcctsp all | Enables debugging on all external call control levels. |
| | debug xcctsp session | Enables debugging on external call control sessions. |

debug xcctsp session

To debug External Call Control Telephony Service Provider (TSP) session information, use the **debug xcctsp session** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug xcctsp session

no debug xcctsp session

Syntax Description This command has no arguments or keywords.

Command Modes Privileged EXEC

| Command History | Release | Modification |
|-----------------|----------|--|
| | 12.0(5)T | This command was introduced. |
| | 12.0(7)T | Support for this command was integrated on the Cisco uBR924 cable modem. |

Examples See the following examples to turn on and off session-level debugging:

```
AS5300-TGW# debug xcctsp session
```

```
External call control session debugging is on
```

```
AS5300-TGW# no debug xcctsp session
```

```
External call control session debugging is off
```

```
AS5300-TGW#
```

| Related Commands | Command | Description |
|------------------|---------------------------|--|
| | debug xcctsp all | Enables debugging on external call control levels. |
| | debug xcctsp error | Enables debugging on external call control errors. |

debug xconnect

To debug a problem related to the Xconnect configuration, use the **debug xconnect** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug xconnect { error | event }

no debug xconnect { error | event }

Syntax Description

| | |
|--------------|--|
| error | Displays errors related to an Xconnect configuration. |
| event | Displays events related to an Xconnect configuration processing. |

Command Modes

Privileged EXEC

Command History

| Release | Modification |
|-----------|--|
| 12.0(23)S | This command was introduced. |
| 12.3(2)T | This command was integrated into Cisco IOS Release 12.3(2)T. |

Usage Guidelines

Use this command to display debugging information about Xconnect sessions.

Examples

The following shows sample output from the **debug xconnect** command for an Xconnect session on an Ethernet interface:

```
Router# debug xconnect

00:01:16: XC AUTH [Et2/1, 5]: Event: start xconnect authorization, state changed from IDLE
to AUTHORIZING
00:01:16: XC AUTH [Et2/1, 5]: Event: found xconnect authorization, state changed from
AUTHORIZING to DONE
00:01:16: XC AUTH [Et2/1, 5]: Event: free xconnect authorization request, state changed
from DONE to END
```

Related Commands

| Command | Description |
|-----------------------|---|
| debug acircuit | Displays events and failures related to attachment circuits. |
| debug vpdn | Displays errors and events relating to L2TP configuration and the surrounding Layer 2 tunneling infrastructure. |

debug xcsp

To display the debugging messages for the External Control Service Provider (XCSP) subsystem, use the **debug xcsp** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

```
debug xcsp {all | cot | event}
```

```
no debug xcsp {all | cot | event}
```

Syntax Description

| | |
|--------------|---|
| all | Provides debug information about XCSP events and continuity testing (COT). |
| cot | Provides debug information about XCSP and COT. The cot keyword is not used with the NAS Package for Media Gateway Control Protocol (MGCP) feature. |
| event | Provides debug information about XCSP events. |

Defaults

No default behavior or values

Command Modes

Privileged EXEC

Command History

| Release | Modification |
|-----------|---|
| 12.2(2)XB | This command was introduced. |
| 12.2(11)T | The command was integrated into Cisco IOS Release 12.2(11)T for the Cisco AS5850. |

Usage Guidelines

This command is used with the Network Access Server Package for MGCP. The XCSP subsystem is not configured directly, but information about it may be useful in troubleshooting. The **debug xcsp** command is used to display the exchange of signaling information between the MGCP protocol stack and end applications such as call switching module (CSM) or dialer.

The **cot** keyword is not used with the Network Access Server Package for MGCP feature.

