debug tag-template event through debug voip application vxml

debug tag-template event

To display the tag application on a session (an Authentication Proxy or Extensible Authentication Protocol [EAP] over UDP session), use the **debug tag-template event**command in privileged EXEC mode. To disable debugging, use the **no** form of this command.

debug tag-template event

no debug tag-template event

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging is turned off.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.4(6)T	This command was introduced.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.

Examples

The following example shows that the tag application on a session is to be displayed:

```
Router# debug tag-template event
```

Related Commands

Command	Description
show epm sessions ip	Displays whether tag policies have been applied.

debug tarp events

To display information on Target Identifier Address Resolution Protocol (TARP) activity, use the **debug tarp events**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug tarp events

no debug tarp events

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

For complete information on the TARP process, use the **debug tarp packets** command along with the **debug tarp events** command. Events are usually related to error conditions.

Examples

The following is sample output from the **debug tarp events**and **debug tarp packets** commands after the **tarp resolve** command was used to determine the network service access point (NSAP) address for the TARP target identifier (TID) named *artemis*.

```
Router# debug tarp events
Router# debug tarp packets
Router# tarp resolve artemis
Type escape sequence to abort.
Sending TARP type 1 PDU, timeout 15 seconds...
 NET corresponding to TID artemis is 49.0001.1111.1111.1111.00
*Mar  1 00:43:59: TARP-PA: Propagated TARP packet, type 1, out on Ethernet0
*Mar  1 00:43:59:        Lft = 100, Seq = 11, Prot type = 0xFE, URC = TRUE
*Mar  1 00:43:59:        Ttid len = 7, Stid len = 8, Prot addr len = 10
*Mar  1 00:43:59:        Destination NSAP: 49.0001.1111.1111.1111.00
*Mar  1 00:43:59:        Originator's NSAP: 49.0001.3333.3333.3333.00
*Mar  1 00:43:59:        Target TID: artemis
*Mar  1 00:43:59:        Originator's TID: cerd
*Mar  1 00:43:59: TARP-EV: Packet not propagated to 49.0001.4444.4444.4444.00 on
         interface Ethernet0 (adjacency is not in UP state)
*Mar  1 00:43:59: TARP-EV: No route found for TARP static adjacency
         55.0001.0001.1111.1111.1111.1111.1111.1111.1111.00 - packet not sent
*Mar  1 00:43:59: TARP-PA: Received TARP type 3 PDU on interface Ethernet0
*Mar  1 00:43:59:        Lft = 100, Seq = 5, Prot type = 0xFE, URC = TRUE
*Mar  1 00:43:59:        Ttid len = 0, Stid len = 7, Prot addr len = 10
*Mar  1 00:43:59:        Packet sent/propagated by 49.0001.1111.1111.1111.af
*Mar  1 00:43:59:        Originator's NSAP: 49.0001.1111.1111.1111.00
*Mar  1 00:43:59:        Originator's TID: artemis
*Mar  1 00:43:59: TARP-PA: Created new DYNAMIC cache entry for artemis
```

The table below describes the significant fields shown in display.

Table 1: debug tarp events Field Descriptions--tarp resolve Command

Field	Descriptions
Sending TARP type 1 PDU	Protocol data unit (PDU) requesting the NSAP of the specified TID.

Field	Descriptions
timeout	Number of seconds the router will wait for a response from the Type 1 PDU. The timeout is set by the **tarp t1-response-timer** command.
NET corresponding to	NSAP address (in this case, 49.0001.1111.1111.1111.00) for the specified TID.
*Mar 1 00:43:59	Debug time stamp.
TARP-PA: Propagated	TARP packet: A Type 1 PDU was sent out on Ethernet interface 0.
Lft	Lifetime of the PDU (in hops).
Seq	Sequence number of the PDU.
Prot type	Protocol type of the PDU.
URC	Update remote cache bit.
Ttid len	Destination TID length.
Stid len	Source TID length.
Prot addr len	Protocol address length (bytes).
Destination NSAP	NSAP address that the PDU is being sent to.
Originator's NSAP	NSAP address that the PDU was sent from.
Target TID	TID that the PDU is being sent to.
Originator's TID	TID that the PDU was sent from.
TARP-EV: Packet not propagated	TARP event: The Type 1 PDU was not propagated on Ethernet interface 0 because the adjacency is not up.
TARP-EV: No route found	TARP event: The Type 1 PDU was not sent because no route was available.
TARP-PA: Received TARP	TARP packet: A Type 3 PDU was received on Ethernet interface 0.
Packet sent/propagated by	NSAP address of the router that sent or propagated the PDU.
TARP-PA: Created new DYNAMIC cache entry	TARP packet: A dynamic entry was made to the local TID cache.

Related Commands

Command	Description
debug tarp packets	Displays general information on TARP packets received, generated, and propagated on the router.

debug tarp packets

To display general information on Target Identifier Address Resolution Protocol (TARP) packets received, generated, and propagated on the router, use the **debug tarp packets**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug tarp packets

no debug tarp packets

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

For complete information on the TARP process, use the **debug tarp events** command along with the **debug tarp packet** command. Events are usually related to error conditions.

Examples

The following is sample output from the **debug tarp packet** command after the **tarp query**command was used to determine the TARP target identifier (TID) for the NSAP address 49.0001.3333.3333.3333.00:

```
Router# debug tarp packets
Router# debug tarp events
Router# tarp query 49.0001.3333.3333.3333.00
Type escape sequence to abort.
Sending TARP type 5 PDU, timeout 40 seconds...
 TID corresponding to NET 49.0001.3333.3333.3333.00 is cerdiwen
*Mar  2 03:10:11: TARP-PA: Originated TARP packet, type 5, to destination
49.0001.3333.3333.3333.00
*Mar  2 03:10:11: TARP-PA: Received TARP type 3 PDU on interface Ethernet0
*Mar  2 03:10:11:        Lft = 100, Seq = 2, Prot type = 0xFE, URC = TRUE
*Mar  2 03:10:11:        Ttid len = 0, Stid len = 8, Prot addr len = 10
*Mar  2 03:10:11:        Packet sent/propagated by 49.0001.3333.3333.3333.af
*Mar  2 03:10:11:        Originator's NSAP: 49.0001.3333.3333.3333.00
*Mar  2 03:10:11:        Originator's TID: cerdiwen
*Mar  2 03:10:11: TARP-PA: Created new DYNAMIC cache entry for cerdiwen
```

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

Table 2: debug tarp packets Field Descriptions--tarp query Command

Field	Descriptions
Sending TARP type 5 PDU	Protocol data unit (PDU) requesting the TID of the specified NSAP.
timeout	Number of seconds the router will wait for a response from the Type 5 PDU. The timeout is set by the **tarp arp-request-timer** command.
TID corresponding to NET	TID (in this case *cerdiwen*) for the specified NSAP address.

Field	Descriptions
*Mar 2 03:10:11	Debug time stamp.
TARP-PA: Originated TARP packet	TARP packet: A Type 5 PDU was sent.
TARP P-A: Received TARP	TARP packet: A Type 3 PDU was received.
Lft	Lifetime of the PDU (in hops).
Seq	Sequence number of the PDU.
Prot type	Protocol type of the PDU.
URC	The update remote cache bit.
Ttid len	Destination TID length.
Stid len	Source TID length.
Prot addr len	Protocol address length (in bytes).
Packet sent/propagated	NSAP address of the router that sent or propagated the PDU.
Originator's NSAP	NSAP address that the PDU was sent from.
Originator's TID	TID that the PDU was sent from.
TARP-PA: Created new DYNAMIC cache entry	TARP packet: A dynamic entry was made to the local TID cache.

