

# debug x25

To display information about X.25 traffic, use one of the following **debug x25** commands in privileged EXEC mode. The commands allow you to display all information or an increasingly restrictive part of the information.

**Caution**

---

This command 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. To prevent this, do one or more of the following:

- Disable logging of debug output to the console. Refer to the **logging console** command for more information.
  - Configure the router to discard console output when the buffer overflows. Refer to the **logging console guaranteed** command for more information.
  - Use this command only when all of the reportable X.25 traffic is flowing at a data rate of less than five packets per second (pps).
- 

To display information about all X.25 traffic, including traffic for X.25, Connection Mode Network Service (CMNS), and X.25 over TCP (XOT) services, use the **debug x25** command (default **all**). To disable debugging output, use the **no** form of this command.

**debug x25**

**no debug x25**

To display information about all X.25 traffic except data and resource record packets, use the **debug x25 events** command. To disable debugging output, use the **no** form of this command.

**debug x25 [events]**

**no debug x25 [events]**

To display information about a specific X.25 service class, use the following form of the **debug x25** command. To disable debugging output, use the **no** form of this command.

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

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

To display information about a specific X.25 or CMNS context, use the following form of the **debug x25** command. To disable debugging output, use the **no** form of this command.

**debug x25 interface {serial-interface | cmns-interface mac mac-address} [events | all]**

**no debug x25 interface {serial-interface | cmns-interface mac mac-address} [events | all]**

To display information about a specific X.25 or CMNS virtual circuit, use the following form of the **debug x25** command. 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]
```

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

To display information about traffic for all virtual circuits that use a given number, use the following form of the **debug x25** command. The **no** form of this command removes the filter for a particular virtual circuit from the **debug x25 all** or **debug x25 events** output. To disable debugging output, use the **no** form of this command.

```
debug x25 vc number [events | all]
```

```
no debug x25 vc number [events | all]
```

To display information about traffic to or from a specific X.25 over TCP (XOT) host, use the following form of the **debug x25 xot** command. 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]
```

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

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

```
debug x25 aodi
```

```
no debug x25 aodi
```

Syntax Description	
<b>events</b>	(Optional) Displays all traffic except Data and Receiver Ready (RR) packets.
<b>only</b>   <b>cmns</b>   <b>xot</b>	(Optional) Displays information about the specified services: X.25 only, CMNS, or XOT.
<b>all</b>	(Optional) Displays all traffic.
<i>serial-interface</i>	X.25 serial interface.
<i>cmns-interface</i> <b>mac</b> <i>mac-address</i>	MAC address of the CMNS interface and remote host. The interface type can be Ethernet, Token Ring, or Fiber Distributed Data Interface (FDDI).
<b>vc number</b>	Virtual circuit number. Range is from 1 to 4095.
<b>remote</b> <i>ip-address</i> <b>[port number]</b>	(Optional) Remote IP address and, optionally, a port number. Range is 1 to 65535.
<b>local</b> <i>ip-address</i> <b>[port number]</b>	(Optional) Local host IP address and, optionally, a port number. Range is 1 to 65535.
<b>aodi</b>	Causes the <b>debug x25</b> command to display Always On/Dynamic ISDN (AO/DI) events and processing information.

**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 <b>debug x25 events</b> command to describe the events that occur while the X.25 address is being resolved to an IP address using a DNS server. The <b>debug domain</b> command can be used along with <b>debug x25 events</b> 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 <b>debug x25 events</b> command to describe events that occur during CUG activity.
	12.2(8)T	The <b>debug x25 events</b> command was enhanced to display events specific to Record Boundary Preservation protocol.

**Usage Guidelines** 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** command forms 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.

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

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 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.

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 **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 AO/DI.