Examples

The following shows sample output from the **debug xcsp all** command and keyword and the **debug xcsp event** command and keyword:

```
Router# debug xcsp all

xcsp all debugging is on

Router# debug xcsp event

xcsp events debugging is on

01:49:14:xcsp_call_msg:Event Call Indication , channel state = Idle for
```

```

slot port channel 7
c5400# 0 23
01:49:14:xcsp_process_sig_fsm:state/event Idle / Call Indication
01:49:14:xcsp_incall:
01:49:14:xcsp_incall CONNECT_IND:cdn=3000 cgn=1000
01:49:14:xcsp:START guard TIMER
01:49:14:xcsp_fsm:slot 7 port 0 chan 23 oldstate = Idle newstate= Connection
in progress mgcpapp_process_mgcp_msg PROCESSED NAS PACKAGE EVENT

01:49:14:Received message on XCSP_CDAPI
01:49:14:process_cdapi_msg :slot/port/channel 7/0/23
01:49:14: process_cdapi_msg:new slot/port/channel 7/0/23
01:49:14:
c5400#Received CONN_RESP:callid=0x7016
01:49:14:process_cdapi:Event CONN_RESP, channel state = 8 for slot port
channel 7 0 23
01:49:14:xcsp_process_sig_fsm:state/event Connection in progress / In Call
accept
mgcpapp_xcsp_alert:
mgcpapp_xcsp_get_chan_cb -Found - Channel state Connection in progress

200 58 Alert
I:630AED90
<---:Ack send SUCCESSFUL

01:49:14:xcsp_fsm:slot 7 p
c5400#ort 0 chan 23 oldstate = Connection in progress newstate= Connection in
progress
01:49:14:Received message on XCSP_CDAPI
01:49:14:process_cdapi_msg :slot/port/channel 7/0/23
01:49:14: process_cdapi_msg:new slot/port/channel 7/0/23
01:49:14: Received CALL_CONN:callid=0x7016
01:49:14:process_cdapi:Event CONN_, channel state = 8 for slot port channel 7
0 23
01:49:14:xcsp_process_sig_fsm:state/event Connection in progress / in call
connect
mgcpapp_xcsp_connect:
mgcpapp_xc
c5400#sp_get_chan_cb -Found - Channel state In Use

01:49:14:STOP TIMER
01:49:14:xcsp_fsm:slot 7 port 0 chan 23 oldstate = Connection in progress
newstate=In Use
c5400#
01:50:23:Received message on XCSP_CDAPI
01:50:23:process_cdapi_msg :slot/port/channel 7/0/23
01:50:23: process_cdapi_msg:new slot/port/channel 7/0/23
01:50:23: Received CALL_DISC_REQ:callid=0x7016
01:50:23:process_cdapi:Event DISC_CONN_REQ, channel state = 7 for slot port
channel 7 0 23
01:50:23:xcsp_process_sig_fsm:state/event In Use / release Request
mgcpapp_xcsp_disconnect
mgcpapp_xcsp_get_chan_cb -Fou
c5400#nd - Channel state In Use
01:50:23:send_mgcp_msg, MGCP Packet sent --->

01:50:23:RSIP 1 *@c5400 MGCP 1.0
RM:restart

DLCX 4 S7/DS1-0/23 MGCP 1.0
C:3
I:630AED90
E:801 /NAS User request

```

```

01:50:23:xcsp_fsm:slot 7 port 0 chan 23 oldstate = In Use  newstate=Out
Release in progress
xcsp_restart Serial7/0:22 vc = 22
xcsp_restart Put idb Serial7/0:22 in down state
01:50:23:MGCP Packet received -
200 4 bye

Data call ack received callp=0x62AEEA70mgcpapp_xcsp
c5400#_ack_recv:mgcpapp_xcsp_get_chan_cb -Found - Channel state Out Release in
progress

mgcpapp_xcsp_ack_recv ACK 200 rcvd:transaction id = 4 endpt=S7/DS1-0/23
01:50:23:xcsp_call_msg:Event Release confirm , channel state = Out Release in
progress for slot port channel 7 0 23
01:50:23:xcsp_process_sig_fsm:state/event Out Release in progress/ Release
confirm
01:50:23:STOP TIMER
01:50:23:xcsp_fsm:slot 7 port 0 chan 23 oldstate = Out Release in progress
newstate= Idle

```

Related Commands

| Command | Description |
|--------------------------|---|
| show vrm vdevices | Displays the status of a router port under the control of the XCSP subsystem. |
| show xcsp slot | Displays the status of a router slot under the control of the XCSP subsystem. |

debug xdsl application

To monitor the xDSL if the digital subscriber line (DSL) does not come up, use the **debug xdsl application** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug xdsl application

no debug xdsl application

Syntax Description This command has no arguments or keywords.