Related Commands

Command	Modification
debug tarp events	Displays information on TARP activity.

debug tbridge virtual-port

To display Transparent Bridging Virtual Port events debug messages, use the **debug tbridge virtual-port**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug tbridge virtual-port

no debug tbridge virtual-port

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

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

Examples

The following is sample output from the **debug tbridge virtual-port**command:

```
Router# debug tbridge virtual-port
Transparent Bridging Virtual Port Events debugging is on
Router#
vBridge-Port: Received packet (vLAN 100) on FastEthernet0/0 matches with lw-vLAN range.
Set packet input interface to vBridgePort2/1.
```

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

Table 3: debug tbridge virtual-port Field Descriptions

Field	Description
vBridge-Port	Identifies the message as a Transparent Bridging Virtual Port debug message.
vLAN 100	The VLAN ID of the packet.
vBridgePort2/1	The interface the packet is to be bridged to.

debug tcam_mgr

To debug the ternary content addressable memory (TCAM) manager, use the **debug tcam_mgr**commandin privileged EXEC configuration mode.

debug tcam_mgr {**error**| **event**| **profile**}

no debug tcam_mgr {**error**| **event**| **profile**}

Syntax Description

error	Enables debug messages related to TCAM manager errors.
event	Enables debug messages for TCAM manager events.
profile	Enables debug messages about the amount of time it takes to add and remove entries from the TCAM regions.

Command Default

No default behavior or values.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0 S	This command was introduced.
12.2(20)S2	Thiscommand was integrated into Cisco IOS Release 12.2(20)S2.

Usage Guidelines

The **debug tcam_mgr** command is intended for use by Cisco Systems technical support personnel.

Caution Because debugging output is assigned high priority in the CPU process, it can render the system unusable. For this reason, use **debug** commands only to troubleshoot specific problems or during troubleshooting sessions with Cisco Systems technical support personnel. Moreover, it is best to use **debug** commands during periods of lower network traffic and fewer users. Debugging during these periods decreases the likelihood that increased **debug** command processing overhead will affect system use.

Examples

The following example enables TCAM manager event debug messages. It shows the messages associated with shutting down and restarting an interface on the the 4-Port 10/100 Fast Ethernet SPA located in the top subslot (0) of the MSC that is installed in slot 4 of the Cisco 7304 router:

```
Router# debug tcam_mgr event
TCAM Manager Events debugging is on
Router# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# int fast 4/0/0
Router(config-if)# shut
Router(config-if)#
4d01h: %LINK-5-CHANGED: Interface FastEthernet4/0/0, changed state to administratively down
4d01h: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet4/0/0, changed state to
 down
Router(config-if)#
Router(config-if)# no shut
Router(config-if)#
4d01h: Freeing VC at 0 from mask at 0
4d01h: Freeing VC at 1 from mask at 0
4d01h: Freeing VC at 0 from mask at 8
4d01h:  Found Mbu at offset 0 index 0
4d01h: Allocated mbu at offset 0 index 0, vc_index 0 region 0
4d01h:  Found Mbu at offset 0 index 0
4d01h: Allocated mbu at offset 0 index 0, vc_index 1 region 0
4d01h:  Found Mbu at offset 0 index 1
4d01h: Allocated mbu at offset 0 index 1, vc_index 0 region 0
4d01h: %LINK-3-UPDOWN: Interface FastEthernet4/0/0, changed state to up
4d01h: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet4/0/0, changed state to
 up
```

Related Commands

Command	Description
show controllers fastethernet	Displays Fasgt Ethernet interface information, transmission statistics and errors, and applicable MAC destination address and VLAN filtering tables.
show controllers gigabitethernet	Displays Gigabit Ethernet interface information, transmission statistics and errors, and applicable MAC destination address and VLAN filtering tables.
show tcam-mgr subslot	Displays TCAM manager information for SPAs.
test hw-module subslot policyram	Tests the policy table used by the FPGA device for TCAM lookup on a SPA.
test hw-module subslot tcam	Tests the TCAM device on a SPA.

debug tccs signaling

To see information about the transparent Common Channel Signaling (CCS) connection, use the **debug tccs signaling** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug tccs signaling

no debug tccs signaling

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(7)XK	This command was introduced.
12.1(2)T	This command was integrated into Release 12.1(2)T and Release 12.1(2)T.

Usage Guidelines

Caution Use this command with caution, because it displays every packet that the D channel transmits to the packet network and to the PBX. This command is CPU-intensive and should be used only as a last resort.

Use this command to debug a transparent CCS connection in the following cases:

- Observe the results of the **ccs connect** command results when you configure the setup.
- Observe CCS traffic at run time; the output shows the actual CCS packets received at run time and the number of packets received and sent.

Examples

The following shows sample output from the command on both the originating and terminating sides:

```
Router# debug tccs signaling
TCCS Domain packet debugging is on
mazurka-4#
01:37:12: 1 tccs packets received from the port.
01:37:12: 1 tccs packets received from the nework.
01:37:12: tx_tccs_fr_pkt:pkt rcvd from network->tx_start
01:37:12: tx_tccs_fr_pkt: dlci=37, cid=100, payld-type =0,
        payld-length=162, cid_type=424
01:37:12: datagramsize=26
01:37:12: [0] A4 40 C0 0
```

```
01:37:12: [4] 86 86 86 86
01:37:12: [8] 86 86 86 86
01:37:12: [12] 86 86 86 86
01:37:12: [16] 86 86 86 86
01:37:12: [20] 86 86 86 86
01:37:12: [24] 86 86 11 48
01:37:12: 2 tccs packets received from the port.
01:37:12: 1 tccs packets received from the nework.
01:37:12: pri_tccs_rx_intr:from port->send_sub_channel
01:37:12: tccs_db->vcd = 37, tccs_db->cid = 100
01:37:12: pak->datagramsize=25
01:37:12: [0] A4 40 C0 0
01:37:12: [4] 42 43 43 43
01:37:12: [8] 43 43 43 43
01:37:12: [12] 43 43 43 43
01:37:12: [16] 43 43 43 43
01:37:12: [20] 43 43 43 43
01:37:12: [24] 43 43 43 0
Router# debug tccs signaling
00:53:26: 61 tccs packets received from the port.
00:53:26: 53 tccs packets received from the nework.
00:53:26: pri_tccs_rx_intr:from port->send_sub_channel
00:53:26: tccs_db->vcd = 37, tccs_db->cid = 100
00:53:26: pak->datagramsize=7
00:53:26: [0] A4 40 C0 0
00:53:26: [4] 0 1 7F 64
00:53:27: 62 tccs packets received from the port.
00:53:27: 53 tccs packets received from the nework.
00:53:27: pri_tccs_rx_intr:from port->send_sub_channel
00:53:27: tccs_db->vcd = 37, tccs_db->cid = 100
00:53:27: pak->datagramsize=7
00:53:27: [0] A4 40 C0 0
00:53:27: [4] 0 1 7F 64
00:53:28: 63 tccs packets received from the port.
00:53:28: 53 tccs packets received from the nework.
00:53:28: pri_tccs_rx_intr:from port->send_sub_channel
00:53:28: tccs_db->vcd = 37, tccs_db->cid = 100
00:53:28: pak->datagramsize=7
00:53:28: [0] A4 40 C0 0
00:53:28: [4] 0 1 7F 64
00:53:29: 64 tccs packets received from the port.
00:53:29: 53 tccs packets received from the nework.
```

debug tdm

To display time-division multiplexing (TDM) bus connection information each time a connection is made on Cisco AS5300 access servers, use the **debug tdm** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug tdm [**api**| **detail**| **dynamic**| **pri**| **test**| **tsi**| **vdev**]

no debug tdm [**api**| **detail**| **dynamic**| **pri**| **test**| **tsi**| **vdev**]

Syntax Description

api	(Optional) Displays a debugging message whenever the TDM subsystem application programming interface (API) is invoked from another subsystem.
detail	(Optional) Displays detailed messages (i.e., trace messages) whenever the TDM software executes.
dynamic	(Optional) Displays TDM debugging information whenever a backplane timeslot is allocated or deallocated.
pri	(Optional) Routes modem back-to-back connections from the modem-to-PRI board to modem board. By default, the modem back-to-back connections route from modem board to motherboard to modem board.
test	(Optional) Simulates the failure of allocating a TDM timeslot. Verifies that the software and TDM hardware recover from the failure.
tsi	(Optional) Displays debugging information about the TSI Chip MT8980/MT90820 driver.
vdev	(Optional) TDM per voice device debug <0-2> slot and port number (that is, 0/1). Displays debugging information whenever a modem board TDM connection is made.

Command Modes

Privileged EXEC

Usage Guidelines

The **debug tdm**command output is to be used primarily by a Cisco technical support representative. The **debug tdm** command enables display of debugging messages for specific areas of code that execute.