**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
```

The following example shows a sequence of increasingly restrictive **debug x25** commands:

```
Router# debug x25

X.25 packet debugging is on

Router# debug x25 events

X.25 special event debugging is on

Router# debug x25 interface serial 0

X.25 packet debugging is on
X.25 debug output restricted to interface Serial0

Router# debug x25 vc 1024

X.25 packet debugging is on
X.25 debug output restricted to VC number 1024

Router# debug x25 interface serial 0 vc 1024

X.25 packet debugging is on
X.25 debug output restricted to interface Serial0
X.25 debug output restricted to VC number 1024

Router# debug x25 interface serial 0 vc 1024 events

X.25 special event debugging is on
X.25 debug output restricted to interface serial 0
X.25 debug output restricted to VC number 1024
```

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 events for X.25 CUGs

The following example of the **debug x25 events** command shows output related to the X.25 CUGs feature. It shows messages concerning a data communications equipment (DCE) device rejecting a call because the selected network CUG was not subscribed to by the caller.

Router# **debug x25 events**

```

00:48:33:Serial1:X.25 I R1 Call (14) 8 lci 1024
00:48:33: From (3):111 To (3):444
00:48:33: Facilities:(2)
00:48:33: Closed User Group (basic):40
00:48:33: Call User Data (4):0x01000000 (pad)
00:48:33:X.25 Incoming Call packet, Closed User Group (CUG) protection, selected network
CUG not subscribed
00:48:33:Serial1:X.25 O R1 Clear (5) 8 lci 1024
00:48:33: Cause 11, Diag 65 (Access barred/Facility code not allowed)

```

### debug x25 events for DNS-Based X.25 Routing

The following example of the **debug x25 events** command shows output related to the DNS-Based X.25 Routing feature. It shows messages concerning access to the DNS server. In the following example, nine alternate addresses for one XOT path are entered in 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 events** command 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 events** command when RBP has been configured using the **x25 pvc rbp local** command.

The following display shows data on the 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 events** command 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 events** command when RBP has been configured using the **x25 pvc rbp remote** command.

The following display shows that the X.25 permanent virtual circuit (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
```

[Table 309](#) describes the significant fields shown in the displays.

**Table 309** *debug x25 Command 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.
R3	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 309** *debug x25 Command 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"> <li>Restart</li> <li>Restart Confirm</li> <li>Diagnostic</li> </ul> <p>P Events</p> <ul style="list-style-type: none"> <li>Call</li> <li>Call Confirm</li> <li>Clear</li> <li>Clear Confirm</li> </ul> <p>D Events</p> <ul style="list-style-type: none"> <li>Reset</li> <li>Reset Confirm</li> </ul> <p>D1 Events</p> <ul style="list-style-type: none"> <li>Data</li> <li>Receiver Not Ready (RNR)</li> <li>RR (Receiver Ready)</li> <li>Interrupt</li> <li>Interrupt Confirm</li> </ul> <p>XOT Overhead</p> <ul style="list-style-type: none"> <li>PVC Setup</li> </ul>
(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. See the appendix “X.25 Cause and Diagnostic Codes” in the <i>Cisco IOS Debug Command Reference</i> 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” in the <i>Cisco IOS Debug Command Reference</i> for an explanation of these codes.
(Network operational/ No additional information)	The standard explanations of the Cause and Diagnostic codes ( <i>cause/diag</i> ).

# 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 example shows sample output for 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 310](#) describes significant fields shown in the display.

**Table 310** *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
<code>debug x25</code>	Displays information about X.25 traffic.

# debug x28

To monitor error information and X.28 connection activity, use the **debug x28** privileged 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.