Command Modes Privileged EXEC

| Command History | Release | Modification |
|-----------------|-----------|--|
| | 12.3(4)XD | This command was introduced on Cisco 2600 series and Cisco 3700 series routers. |
| | 12.3(4)XG | This command was integrated into the Cisco IOS Release 12.3(4)XG on the Cisco 1700 series routers. |
| | 12.3(7)T | This command was integrated into Cisco IOS Release 12.3(7)T on Cisco 2600 series, Cisco 3631, and Cisco 3700 series routers. |
| | 12.3(11)T | This command was integrated into Cisco IOS Release 12.3(11)T on Cisco 2800 series and Cisco 3800 series routers. |

Usage Guidelines The **debug xdsl application** command details what occurs during the Cisco IOS SHDSL process events and signal-to-noise ratio sampling of the SHDSL chip. This information can be used more for software debugging in analyzing the internal events.

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

```
Router# debug xdsl application

xDSL application debugging is on
Router#
```

The following lines show that the application is starting on the router and waiting for a response:

```
00:47:40: DSL 0/0 process_get_wakeup
00:47:41: DSL 0/0 process_get_wakeup
00:47:42: DSL 0/0 process_get_wakeup
00:47:43: DSL 0/0 process_get_wakeup
00:47:44: DSL 0/0 process_get_wakeup
00:47:45: DSL 0/0 process_get_wakeup
00:47:46: DSL 0/0 process_get_wakeup
00:47:47: DSL 0/0 process_get_wakeup
00:47:48: DSL 0/0 process_get_wakeup
00:47:49: DSL 0/0 process_get_wakeup
00:47:49: DSL 0/0 process_get_wakeup
```

The following lines show that the controller link comes up:

```
00:47:49: DSL 0/0 xdsl_background_process: XDSL link up boolean event received
00:47:49: DSL 0/0 controller Link up! line rate: 1600 Kbps
```

The following lines show that the DSL controller comes up:

```
00:47:49: DSL 0/0 xdsl_controller_reset: cdb-state=up
00:47:49: %CONTROLLER-5-UPDOWN: Controller DSL 0/0, changed state to up
00:47:49: Dslsar data rate 1600
00:47:49: DSL 0/0 TipRing 1, Xmit_Power Val 75, xmit_power 7.5
00:47:49: DSL 0/0 Mode 2, BW 1600, power_base_value 135, power_backoff 6
00:47:50: DSL 0/0 process_get_wakeup
00:47:51: DSL 0/0 process_get_wakeup
00:47:52: DSL 0/0 process_get_wakeup
00:47:53: DSL 0/0 process_get_wakeup
00:47:54: DSL 0/0 process_get_wakeup
00:47:55: DSL 0/0 process_get_wakeup
00:47:56: DSL 0/0 process_get_wakeup
```

The following lines show signal-to-noise ratio sampling:

```
00:47:56: DSL 0/0 SNR Sampling: 42 dB
00:47:57: DSL 0/0 process_get_wakeup
00:47:57: DSL 0/0 SNR Sampling: 41 dB
00:47:58: DSL 0/0 process_get_wakeup
00:47:58: DSL 0/0 SNR Sampling: 40 dB
00:47:59: DSL 0/0 process_get_wakeup
00:47:59: DSL 0/0 SNR Sampling: 40 dB
00:48:00: DSL 0/0 process_get_wakeup
00:48:00: DSL 0/0 SNR Sampling: 39 dB
00:48:01: DSL 0/0 process_get_wakeup
00:48:01: DSL 0/0 SNR Sampling: 39 dB
00:48:02: DSL 0/0 process_get_wakeup
00:48:02: DSL 0/0 SNR Sampling: 38 dB
00:48:03: DSL 0/0 process_get_wakeup
00:48:03: DSL 0/0 SNR Sampling: 38 dB
00:48:04: DSL 0/0 process_get_wakeup
00:48:04: DSL 0/0 SNR Sampling: 38 dB
00:48:05: DSL 0/0 process_get_wakeup
00:48:05: DSL 0/0 SNR Sampling: 37 dB
00:48:06: DSL 0/0 process_get_wakeup
00:48:06: DSL 0/0 SNR Sampling: 37 dB
00:48:07: DSL 0/0 process_get_wakeup
00:48:07: DSL 0/0 SNR Sampling: 36 dB
```

The following lines show that the link comes up:

```
00:48:07: %LINK-3-UPDOWN: Interface ATM0/0, changed state to up
00:48:08: DSL 0/0 process_get_wakeup
00:48:08: DSL 0/0 SNR Sampling: 36 dB
```