Examples

The following examples show the turning on of the debug option, performing a modem call, and turning off the debug option:

```
Router# debug tdm api
TDM API debugging is on
Router#
23:16:04: TDM(vdev reg: 0x3C500100/PRI reg: 0x3C400100): two way connection requested.
23:16:04: TDM(reg: 0x3C500100): Close connection to STo8, channel 1
23:16:04: TDM(reg: 0x3C500100): Connect STi4, channel 1 to STo8, channel 1
23:16:04: TDM(reg: 0x3C500100): Close connection to STo4, channel 1
23:16:04: TDM(reg: 0x3C500100): Connect STi8, channel 1 to STo4, channel 1
23:16:04: TDM(reg: 0x3C400100): Close connection to STo12, channel 31
23:16:04: TDM(reg: 0x3C400100): Close connection to STo8, channel 31
23:16:04: TDM(reg: 0x3C400100): Connect STi12, channel 31 to STo4, channel 1
23:16:04: TDM(reg: 0x3C400100): Connect STi4, channel 1 to STo12, channel 31
23:18:22: TDM(reg: 0x3C500100): default RX connection requested.
23:18:22: TDM(reg: 0x3C500100): Close connection to STo8, channel 1
23:18:22: TDM(reg: 0x3C500100): default TX connection requested.
23:18:22: TDM(reg: 0x3C500100): Close connection to STo4, channel 1
23:18:22: TDM(reg: 0x3C500100): Close connection to STo8, channel 1
23:18:22: TDM(reg: 0x3C500100): Close connection to STo4, channel 1
23:18:22: TDM(reg: 0x3C400100): default RX connection requested.
23:18:22: TDM(reg: 0x3C400100): Close connection to STo4, channel 1
23:18:22: TDM(reg: 0x3C400100): Connect STi12, channel 31 to STo8, channel 31
23:18:22: TDM(reg: 0x3C400100): default TX connection requested.
23:18:22: TDM(reg: 0x3C400100): Close connection to STo12, channel 31
23:18:22: TDM(reg: 0x3C400100): Connect STi8, channel 31 to STo12, channel 31
Router# no debug tdm api
TDM API debugging is off
Router# debug tdm detail
TDM Detail Debug debugging is on
router_2#show tdm pool
```

Examples

```
Grp ST Ttl/Free Req(Cur/Ttl/Fail)    Queues(Free/Used)   Pool Ptr
 0 0-3 128 128  0  0    0    0x60CB6B30 0x60CB6B30 0x60CB6B28
 1 4-7 128 128  0  3    0    0x60CB6B40 0x60CB6B40 0x60CB6B2C
Router#
Router# no debug tdm detail
TDM Detail Debug debugging is off
Router# debug tdm dynamic
TDM Dynamic BP Allocation debugging is on
Router#
23:30:16: tdm_allocate_bp_ts(), slot# 1, chan# 3
23:30:16: TDM(reg: 0x3C500100): Open Modem RX ST8, CH3 to BP ST4 CH3
23:30:16: TDM(reg: 0x3C500100): Open Modem TX ST8, CH3 to BP ST4 CH3
23:30:16: TDM Backplane Timeslot Dump @ 0x60E6D244, tdm_free_bptsCount[1] = 127
 vdev_slot : 0x01   bp_stream : 0x04
 vdev_channel : 0x03  bp_channel : 0x03   freeQueue : 0x60CB6B40
23:30:16: TDM(PRI:0x3C400100):Close PRI framer st12 ch31
23:30:16: TDM(PRI:0x3C400100):Close HDLC controller st8 ch31
23:30:43: tdm_deallocate_bp_ts(), slot# 1, chan# 3
23:30:43: TDM(reg: 0x3C500100):Close Modem RX ST8, CH3 to BP ST4 CH3
23:30:43: TDM(reg: 0x3C500100):Close Modem TX ST8, CH3 to BP ST4 CH3
23:30:43: TDM Backplane Timeslot Dump @ 0x60E6D244, tdm_free_bptsCount[1] = 128
 vdev_slot : 0x01   bp_stream : 0x04
 vdev_channel : 0x03  bp_channel : 0x03   freeQueue : 0x60CB6B40
Router#
Router# no debug tdm dynamic
TDM Dynamic BP Allocation debugging is off
Router# debug tdm pri
TDM connectvia PRI feature board debugging is on
Router# no debug tdm pri
TDM connectvia PRI feature board debugging is off
Router# debug tdm test
TDM Unit Test debugging is on
23:52:01: Bad tdm_allocate_bp_ts() call, simulating error condition for vdev in slot 1
```

```
port 5
Router# no debug tdm test
TDM Unit Test debugging is off
Router# debug tdm tsi
TDM TSI debugging is on
Router#
23:56:40: MT90820(reg: 0x3C500100): Close connection to STi8, channel 9
23:56:40: MT90820(reg: 0x3C500100): Connect STi4, channel 10 to STo8, channel 9
23:56:40: MT90820(reg: 0x3C500100): Close connection to STi4, channel 10
23:56:40: MT90820(reg: 0x3C500100): Connect STi8, channel 9 to STo4, channel 10
23:56:40: MT90820(reg: 0x3C400100): Close connection to STi12, channel 31
23:56:40: MT90820(reg: 0x3C400100): Close connection to STi8, channel 31
23:56:40: MT90820(reg: 0x3C400100): Connect STi12, channel 31 to STo4, channel 10
23:56:40: MT90820(reg: 0x3C400100): Connect STi4, channel 10 to STo12, channel 31
23:57:03: MT90820(reg: 0x3C500100): Close connection to STi8, channel 9
23:57:03: MT90820(reg: 0x3C500100): Close connection to STi4, channel 10
23:57:03: MT90820(reg: 0x3C500100): Close connection to STi8, channel 9
23:57:03: MT90820(reg: 0x3C500100): Close connection to STi4, channel 10
23:57:03: MT90820(reg: 0x3C400100): Close connection to STi4, channel 10
23:57:03: MT90820(reg: 0x3C400100): Connect STi12, channel 31 to STo8, channel 31
23:57:03: MT90820(reg: 0x3C400100): Close connection to STi12, channel 31
23:57:03: MT90820(reg: 0x3C400100): Connect STi8, channel 31 to STo12, channel 31
Router#
Router# no debug tdm tsi
TDM TSI debugging is off
Router# debug tdm vdev ?
 <0-2> Slot/port number (i.e. 0/1)
Router# debug tdm vdev 1/8
Enabling TDM debug for voice device in slot 0 port 1
Router#
23:55:00: TDM(vdev reg: 0x3C500100/PRI reg: 0x3C400100): two way connection requested.
23:55:00: tdm_allocate_bp_ts(), slot# 1, chan# 8
23:55:00: TDM(reg: 0x3C500100): Open Modem RX ST8, CH8 to BP ST4 CH9
23:55:00: TDM(reg: 0x3C500100): Open Modem TX ST8, CH8 to BP ST4 CH9
23:55:00: TDM Backplane Timeslot Dump @ 0x60E6D2D4, tdm_free_bptsCount[1] = 127
 vdev_slot : 0x01   bp_stream : 0x04
 vdev_channel : 0x08  bp_channel : 0x09   freeQueue : 0x60CB6B40

23:55:00: TDM(PRI:0x3C400100):Close PRI framer st12 ch31
23:55:00: TDM(PRI:0x3C400100):Close HDLC controller st8 ch31
23:55:31: TDM(reg: 0x3C500100): default RX connection requested.
23:55:31: TDM(reg: 0x3C500100): default TX connection requested.
23:55:31: tdm_deallocate_bp_ts(), slot# 1, chan# 8
23:55:31: TDM(reg: 0x3C500100):Close Modem RX ST8, CH8 to BP ST4 CH9
23:55:31: TDM(reg: 0x3C500100):Close Modem TX ST8, CH8 to BP ST4 CH9
23:55:31: TDM Backplane Timeslot Dump @ 0x60E6D2D4, tdm_free_bptsCount[1] = 128
 vdev_slot : 0x01   bp_stream : 0x04
 vdev_channel : 0x08  bp_channel : 0x09   freeQueue : 0x60CB6B40
Router#
Router# no debug tdm vdev 1/8
Disabling TDM debug for voice device in slot 0 port 1
Router#
```

debug telco-return msg

To display debugging messages for telco-return events, use the **debug cable telco-return msg** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug cable telco-return msg

no debug cable telco-return msg

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging for telco-return messages is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(4)XI	This command was introduced.