---

## Examples

The following is sample output while the 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 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 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 xcct all
External call control all debugging is off

AS5300-TGW#
```

## Related Commands

Command	Description
<a href="#">debug xcctsp error</a>	Enables debugging on external call control errors.
<a href="#">debug xcctsp session</a>	Enables debugging on external call control sessions.

# debug xcctsp error

To debug External Call Control 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 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 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
<a href="#">debug xcctsp all</a>	Enables debugging on all external call control levels.
<a href="#">debug xcctsp session</a>	Enables debugging on external call control sessions.

# debug xcctsp session

To debug External Call Control 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 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 session-level debugging:

```
AS5300-TGW# debug xcct session  
External call control session debugging is on
```

```
AS5300-TGW# no debug xcct session  
External call control session debugging is off
```

```
AS5300-TGW#
```

## Related Commands

Command	Description
<a href="#">debug xcctsp all</a>	Enables debugging on external call control levels.
<a href="#">debug xcctsp error</a>	Enables debugging on external call control errors.

# debug xcsp

To display the debug 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

<b>all</b>	Provides debug information about XCSP events and continuity testing (COT).
<b>cot</b>	Provides debug information about XCSP and COT. The <b>cot</b> keyword is not used with the NAS Package for MGCP feature.
<b>event</b>	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 and implemented on the Cisco AS5850.

## Usage Guidelines

This command is used with the Network Access Server Package for Media Gateway Control Protocol feature. 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 examples show output for 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**

<b>Command</b>	<b>Description</b>
<b>show vrm vdevices</b>	Displays the status of a router port under the control of the XCSP subsystem.
<b>show xcsp slot</b>	Displays the status of a router slot under the control of the XCSP subsystem.

# 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

<b>full-guid</b>	Displays the GUID in a 16-byte header.  <b>Note</b> When the <b>no</b> version of this command is input with the <b>full-guid</b> keyword, the short 6-byte version is displayed. This is the default.
<b>short-header</b>	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, 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 argument was removed and the short-header argument 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 for 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 311](#) describes the significant fields shown in the display.

**Table 311 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 for 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
	<b>debug rtsp api</b>	Displays debug output for the RTSP client API.
	<b>debug rtsp client</b>	Displays debug output for the RTSP client data.
	<b>debug rtsp error</b>	Displays error message for RTSP data.
	<b>debug rtsp pmh</b>	Displays debug messages for the PMH.
	<b>debug rtsp socket</b>	Displays debug output for the RTSP client socket data.
	<b>debug voip ccapi error</b>	Traces error logs in the CCAPI.
	<b>debug voip ccapi inout</b>	Traces the execution path through the CCAPI.
	<b>debug voip ivr all</b>	Displays all IVR messages.
	<b>debug voip ivr applib</b>	Displays IVR API libraries being processed.
	<b>debug voip ivr callsetup</b>	Displays IVR call setup being processed.
	<b>debug voip ivr digitcollect</b>	Displays IVR digits collected during the call.
	<b>debug voip ivr dynamic</b>	Displays IVR dynamic prompt play debug.
	<b>debug voip ivr error</b>	Displays IVR errors.
	<b>debug voip ivr script</b>	Displays IVR script debug.
	<b>debug voip ivr settlement</b>	Displays IVR settlement activities.
	<b>debug voip ivr states</b>	Displays IVR states.
	<b>debug voip ivr telcommands</b>	Displays the TCL commands used in the script.
	<b>debug voip rawmsg</b>	Displays the raw VoIP message.
	<b>debug vtsp all</b>	Enables <b>debug vtsp session</b> , <b>debug vtsp error</b> , and <b>debug vtsp dsp</b> .
	<b>debug vtsp dsp</b>	Displays messages from the DSP.
	<b>debug vtsp error</b>	Displays processing errors in the VTSP.
	<b>debug vtsp event</b>	Displays the state of the gateway and the call events.
	<i>debug vtsp port</i>	Limits VTSP debug output to a specific voice port.
	<b>debug vtsp rtp</b>	Displays the voice telephony RTP packet debugging.
	<b>debug vtsp send-nse</b>	Triggers the VTSP software module to send a triple redundant NSE.
	<b>debug vtsp session</b>	Traces how the router interacts with the DSP.
	<b>debug vtsp stats</b>	Debugs periodic statistical information sent and received from the DSP
	<b>debug vtsp vofr subframe</b>	Displays the first 10 bytes of selected VoFR subframes for the interface.
	<b>debug vtsp tone</b>	Displays the types of tones generated by the VoIP gateway.