The following lines show that the line protocol comes up:

```
00:48:08: %LINEPROTO-5-UPDOWN: Line protocol on Interface ATM0/0, changed state to up
00:48:09: DSL 0/0 process_get_wakeup
00:48:09: DSL 0/0 SNR Sampling: 36 dB
00:48:10: DSL 0/0 process_get_wakeup
00:48:10: DSL 0/0 SNR Sampling: 36 dB
00:48:11: DSL 0/0 process_get_wakeup
00:48:11: DSL 0/0 SNR Sampling: 35 dB
00:48:12: DSL 0/0 process_get_wakeup
00:48:12: DSL 0/0 SNR Sampling: 36 dB
00:48:13: DSL 0/0 process_get_wakeup
```

```
00:48:13: DSL 0/0 SNR Sampling: 36 dB
00:48:14: DSL 0/0 process_get_wakeup
00:48:14: DSL 0/0 SNR Sampling: 36 dB
00:48:15: DSL 0/0 process_get_wakeup
00:48:15: DSL 0/0 SNR Sampling: 36 dB
00:48:16: DSL 0/0 process_get_wakeup
00:48:16: DSL 0/0 SNR Sampling: 36 dB
00:48:17: DSL 0/0 process_get_wakeup
00:48:17: DSL 0/0 SNR Sampling: 35 dB
00:48:18: DSL 0/0 process_get_wakeup
00:48:18: DSL 0/0 SNR Sampling: 35 dB
00:48:19: DSL 0/0 process_get_wakeup
```

Related Commands

| Command | Description |
|--------------------------|---|
| debug xdsl driver | Monitors what is happening when downloading and installing the drivers. |
| debug xdsl eoc | Monitors what is in the embedded operations channel messages. |
| debug xdsl error | Monitors the errors of the xDSL process and firmware. |

debug xdsl driver

To display what is happening when the drivers are downloaded and installed, use the **debug xdsl driver** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug xdsl driver

no debug xdsl driver

Syntax Description This command has no arguments or keywords.

Command Modes Privileged EXEC

| Command History | Release | Modification |
|-----------------|-----------|--|
| | 12.3(4)XD | This command was introduced on Cisco 2600 series and Cisco 3700 series routers. |
| | 12.3(4)XG | This command was integrated into the Cisco IOS Release 12.3(4)XG on the Cisco 1700 series routers. |
| | 12.3(7)T | This command was integrated into Cisco IOS Release 12.3(7)T on Cisco 2600 series, Cisco 3631, and Cisco 3700 series routers. |
| | 12.3(11)T | This command was integrated into Cisco IOS Release 12.3(11)T on Cisco 2800 series and Cisco 3800 series routers. |

Usage Guidelines Use the **debug xdsl driver** command to monitor what is happening when downloading the firmware. This debugging command displays the Globespan DSL Driver details and provides framer interrupt information and line training failure information. This information can help you understand the problems faced while downloading the firmware, why the line went down, and so forth.

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

```
Router# debug xdsl driver
```

```
xDSL driver debugging is on
```

The following lines show that the DSP interrupt download is running:

```
*Mar 12 08:01:04.772: DSL 0/2 dsp interrupt-download next block for line-0
*Mar 12 08:01:04.780: DSL 0/2 framer intr_status 0xC0
*Mar 12 08:01:05.072: DSL 0/2 dsp interrupt-download next block for line-0
*Mar 12 08:01:05.080: DSL 0/2 framer intr_status 0xC0
*Mar 12 08:01:06.484: DSL 0/2 dsp interrupt-download next block for line-0
*Mar 12 08:01:06.492: DSL 0/2 framer intr_status 0xC0
*Mar 12 08:01:08.092: DSL 0/2 dsp interrupt-download next block for line-0
*Mar 12 08:01:08.096: DSL 0/2 framer intr_status 0xC0
*Mar 12 08:01:19.180: DSL 0/2 dsp interrupt-download next block for line-0
*Mar 12 08:01:19.184: DSL 0/2 framer intr_status 0xC0
*Mar 12 08:01:19.480: DSL 0/2 dsp interrupt-download next block for line-0
```

```
*Mar 12 08:01:19.484: DSL 0/2 framer intr_status 0xC0
*Mar 12 08:01:19.680: DSL 0/2 dsp interrupt-download next block for line-0
```