Examples

The following is sample output from the **debug cable telco-return msg** command:

```
ubr7223# debug cable telco-return msg
CMTS telco-return msg debugging is on
```

debug telnet

To display information about Telnet option negotiation messages for incoming Telnet connections to a Cisco IOS Telnet server, use the **debug telnet**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug telnet

no debug telnet

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

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

Examples

The following is sample output from the **debug telnet**command:

```
Router# debug telnet
*Oct 28 21:31:12.035:Telnet1/00:1 1 251 1
*Oct 28 21:31:12.035:TCP1/00:Telnet sent WILL ECHO (1)
*Oct 28 21:31:12.035:Telnet1/00:2 2 251 3
*Oct 28 21:31:12.035:TCP1/00:Telnet sent WILL SUPPRESS-GA (3)
*Oct 28 21:31:12.035:Telnet1/00:4 4 251 0
*Oct 28 21:31:12.035:TCP1/00:Telnet sent WILL BINARY (0)
*Oct 28 21:31:12.035:Telnet1/00:40000 40000 253 0
*Oct 28 21:31:12.035:TCP1/00:Telnet sent DO BINARY (0)
*Oct 28 21:31:12.035:Telnet1/00:10000000 10000000 253 31
*Oct 28 21:31:12.035:TCP1/00:Telnet sent DO WINDOW-SIZE (31)
*Oct 28 21:31:12.035:TCP1/00:Telnet received WILL TTY-TYPE (24)
*Oct 28 21:31:12.035:TCP1/00:Telnet sent DO TTY-TYPE (24)
*Oct 28 21:31:12.035:Telnet1/00:Sent SB 24 1
*Oct 28 21:31:12.035:TCP1/00:Telnet received WILL TTY-SPEED (32) (refused)
*Oct 28 21:31:12.035:TCP1/00:Telnet sent DONT TTY-SPEED (32)
*Oct 28 21:31:12.035:TCP1/00:Telnet received DO SUPPRESS-GA (3)
*Oct 28 21:31:12.035:TCP1/00:Telnet received WILL SUPPRESS-GA (3)
*Oct 28 21:31:12.035:TCP1/00:Telnet sent DO SUPPRESS-GA (3)
*Oct 28 21:31:12.035:TCP1/00:Telnet received DO ECHO (1)
*Oct 28 21:31:12.035:TCP1/00:Telnet received DO BINARY (0)
*Oct 28 21:31:12.035:TCP1/00:Telnet received WILL BINARY (0)
*Oct 28 21:31:12.059:TCP1/00:Telnet received WILL COMPORT (44)
*Oct 28 21:31:12.059:TCP1/00:Telnet sent DO COMPORT (44)
*Oct 28 21:31:12.059:TCP1/00:Telnet received DO COMPORT (44)
*Oct 28 21:31:12.059:TCP1/00:Telnet sent WILL COMPORT (44)
*Oct 28 21:31:12.059:TCP1/00:Telnet received WONT WINDOW-SIZE (31)
*Oct 28 21:31:12.059:TCP1/00:Telnet sent DONT WINDOW-SIZE (31)
*Oct 28 21:31:12.059:Telnet1/00:recv SB 24 0
*Oct 28 21:31:12.091:Telnet1/00:recv SB 44 10 TTY1/00:Telnet COMPORT rcvd bad
suboption:0xA/0x1E
*Oct 28 21:31:12.091:Telnet1/00:recv SB 44 1
```

```
*Oct 28 21:31:12.091:Telnet_CP-1/00 baudrate index 0
*Oct 28 21:31:12.091:Telnet1/00:Sent SB 44 101 X.dctBXctBXctBX`W`P`>
*Oct 28 21:31:12.091:Telnet1/00:recv SB 44 2
*Oct 28 21:31:12.091:Telnet_CP-1/00 datasize index 8  8
*Oct 28 21:31:12.091:Telnet1/00:Sent SB 44 102X.dctBXctBXctBX`W`P`>
*Oct 28 21:31:12.091:Telnet1/00:recv SB 44 3
*Oct 28 21:31:12.091:Telnet_CP-1/00 parity index 1  0
*Oct 28 21:31:12.091:Telnet1/00:Sent SB 44 103 X.dctBXctBXctBX`W`P`>
*Oct 28 21:31:12.091:Telnet1/00:recv SB 44 4
*Oct 28 21:31:12.091:Telnet_CP-1/00 stopbits index 1
*Oct 28 21:31:12.091:Telnet1/00:Sent SB 44 104 X.dctBXctBXctBX`W`P`>
*Oct 28 21:31:12.091:Telnet1/00:recv SB 44 5
*Oct 28 21:31:12.091:Telnet_CP-1/00 HW flow on
*Oct 28 21:31:12.091:Telnet1/00:Sent SB 44 105 X.dctBXctBXctBX`W`P`>
*Oct 28 21:31:12.091:Telnet1/00:recv SB 44 11 nTTY1/00:Telnet COMPORT rcvd ba
d suboption:0xB/0xEE
*Oct 28 21:31:12.091:Telnet1/00:recv SB 44 5
*Oct 28 21:31:12.091:Telnet_CP-1/00 unimplemented option 0x10
*Oct 28 21:31:12.091:Telnet1/00:Sent SB 44 105
*Oct 28 21:31:12.091:Telnet1/00:recv SB 44 5
*Oct 28 21:31:12.091:Telnet_CP-1/00 DTR on
*Oct 28 21:31:12.091:Telnet1/00:Sent SB 44 105X.dctBXctBXctBX`W`P`>
*Oct 28 21:31:12.091:TCP1/00:Telnet received WONT WINDOW-SIZE (31)
*Oct 28 21:31:12.099:Telnet1/00:Sent SB 44 107 3
*Oct 28 21:31:12.099:COMPORT1/00:sending notification 0x33
```

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

Table 4: debug telnet Field Descriptions

Field	Description
Telnet1/00: 1 1 251 1	Untranslated decimal option negotiations that are sent. 1/00 denotes the line number that the Telnet server is operating on.
TCP1/00:	Symbolically decoded option negotiations. 1/00 denotes the line number that the Telnet server is operating on. Telnet option negotiations are defined in the following RFCs: • RFC 854--*Telnet Protocol Specification* • RFC 856--*Telnet Binary Transmission* • RFC 858--*Telnet Suppress Go Ahead Option* • RFC 1091--*Telnet Terminal-Type Option* • RFC 1123, sec. 3--*Requirements for Internet Hosts--Application and Support* • RFC 2217--*Telnet Com Port Control Option*

Related Commands

Command	Description
debug ip tcp transactions	Displays information on significant TCP transactions such as state changes, retransmissions, and duplicate packets.

Command	Description
debug modem	Displays modem line activity on an access server.

debug text-to-fax

To show information relating to the off-ramp text-to-fax conversion, use the **debug text-to-fax** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug text-to-fax

no debug text-to-fax

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(4)T	This command was introduced.

Examples

The following debug output shows the off-ramp text-to-fax conversion.

```
Router# debug text-to-fax Text to fax debugging is on
Router#6d03h: text2fax_data_handler: START_OF_CONNECTION
6d03h: text2fax_data_handler: new_context
6d03h: text2fax_data_handler: resolution: fine
6d03h: text2fax_data_handler: buffer size: 50
6d03h: text2fax_put_buffer: START_OF_FAX_PAGE
6d03h: text2fax_put_buffer: START_OF_FAX_PAGE
6d03h: text2fax_put_buffer: END_OF_FAX_PAGE. Dial now ...if not in progress
6d03h: text2fax_data_handler: START_OF_DATA
6d03h: text2fax_data_handler: END_OF_DATA
6d03h: text2fax_data_handler: Dispose context
6d03h: text2fax_data_handler: START_OF_CONNECTION
6d03h: text2fax_data_handler: END_OF_CONNECTION
6d03h: %FTSP-6-FAX_CONNECT: Transmission
6d03h: %FTSP-6-FAX_DISCONNECT: Transmission
6d03h: %LINK-3-UPDOWN: Interface Serial1:22, changed state to down
```

debug tftp

To display Trivial File Transfer Protocol (TFTP) debugging information when encountering problems netbooting or using the **copy tftp system:running-config** or **copy system:running-config tftp** commands, use the **debug tftp** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug tftp

no debug tftp

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Examples

The following is sample output from the **debug tftp** command from the **copy system:running-config tftp** EXEC command:

```
Router# debug tftp
TFTP: msclock 0x292B4; Sending write request (retry 0), socket_id 0x301DA8
TFTP: msclock 0x2A63C; Sending write request (retry 1), socket_id 0x301DA8
TFTP: msclock 0x2A6DC; Received ACK for block 0,  socket_id 0x301DA8
TFTP: msclock 0x2A6DC; Received ACK for block 0,  socket_id 0x301DA8
TFTP: msclock 0x2A6DC; Sending block 1 (retry 0), socket_id 0x301DA8
TFTP: msclock 0x2A6E4; Received ACK for block 1,  socket_id 0x301DA8
```

The table below describes the significant fields in the first line of output.

Table 5: debug tftp Field Descriptions

Message	Description
TFTP:	TFTP packet.
msclock 0x292B4;	Internal timekeeping clock (in milliseconds).
Sending write request (retry 0)	TFTP operation.
socket_id 0x301DA8	Unique memory address for the socket for the TFTP connection.

debug tgrep error

To turn on debugging for any Telephony Gateway Registration Protocol (TGREP) errors, use the **debug tgrep error** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug tgrep error

no debug tgrep error

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(1)	This command was introduced.

Usage Guidelines

There is always a performance penalty when using **debug** commands.

The “We already have connection with such itad/tripid combo in progress” message appears when an error occurs where two location servers with the same Internet Telephony Administrative Domain (ITAD), and TripID initiate a Telephony Routing over IP (TRIP) connection to the gateway. When the second OPEN message arrives at the gateway, the **debug trip error** command displays the message.

Examples

The following shows sample output from the **debug tgrep error** command:

```
Errors : Process socket event has an invalid fd to work on
Errors : Process socket event has an invalid fd to work on
Errors : Process socket event has an invalid fd to work on
Errors : Process socket event has an invalid fd to work on
```

After the errors are reported, the open dump begins. The ITAD is identified in the dump.

```
-
---------------------- OPEN DUMP BEGINS ------------------------
 0x1 0xFFFFFFFF 0x0 0xFFFFFFB4 0x0
 0x0 0x4 0x58 0x6 0x7
 0xFFFFFF98 0xFFFFFFA9 0x0 0xC 0x0
 0x1 0x0 0x8 0x0 0x2
 0x0 0x4 0x0 0x0 0x0
 0x3
        Version    :1
        Hold Time    :180
        My ITAD      :1112
        TRIP ID      :101161129
                 Option Paramater #1
                 Param Type: Capability
                 Length 8
```

```
                        Cap Code :Send Receive Capability
                        Cap Len  :4
                                 Send Rec Cap: RCV ONLY MODE
        -->All route types supported
----------------------- OPEN DUMP ENDS -----------------------
```

The “We already have connection with such itad/tripid combo in progress” message appears when an error occurs where two location servers with the same ITAD and TripID initiate a TRIP connection to the gateway.

```
We already have connection with such itad/tripid combo in progress
NBR:16.1.1.203 is not eligible to write, no non(-1) fd yet
Errors : Process socket event has an invalid fd to work on
Error: Active connection to the nbr failed NBR:16.1.1.203
Errors : Process socket event has an invalid fd to work on
Errors : Process socket event has an invalid fd to work on
NBR:16.1.1.203 is not eligible to write, no non(-1) fd yet
Errors : Process socket event has an invalid fd to work on
Errors : Process socket event has an invalid fd to work on
NBR:16.1.1.203 is not eligible to write, no non(-1) fd yet
Errors : Process socket event has an invalid fd to work on
```

Related Commands

Command	Description
debug tgrep events	Turns on debugging for main events occurring throughout the subsystem.
debug tgrep fsm	Turns on debugging for FSM activity.
debug tgrep io	Turns on debugging for detailed socket level activities.
debug tgrep messages	Turns on debugging for the movement of TGREP messages.
debug tgrep msgdump	Turns on debugging for the dump of the details of TGREP messages.
debug tgrep timer-event	Turns on debugging for events that are related to the timer.
debug tgrep timers	Turns on debugging for timer activity.
debug tgrep tripr	Turns on debugging for the TRIP Reporter.
debug voip eddri	Turns on debugging for the EDDRI.

debug tgrep events

To turn on debugging for main events occurring throughout the subsystem, use the **debug tgrep events** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug tgrep events

no debug tgrep events

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(1)	This command was introduced.

Usage Guidelines

There is always a performance penalty when using **debug** commands.

Examples

The following example shows output from the **debug tgrep events** command:

```
tgrep-gw-1-02#Received a TGREP_UPD_TIMER timeout
The bulkSyncQ size is 0 at this time
The tgrepQ size is 0 at this time
```

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

Table 6: debug tgrep events Field Descriptions

Field	Description
Received a TGREP_UPD_TIMER timeout	This event shows that a TGREP update timer timeout event occurred.
The bulkSyncQ size is 0 at this time	This event indicates the size of bulk sync queue.
The tgrepQ size is 0 at this time	This event indicates the size of TGREP queue.

Related Commands

Command	Description
debug tgrep error	Turns on debugging for any errors in functioning.
debug tgrep fsm	Turns on debugging for FSM activity.
debug tgrep io	Turns on debugging for detailed socket level activities.
debug tgrep messages	Turns on debugging for the movement of TGREP messages.
debug tgrep msgdump	Turns on debugging for the dump of the details of TGREP messages.
debug tgrep timer-event	Turns on debugging for events that are related to the timer.
debug tgreptimers	Turns on debugging for timer activity.
debug tgrep tripr	Turns on debugging for the TRIP Reporter.
debug voip eddri	Turns on debugging for the EDDRI.

debug tgrep fsm

To turn on debugging for Finite State Machine (FSM) events, use the **debug tgrep fsm** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug tgrep fsm

no debug tgrep fsm

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(1)	This command was introduced.

Usage Guidelines

There is always a performance penalty when using **debug** commands.

Examples

The following shows sample output from the **debug tgrep fsm** command:

```
Generic routes combined : 0x61FA38B4, 13 bytes
++++++++++++++++++++++++++++++++++++
 0x0 0x2 0x0 0x9 0x0
 0x5 0x0 0x0 0x0 0x3
 0x6D 0x63 0x69
------------------------------------
++++++++++++++++++++++++++++++++++++
 NEXT HOP SERVER : 0x61FA38C1, 10 bytes
++++++++++++++++++++++++++++++++++++
 0x0 0x3 0x0 0x6 0x0
 0x0 0x4 0xFFFFFFD2 0x0 0x0
------------------------------------
++++++++++++++++++++++++++++++++++++
 AD RD PATH : 0x61FA38CB, 10 bytes
++++++++++++++++++++Getting a major event 4 on I/O
```

Here, a write event occurs. Note how the finite state machine details each step of the writing process.

```
Received a TRIP_IO_WRITEQ_BOOLEAN event 313
The peer connection check for fd 1 is success
Writing some pending stuff first NBR:14.1.1.210
Moving ahead with more reading rc = 4
-->Starting regular write for nbr NBR:14.1.1.210
The queuesize before we start is 1
Selected primary socket for NBR:14.1.1.210
The peer connection check for fd 1 is success
Dequeued 1 message (left 0) for NBR:14.1.1.210 for writing to socket
A socket has gulped all that we fed it NBR:14.1.1.210 -- 92 bytes
```

```
Dequeued 0 message (left 0) for NBR:14.1.1.210 for writing to socket
Wrote out the whole socket buffer or Q in 2 attempts NBR:14.1.1.210 rc 4 was
NBR:14.1.1.210 Starting keepalive timer after writing something
Getting a major event 512 on I/O
Received an event on a socket for some nbr
Received Mask event of 0x1 for fd 1
Looking for fd match on nbr NBR:14.1.1.210
```