The following lines show that the DSP interrupt has been disabled and that the framer interrupt has been enabled:

```
*Mar 12 08:01:19.680: DSL 0/2 DSP interrupt disabled
*Mar 12 08:01:19.680: DSL 0/2 Download completed for line-0
*Mar 12 08:01:19.680: DSL 0/2 Framer interrupt enabled
*Mar 12 08:01:19.680: DSL 0/2 framer intr_status 0xC0
*Mar 12 08:01:19.680: DSL 0/2 controller Link up! line rate: 2304 Kbps
```

The following lines show that the digital subscriber line (DSL) controller has come up on slot 0 and port 2:

```
*Mar 12 08:01:19.680: %CONTROLLER-5-UPDOWN: Controller DSL 0/2, changed state to up
*Mar 12 08:01:19.680: Dsisar data rate 2304
*Mar 12 08:01:22.528: %LINK-3-UPDOWN: Interface ATM0/2, changed state to up
*Mar 12 08:01:23.528: %LINEPROTO-5-UPDOWN: Line protocol on Interface ATM0/2, changed state to up
```

The following lines show that the framer interrupt status is running:

```
*Mar 12 08:01:23.812: DSL 0/2 framer intr_status 0xC4
*Mar 12 08:01:23.816: DSL 0/2 framer intr_status 0xC4
*Mar 12 08:01:23.904: DSL 0/2 framer intr_status 0xC1
*Mar 12 08:01:28.612: DSL 0/2 framer intr_status 0xC4
*Mar 12 08:01:28.616: DSL 0/2 framer intr_status 0xC4
*Mar 12 08:01:28.708: DSL 0/2 framer intr_status 0xC1
*Mar 12 08:01:28.804: DSL 0/2 framer intr_status 0xC1
*Mar 12 08:01:33.412: DSL 0/2 framer intr_status 0xC4
*Mar 12 08:01:33.420: DSL 0/2 framer intr_status 0xC4
*Mar 12 08:01:33.508: DSL 0/2 framer intr_status 0xC1
*Mar 12 08:01:33.604: DSL 0/2 framer intr_status 0xC1
*Mar 12 08:01:33.700: DSL 0/2 framer intr_status 0xC1
*Mar 12 08:01:38.212: DSL 0/2 framer intr_status 0xC4
*Mar 12 08:01:38.220: DSL 0/2 framer intr_status 0xC4
*Mar 12 08:01:38.308: DSL 0/2 framer intr_status 0xC1
```

Related Commands

| Command | Description |
|-------------------------------|---|
| debug xdsl application | Monitors the xDSL if the DSL does not come up. |
| debug xdsl eoc | Monitors what is in the embedded operations channel messages. |
| debug xdsl error | Monitors the errors of the xDSL process and firmware. |

debug xdsl eoc

To display the flow of the embedded operations channel (EOC) messages received, processed, and transmitted, use the **debug xdsl eoc** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

```
debug xdsl eoc
```

```
no debug xdsl eoc
```

Syntax Description This command has no arguments or keywords.

Command Modes Privileged EXEC

| Command History | Release | Modification |
|-----------------|-----------|--|
| | 12.3(4)XD | This command was introduced on Cisco 2600 series and Cisco 3700 series routers. |
| | 12.3(4)XG | This command was integrated into the Cisco IOS Release 12.3(4)XG on the Cisco 1700 series routers. |
| | 12.3(7)T | This command was integrated into Cisco IOS Release 12.3(7)T on Cisco 2600 series, Cisco 3631, and Cisco 3700 series routers. |
| | 12.3(11)T | This command was integrated into Cisco IOS Release 12.3(11)T on Cisco 2800 series and Cisco 3800 series routers. |

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

```
Router# debug xdsl eoc
```

```
xDSL EOC debugging is on
Router#
```

The following lines show the embedded operations channel message being received and copied to the buffer. The `xdsl_background_process` is performed. The `data_transparency_remove` is performed.

```
00:02:55: Incoming EOC received
00:02:55: Copy the EOC to buffer
00:02:55: Incoming EOC received
00:02:55: Copy the EOC to buffer
00:02:55: End of EOC received, Notify task
00:02:55: xdsl_background_process:
00:02:55: Rx EOC remove transparency:: 12 C A 63
00:02:55: data_transparency_remove: Done, eoc packet size = 4
```

The following lines show that the packet of the embedded operations channel messages was received and verified as good. The data_transparency_add is performed.

```
00:02:55: Good eoc packet received
00:02:55: incoming request eocmsgid: 12
00:02:55: Tx Converted EOC message:: 21 8C 0 28 0 0 0 0 0 0 0 1 1 713
00:02:55: data_transparency_add: eoc packet size - before 15, after 15
```