Now a read event occurs. After this event, the total number of TRIP messages read is displayed.

```
Recieved READ_EVENT for nbr NBR:14.1.1.210
Read 3 bytes from that network for nbr NBR:14.1.1.210
++++++++++++++++++++++++++++++++++++
 This is what we READ : 0x63E79090, 3 bytes
++++++++++++++++++++++++++++++++++++
 0x0 0x3 0x4
------------------------------------
NBR:14.1.1.210 Re-starting hold timer after a message is read
tmsg malloc total memory allocated is 95
Allocated another buffer for TRIP message
TRIP Messages Read so far 1
++++++++++++++++++++++++++++++++++++
 Enqueing this tmsg : 0x691D09DC, 3 bytes
++++++++++++++++++++++++++++++++++++
 0x0 0x3 0x4
------------------------------------
Enqueuing a message into the ReadQ of nbr: NBR:14.1.1.210
Read -1 bytes from that network for nbr NBR++++++++++++++++++
 0x0 0x4 0x0 0x6 0x2
 0x1 0x0 0x0 0x4 0xFFFFFFD2
------------------------------------
```

Statistics for available circuits, total circuits, and call success rate are displayed.

```
++++++++++++++++++++++++++++++++++++
 AD RD PATH : 0x61FA38D5, 10 bytes
++++++++++++++++++++++++++++++++++++
 0x0 0x5 0x0 0x6 0x2
 0x1 0x0 0x0 0x4 0xFFFFFFD2
------------------------------------
++++++++++++++++++++++++++++++++++++
 LOCAL PREF : 0x61FA38DF, 8 bytes
++++++++++++++++++++++++++++++++++++
 0x0 0x7 0x0 0x4 0x0
 0x0 0x0 0x5
------------------------------------
++++++++++++++++++++++++++++++++++++
 Available Ckts : 0x61FA38E7, 8 bytes
++++++++++++++++++++++++++++++++++++
 0x0 0xF 0x0 0x4 0x0
 0x0 0x0 0x17
------------------------------------
++++++++++++++++++++++++++++++++++++
 TOTAL CIRCUITS : 0x61FA38EF, 8 bytes
++++++++++++++++++++++++++++++++++++
 0x0 0x10 0x0 0x4 0x0
 0x0 0x0 0x17
------------------------------------
++++++++++++++++++++++++++++++++++++
 CALL SUCCESS RATE : 0x61FA38F7, 12 bytes
++++++++++++++++++++++++++++++++++++
 0x0 0x11 0x0
tgrep-gw-1-02#
tgrep-gw-1-02#und al:14.1.1.210
Getting a major event 512 on I/O
Errors : Process socket event has an invalid fd to work on
l 0x8 0x0
 0x0 0x0 0x78 0x0 0x0
 0x0 0x7F
------------------------------------
++++++++++++++++++++++++++++++++++++
 PREFIX_ATTRIBUTE : 0x61FA3903, 64 bytes
++++++++++++++++++++++++++++++++++++
```

The prefix is shown here in hex format.

```
0x0 0x12 0x0 0x3C 0x0
0x4 0x31 0x31 0x32 0x38
0x0 0x4 0x31 0x31 0x32
0x37 0x0 0x4 0x31 0x31
0x32 0x36 0x0 0x4 0x31
0x31 0x32 0x35 0x0 0x4
0x31 0x31 0x32 0x34 0x0
0x4 0x31 0x31 0x32 0x33
0x0 0x4 0x31 0x31 0x32
0x32 0x0 0x5 0x39 0x39
0x39 0x39 0x39 0x0 0x9
0x31 0x32 0x33 0x34 0x35
0x36
```

Related Commands

Command	Description
debug tgrep error	Turns on debugging for any errors in functioning.
debug tgrep events	Turns on debugging for main events occurring throughout the subsystem.
debug tgrep io	Turns on debugging for detailed socket level activities.
debug tgrep messages	Turns on debugging for the movement of TGREP messages.
debug tgrep msgdump	Turns on debugging for the dump of the details of TGREP messages.
debug tgrep timer-event	Turns on debugging for events that are related to the timer.
debug tgrep timers	Turns on debugging for timer activity.
debug tgrep tripr	Turns on debugging for the TRIP Reporter.
debug voip eddri	Turns on debugging for the EDDRI.

debug tgrep io

To turn on debugging for detailed socket-level activities, use the **debug tgrep io** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug tgrep io

no debug tgrep io

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(1)	This command was introduced.

Usage Guidelines

There is always a performance penalty when using **debug** commands.

Examples

The following shows sample output from the **debug tgrep io** command:

```
Dispatching a TRIP_EV_NBR_IO_ASYNC_RESET to I/O for NBR:16.1.1.202
Dispatching a TRIP_EV_NBR_IO_ASYNC_RESET to I/O for NBR:16.1.1.203
A socket has gulped all that we fed it NBR:16.1.1.202 -- 5 bytes
Closing all the fds for NBR:16.1.1.202
NBR:16.1.1.202 is not eligible to write, no non(-1) fd yet
NBR:16.1.1.203 is not eligible to write, no non(-1) fd yet
A Socket error has caused a write failure NBR:16.1.1.203 errno 13
Closing all the fds for NBR:16.1.1.203
NBR:16.1.1.202 is not eligible to write, no non(-1) fd yet
NBR:16.1.1.203 is not eligible to write, no non(-1) fd yet
```

At this point, the connection is initiated.

```
Going to initiate a connect to 16.1.1.202
Called a socket_connect with errno 11, confirmation later
Initiated a Async connect call for nbr NBR:16.1.1.202 fd 1
Received Mask event of 0x1 for fd 1
Recieved WRITE_EVENT for nbr NBR:16.1.1.202
Only Active Open Succeeded
Post connect succeeded for the nbr NBR:16.1.1.202, fd 1
A socket has gulped all that we fed it NBR:16.1.1.202 -- 29 bytes
Wrote out the whole socket buffer or Q in 2 attempts NBR:16.1.1.202 rc 4 was
NBR:16.1.1.203 is not eligible to write, no non(-1) fd yet
Received Mask event of 0x1 for fd 1
Recieved READ_EVENT for nbr NBR:16.1.1.202
Read -1 bytes from that network for nbr NBR:16.1.1.202
Received Mask event of 0x1 for fd 1
Recieved READ_EVENT for nbr NBR:16.1.1.202
```

```
Read 3 bytes from that network for nbr NBR:16.1.1.202
Read -1 bytes from that network for nbr NBR:16.1.1.202
```

Errors begin to appear here.

```
Errors : Process socket event has an invalid fd to work on
Received Mask event of 0x1 for fd 1
Recieved READ_EVENT for nbr NBR:16.1.1.202
Read 29 bytes from that network for nbr NBR:16.1.1.202
Read -1 bytes from that network for nbr NBR:16.1.1.202
Errors : Process socket event has an invalid fd to work on
```