The following lines show another embedded operations channel message coming in and copied to the buffer. The xdsl_background_process is run on this message as before.

```
00:02:55: size of eoc status response :: 13
00:02:56: Incoming EOC received
00:02:56: Copy the EOC to buffer
00:02:56: Incoming EOC received
00:02:56: Copy the EOC to buffer
00:02:56: End of EOC received, Notify task
00:02:56: xdsl_background_process:
00:02:56: Rx EOC remove transparency:: 12 C A 63
00:02:56: data_transparency_remove: Done, eoc packet size = 4
```

Related Commands

| Command | Description |
|-------------------------------|---|
| debug xdsl application | Monitors the xDSL if the DSL does not come up. |
| debug xdsl driver | Monitors what is happening when downloading and installing the drivers. |
| debug xdsl error | Monitors the errors of the xDSL process and firmware. |

debug xdsl error

To display the errors of xDSL process and firmware, use the **debug xdsl error** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug xdsl error

no debug xdsl error

Syntax Description This command has no arguments or keywords.

Command Modes Privileged EXEC

| Command History | Release | Modification |
|-----------------|-----------|--|
| | 12.3(4)XD | This command was introduced on Cisco 2600 series and Cisco 3700 series routers. |
| | 12.3(4)XG | This command was integrated into the Cisco IOS Release 12.3(4)XG on the Cisco 1700 series routers. |
| | 12.3(7)T | This command was integrated into Cisco IOS Release 12.3(7)T on Cisco 2600 series, Cisco 3631, and Cisco 3700 series routers. |
| | 12.3(11)T | This command was integrated into Cisco IOS Release 12.3(11)T on Cisco 2800 series and Cisco 3800 series routers. |

Usage Guidelines Use the **debug xdsl error** command to display the errors during driver initialization and any Globespan firmware API failures.

Examples The following is sample output from the **debug xdsl error** command. When the debug is enabled, a message indicates that DSL error debugging is on.

```
Router# debug xdsl error

xDSL error debugging is on
Router#
```

| Related Commands | Command | Description |
|------------------|-------------------------------|---|
| | debug xdsl application | Monitors the xDSL if the DSL does not come up. |
| | debug xdsl driver | Monitors what is happening when downloading and installing the drivers. |
| | debug xdsl eoc | Monitors what is in the embedded operations channel messages. |

voice call debug

To debug a voice call, use the **voice call debug** command in global configuration mode. To display a full globally unique identifier (GUID) or header as explained in the “Usage Guidelines” section, use the **no** form of this command.

voice call debug full-guid | short-header

no voice call debug full-guid | short-header

Syntax Description

| | |
|---------------------|--|
| full-guid | Displays the GUID in a 16-byte header. Note When the no version of this command is input with the full-guid keyword, the short 6-byte version is displayed. This is the default. |
| short-header | Displays the CallEntry ID in the header without displaying the GUID or module-specific parameters. |

Defaults

The short 6-byte header is displayed.

Command Modes

Global configuration

Command History

| Release | Modification |
|-----------|--|
| 12.2(11)T | The new debug header was added to the following Cisco routers: Cisco 2600 series, Cisco 3620, and Cisco 3640. and Cisco 3660; on the following universal gateways: Cisco AS5350, Cisco AS5400, and Cisco AS5850; on the following universal access servers: Cisco AS5300, and Cisco AS5800; and on the Cisco MC3810 multiservice access concentrators. |
| 12.2(15)T | The header-only keyword was removed and the short-header keyword was added. |

Usage Guidelines

The user can control the contents of the standardized header. The display options for the header are as follows:

- Short 6-byte GUID
- Full 16-byte GUID
- Short header which contains only the CallEntry ID

The format of the GUID headers is as follows:

//CallEntryID/GUID/Module-Dependent-List/Function-name:.

The format of the short header is as follows:

//CallEntryID/Function-name:.

When the **voice call debug short-header** command is entered, the header is displayed with no GUID or module-specific parameters. When the **no voice call debug short-header** command is entered, the header, the 6-byte GUID, and module-dependent parameter output are displayed. The default option is to display the 6-byte GUID trace.

**Note**

Using the **no** form of this command does not turn off the debugging.