After the errors are detected, a dump occurs. The Internet Telephony Administrative Domain (ITAD) and Telephony Routing over IP (TRIP) ID are displayed.

```
------------------------ OPEN DUMP BEGINS -------------------------
 0x1 0xFFFFFFFF 0x0 0xFFFFFFB4 0x0
 0x0 0x4 0x58 0x6 0x7
 0xFFFFFF98 0xFFFFFFA9 0x0 0xC 0x0
 0x1 0x0 0x8 0x0 0x2
 0x0 0x4 0x0 0x0 0x0
 0x3
        Version    :1
        Hold Time    :180
        My ITAD      :1112
        TRIP ID      :101161129
                 Option Paramater #1
                 Param Type: Capability
                 Length 8
                         Cap Code :Send Receive Capability
                         Cap Len  :4
                                 Send Rec Cap: RCV ONLY MODE
        -->All route types supported
------------------------ OPEN DUMP ENDS -------------------------
Doing fd reassignment for nbr NBR:16.1.1.202
Moving ahead with more reading rc = 4
A socket has gulped all that we fed it NBR:16.1.1.202 -- 3 bytes
Wrote out the whole socket buffer or Q in 2 attempts NBR:16.1.1.202 rc 4 was
NBR:16.1.1.203 is not eligible to write, no non(-1) fd yet
Received Mask event of 0x1 for fd 1
Recieved READ_EVENT for nbr NBR:16.1.1.202
Read 3 bytes from that network for nbr NBR:16.1.1.202
Read -1 bytes from that network for nbr NBR:16.1.1.202
Moving ahead with more reading rc = 4
A socket has gulped all that we fed it NBR:16.1.1.202 -- 598 bytes
Wrote out the whole socket buffer or Q in 2 attempts NBR:16.1.1.202 rc 4 was
NBR:16.1.1.203 is not eligible to write, no non(-1) fd yet
Errors : Process socket event has an invalid fd to work on
Received Mask event of 0x1 for fd 1
Recieved READ_EVENT for nbr NBR:16.1.1.202
Read 3 bytes from that network for nbr NBR:16.1.1.202
Read -1 bytes from that network for nbr NBR:16.1.1.202
Errors : Process socket event has an invalid fd to work on
Received Mask event of 0x1 for fd 1
Recieved READ_EVENT for nbr NBR:16.1.1.202
Read 15 bytes from that network for nbr NBR:16.1.1.202
Read -1 bytes from that network for nbr NBR:16.1.1.202
Errors : Process socket event has an invalid fd to work on
Going to initiate a connect to 16.1.1.203
Called a socket_connect with errno 11, confirmation later
Initiated a Async connect call for nbr NBR:16.1.1.203 fd 2
Received Mask event of 0x1 for fd 2
Recieved WRITE_EVENT for nbr NBR:16.1.1.203
The Active connect never succeeded, no passive yet, resetting NBR:16.1.1.203
Error: Active connection to the nbr failed NBR:16.1.1.203
A Socket error has caused a write failure NBR:16.1.1.203 errno 13
Closing all the fds for NBR:16.1.1.203
Post connect succeeded for the nbr NBR:16.1.1.203, fd -1
Moving ahead with more reading rc = 4
NBR:16.1.1.203 is not eligible to write, no non(-1) fd yet
Errors : Process socket event has an invalid fd to work on
Going to initiate a connect to 16.1.1.203
```

```
Called a socket_connect with errno 11, confirmation later
Initiated a Async connect call for nbr NBR:16.1.1.203 fd 2
Received Mask event of 0x1 for fd 2
```

Errors continue to occur. Note that the router still attempts to write, but the connection is not active.

```
Recieved WRITE_EVENT for nbr NBR:16.1.1.203
The Active connect never succeeded, no passive yet, resetting NBR:16.1.1.203
Error: Active connection to the nbr failed NBR:16.1.1.203
A Socket error has caused a write failure NBR:16.1.1.203 errno 13
Closing all the fds for NBR:16.1.1.203
Post connect succeeded for the nbr NBR:16.1.1.203, fd -1
Moving ahead with more reading rc = 4
NBR:16.1.1.203 is not eligible to write, no non(-1) fd yet
Errors : Process socket event has an invalid fd to work on
Received Mask event of 0x1 for fd 1
Recieved READ_EVENT for nbr NBR:16.1.1.202
Read 3 bytes from that network for nbr NBR:16.1.1.202
Read -1 bytes from that network for nbr NBR:16.1.1.202
Errors : Process socket event has an invalid fd to work on
Going to initiate a connect to 16.1.1.203
Called a socket_connect with errno 11, confirmation later
Initiated a Async connect call for nbr NBR:16.1.1.203 fd 2
Received Mask event of 0x1 for fd 2
Recieved WRITE_EVENT for nbr NBR:16.1.1.203
The Active connect never succeeded, no passive yet, resetting NBR:16.1.1.203
Error: Active connection to the nbr failed NBR:16.1.1.203
A Socket error has caused a write failure NBR:16.1.1.203 errno 13
Closing all the fds for NBR:16.1.1.203
Post connect succeeded for the nbr NBR:16.1.1.203, fd -1
Moving ahead with more reading rc = 4
NBR:16.1.1.203 is not eligible to write, no non(-1) fd yet
Errors : Process socket event has an invalid fd to work on
```

Related Commands

Command	Description
debug tgrep error	Turns on debugging for any errors in functioning.
debug tgrep events	Turns on debugging for main events occurring throughout the subsystem.
debug tgrep fsm	Turns on debugging for FSM activity.
debug tgrep messages	Turns on debugging for the movement of TGREP messages.
debug tgrep msgdump	Turns on debugging for the dump of the details of TGREP messages.
debug tgrep timer-event	Turns on debugging for events that are related to the timer.
debug tgrep timers	Turns on debugging for timer activity.
debug tgrep tripr	Turns on debugging for the TRIP Reporter.
debug voip eddri	Turns on debugging for the EDDRI.

debug tgrep messages

To turn on debugging for movement of Telephony Gateway Registration Protocol (TGREP) messages, use the **debug tgrep messages** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug tgrep messages

no debug tgrep messages

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(1)	This command was introduced.

Usage Guidelines

There is always a performance penalty when using **debug** commands.

Examples

The following shows sample output from the **debug tgrep messages** command:

```
tgrep-gw(config-tgrep)#Received an OPEN NBR:14.1.1.210
----------------------- OPEN DUMP BEGINS -----------------------
 0x1 0x0 0x0 0xFFFFFFB4 0x0
 0x0 0x0 0x19 0x0 0x0
 0x45 0x67 0x0 0x0
        Version    :1
        Hold Time    :180
        My ITAD      :25
        TRIP ID      :17767
        No optional parameters -- hence all route types supported.
        Send-Recv capability in effect
----------------------- OPEN DUMP ENDS -----------------------
```

After the dump occurs, the TRGREP messages are displayed. In this case, keepalive messages are being received by this gateway.

```
Enqueued a Keepalive for NBR:14.1.1.210
Received an KEEPALIVE NBR:14.1.1.210
Received Keepalive for NBR:14.1.1.210
Received an KEEPALIVE NBR:14.1.1.210
```

Related Commands

Command	Description
debug tgrep error	Turns on debugging for any errors in functioning.
debug tgrep events	Turns on debugging for main events occurring throughout the subsystem.
debug tgrep fsm	Turns on debugging for FSM activity.
debug tgrep io	Turns on debugging for detailed socket level activities.
debug tgrep msgdump	Turns on debugging for the dump of the details of TGREP messages.
debug tgrep timer-event	Turns on debugging for events that are related to the timer.
debug tgrep timers	Turns on debugging for timer activity.
debug tgrep tripr	Turns on debugging for the TRIP Reporter.
debug voip eddri	Turns on debugging for the EDDRI.

debug tgrep msgdump

To turn on debugging for the dump of the details of Telephony Gateway Registration Protocol (TGREP) messages, use the **debug tgrep msgdump** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug tgrep msgdump

no debug tgrep msgdump

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(1)	This command was introduced.

Usage Guidelines

There is always a performance penalty when using **debug** commands.

Examples

The following shows sample output from the **debug tgrep msgdump** command:

```
tgrep-gw-1-02#Received an KEEPALIVE NBR:14.1.1.210
++++++++++++++++++++++++++++++++++++
 TMSG datagramstart : 0x69188648, 150 bytes
++++++++++++++++++++++++++++++++++++++++
 0x0 0xFFFFFF96 0x2 0x0 0x1
 0x0 0x0 0x0 0x2 0x0
 0x9 0x0 0x5 0x0 0x0
 0x0 0x3 0x6D 0x63 0x69
 0x0 0x3 0x0 0x6 0x0
 0x0 0x4 0xFFFFFFD2 0x0 0x0
 0x0 0x4 0x0 0x6 0x2
 0x1 0x0 0x0 0x4 0xFFFFFFD2
 0x0 0x5 0x0 0x6 0x2
 0x1 0x0 0x0 0x4 0xFFFFFFD2
 0x0 0x7 0x0 0x4 0x0
 0x0 0x0 0x5 0x0 0xF
 0x0 0x4 0x0 0x0 0x0
 0x16 0x0 0x10 0x0 0x4
 0x0 0x0 0x0 0x17 0x0
 0x11 0x0 0x8 0x0 0x0
 0x0 0x74 0x0 0x0 0x0
 0x7B 0x0 0x12 0x0 0x3C
 0x0 0x4 0x31 0x31 0x32
 0x38 0x0 0x4 0x31 0x31
 0x32 0x37 0x0 0x4 0x31
 0x31 0x32 0x36 0x0 0x4
 0x31 0x31 0x32 0x35 0x0
```

```
 0x4 0x31 0x31 0x32 0x34
 0x0 0x4 0x31 0x31 0x32
 0x33 0x0 0x4 0x31 0x31
 0x32 0x32 0x0 0x5 0x39
 0x39 0x39 0x39 0x39 0x0
 0x9 0x31 0x32 0x33 0x34
 0x35 0x36 0x37 0x38 0x39
```