Examples

The following is sample output from the **voice call debug** command when the **full-guid** keyword is specified:

```
Router# voice call debug full-guid

!
00:05:12: //1/0E2C8A90-BC00-11D5-8002-DACCFDCEF87D/VTSP:(0:D):0:0:4385/vtsp_insert_cdb:
00:05:12: //-1/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx/CCAPI/cc_incr_if_call_volume:
00:05:12: //1/0E2C8A90-BC00-11D5-8002-DACCFDCEF87D/VTSP:(0:D):0:0:4385/vtsp_open_voice_and
_set_params:
00:05:12: //1/0E2C8A90-BC00-11D5-8002-DACCFDCEF87D/VTSP:(0:D):0:0:4385/vtsp_modem_proto_fr
om_cdb:
00:05:12: //1/0E2C8A90-BC00-11D5-8002-DACCFDCEF87D/VTSP:(0:D):0:0:4385/set_playout_cdb:
00:05:12: //1/0E2C8A90-BC00-11D5-8002-DACCFDCEF87D/VTSP:(0:D):0:0:4385/vtsp_dsp_echo_cance
ller_control:
```

The “//1/” output indicates that CallEntryID for the call control API (CCAPI) module is not available. [Table 342](#) describes the significant fields shown in the display.

Table 342 voice call debug full-guid Field Descriptions

| Field | Description |
|---------------------|--|
| VTSP:(0:D):0:0:4385 | Identifies the VTSP module, port name, channel number, DSP slot, and DSP channel number. |
| vtsp_insert_cdb | Identifies the function name. |
| CCAPI | Identifies the CCAPI module. |

The following is sample output from the **voice call debug** command when the **short-header** keyword is specified:

```
Router(config)# voice call debug short-header

!
00:05:12: //1/vtsp_insert_cdb:
00:05:12: //-1/cc_incr_if_call_volume:
00:05:12: //1/vtsp_open_voice_and_set_params:
00:05:12: //1/vtsp_modem_proto_from_cdb:
00:05:12: //1/set_playout_cdb:
00:05:12: //1/vtsp_dsp_echo_canceller_control:
```

The output “//1/” indicates that CallEntryID for CCAPI is not available.

| Related Commands | Command | Description |
|------------------|------------------------------------|---|
| | debug rtsp api | Displays debug output for the RTSP client API. |
| | debug rtsp client | Displays debug output for the RTSP client data. |
| | debug rtsp error | Displays error message for RTSP data. |
| | debug rtsp pmh | Displays debug messages for the PMH. |
| | debug rtsp socket | Displays debug output for the RTSP client socket data. |
| | debug voip ccapi error | Traces error logs in the CCAPI. |
| | debug voip ccapi inout | Traces the execution path through the CCAPI. |
| | debug voip ivr all | Displays all IVR messages. |
| | debug voip ivr applib | Displays IVR API libraries being processed. |
| | debug voip ivr callsetup | Displays IVR call setup being processed. |
| | debug voip ivr digitcollect | Displays IVR digits collected during the call. |
| | debug voip ivr dynamic | Displays IVR dynamic prompt play debug. |
| | debug voip ivr error | Displays IVR errors. |
| | debug voip ivr script | Displays IVR script debug. |
| | debug voip ivr settlement | Displays IVR settlement activities. |
| | debug voip ivr states | Displays IVR states. |
| | debug voip ivr telcommands | Displays the TCL commands used in the script. |
| | debug voip rawmsg | Displays the raw VoIP message. |
| | debug vtsp all | Enables debug vtsp session , debug vtsp error , and debug vtsp dsp . |
| | debug vtsp dsp | Displays messages from the DSP. |
| | debug vtsp error | Displays processing errors in the VTSP. |
| | debug vtsp event | Displays the state of the gateway and the call events. |
| | debug vtsp port | Limits VTSP debug output to a specific voice port. |
| | debug vtsp rtp | Displays the voice telephony RTP packet debugging. |
| | debug vtsp send-nse | Triggers the VTSP software module to send a triple redundant NSE. |
| | debug vtsp session | Traces how the router interacts with the DSP. |
| | debug vtsp stats | Debugs periodic statistical information sent and received from the DSP |
| | debug vtsp vofr subframe | Displays the first 10 bytes of selected VoFR subframes for the interface. |
| | debug vtsp tone | Displays the types of tones generated by the VoIP gateway. |