After each event occurs, a dump of the message appears. The entire dump of each keepalive is being displayed.

```
-----------------------------------
Received an KEEPALIVE NBR:14.1.1.210
+++++++++++++++++++++++++++++++++++++
 TMSG datagramstart : 0x691B0CA0, 92 bytes
+++++++++++++++++++++++++++++++++++++
 0x0 0x5C 0x2 0x0 0x1
 0x0 0x0 0x0 0x2 0x0
 0xF 0x0 0x3 0x0 0x0
 0x0 0x9 0x31 0x32 0x33
 0x34 0x35 0x36 0x37 0x38
 0x39 0x0 0x3 0x0 0x6
 0x0 0x0 0x4 0xFFFFFFD2 0x0
 0x0 0x0 0x4 0x0 0x6
 0x2 0x1 0x0 0x0 0x4
 0xFFFFFFD2 0x0 0x5 0x0 0x6
 0x2 0x1 0x0 0x0 0x4
 0xFFFFFFD2 0x0 0x7 0x0 0x4
 0x0 0x0 0x0 0x5 0x0
 0xF 0x0 0x4 0x0 0x0
 0x0 0x17 0x0 0x10 0x0
 0x4 0x0 0x0 0x0 0x17
 0x0 0x11 0x0 0x8 0x0
 0x0 0x0 0x75 0x0 0x0
 0x0 0x78
-----------------------------------
+++++++++++++++++++++++++++++++++++++
 TMSG datagramstart : 0x691885EC, 150 bytes
+++++++++++++++++++++++++++++++++++++
 0x0 0xFFFFFF96 0x2 0x0 0x1
 0x0 0x0 0x0 0x2 0x0
 0x9 0x0 0x5 0x0 0x0
 0x0 0x3 0x6D 0x63 0x69
 0x0 0x3 0x0 0x6 0x0
 0x0 0x4 0xFFFFFFD2 0x0 0x0
 0x0 0x4 0x0 0x6 0x2
 0x1 0x0 0x0 0x4 0xFFFFFFD2
 0x0 0x5 0x0 0x6 0x2
 0x1 0x0 0x0 0x4 0xFFFFFFD2
 0x0 0x7 0x0 0x4 0x0
 0x0 0x0 0x5 0x0 0xF
 0x0 0x4 0x0 0x0 0x0
 0x16 0x0 0x10 0x0 0x4
 0x0 0x0 0x0 0x17 0x0
 0x11 0x0 0x8 0x0 0x0
 0x0 0x75 0x0 0x0 0x0
 0x7C 0x0 0x12 0x0 0x3C
 0x0 0x4 0x31 0x31 0x32
 0x38 0x0 0x4 0x31 0x31
 0x32 0x37 0x0 0x4 0x31
 0x31 0x32 0x36 0x0 0x4
 0x31 0x31 0x32 0x35 0x0
 0x4 0x31 0x31 0x32 0x34
 0x0 0x4 0x31 0x31 0x32
 0x33 0x0 0x4 0x31 0x31
 0x32 0x32 0x0 0x5 0x39
 0x39 0x39 0x39 0x39 0x0
 0x9 0x31 0x32 0x33 0x34
 0x35 0x36 0x37 0x38 0x39
-----------------------------------
Received an KEEPALIVE NBR:14.1.1.210
Received an KEEPALIVE NBR:14.1.1.210
```

Related Commands

Command	Description
debug tgrep error	Turns on debugging for any errors in functioning.
debug tgrep events	Turns on debugging for main events occurring throughout the subsystem.
debug tgrep fsm	Turns on debugging for FSM activity.
debug tgrep io	Turns on debugging for detailed socket level activities.
debug tgrep messages	Turns on debugging for the movement of TGREP messages.
debug tgrep timer-event	Turns on debugging for events that are related to the timer.
debug tgrep timers	Turns on debugging for timer activity.
debug tgrep tripr	Turns on debugging for the TRIP Reporter.
debug voip eddri	Turns on debugging for the EDDRI.

debug tgrep timer-event

To turn on debugging for events that are related to the timer, use the **debug tgrep timer-event** command in privileged EXEC mode. To turn off debugging, use the **no** form of this command.

debug tgrep timer-event

no debug tgrep timer-event

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(1)	This command was introduced.

Usage Guidelines

There is always a performance penalty when using **debug** commands.

Examples

The following shows sample output from the **debug tgrep timer-event** command:

```
Received a TGREP_UPD_TIMER timeout
The bulkSyncQ size is 0 at this time
The tgrepQ size is 0 at this time
Restarting the router UPD timer after expiry
Received a TGREP_UPD_TIMER timeout
The bulkSyncQ size is 0 at this time
The tgrepQ size is 0 at this time
Restarting the router UPD timer after expiry
```

The Telephony Routing over IP (TRIP) timer registers timeouts until the next event occurs. Here, the timers are reset.

```
Entering trip_reset_nbr_timers to reset timers
Starting the CONNECT timer for nbr NBR:16.1.1.202 for value of 30 seconds
Stopping hold timer and keepalive timer while resetting NBR:16.1.1.202
Received a TGREP_UPD_TIMER timeout
The bulkSyncQ size is 0 at this time
The tgrepQ size is 0 at this time
Restarting the router UPD timer after expiry
Received a TGREP_UPD_TIMER timeout
The bulkSyncQ size is 3 at this time
The tgrepQ size is 0 at this time
Restarting the router UPD timer after expiry
Received a TGREP_UPD_TIMER timeout
The bulkSyncQ size is 3 at this time
The tgrepQ size is 0 at this time
Restarting the router UPD timer after expiry
```

Timeouts are again reported until the next event.

```
Received a TGREP_UPD_TIMER timeout
The bulkSyncQ size is 3 at this time
The tgrepQ size is 0 at this time
Restarting the router UPD timer after expiry
```

Here, the TRIP neighbor is cleared, which causes the timer to reset.

```
Router#clear trip nei *
Router#Entering trip_reset_nbr_timers to reset timers
Starting the CONNECT timer for nbr NBR:16.1.1.202 for value of 30 seconds
Stopping hold timer and keepalive timer while resetting NBR:16.1.1.202
Received a TGREP_UPD_TIMER timeout
The bulkSyncQ size is 0 at this time
The tgrepQ size is 0 at this time
Restarting the router UPD timer after expiry
Received a TGREP_UPD_TIMER timeout
The bulkSyncQ size is 3 at this time
The tgrepQ size is 0 at this time
Restarting the router UPD timer after expiry
Received a TGREP_UPD_TIMER timeout
The bulkSyncQ size is 3 at this time
The tgrepQ size is 0 at this time
Restarting the router UPD timer after expiry
Received a TGREP_UPD_TIMER timeout
The bulkSyncQ size is 3 at this time
The tgrepQ size is 0 at this time
Restarting the router UPD timer after expiry
Received a TGREP_UPD_TIMER timeout
The bulkSyncQ size is 3 at this time
The tgrepQ size is 0 at this time
Restarting the router UPD timer after expiry
Received a TGREP_UPD_TIMER timeout
The bulkSyncQ size is 3 at this time
The tgrepQ size is 0 at this time
Restarting the router UPD timer after expiry
Received a TGREP_UPD_TIMER timeout
The bulkSyncQ size is 3 at this time
The tgrepQ size is 0 at this time
Restarting the router UPD timer after expiry
Received a TGREP_UPD_TIMER timeout
The bulkSyncQ size is 3 at this time
The tgrepQ size is 0 at this time
Restarting the router UPD timer after expiry
Received a TGREP_UPD_TIMER timeout
The bulkSyncQ size is 3 at this time
The tgrepQ size is 0 at this time
Restarting the router UPD timer after expiry
IO_CONNECT TIMER for nbr NBR:16.1.1.202 has expired
NBR:16.1.1.202 -Restarting the connect timer
NBR:16.1.1.202 starting the holder timer after post connect with large value
------------------------ OPEN DUMP BEGINS ------------------------
 0x1 0xFFFFFFFF 0x0 0xFFFFFFB4 0x0
 0x0 0x4 0x58 0x6 0x7
 0xFFFFFF98 0xFFFFFFA9 0x0 0xC 0x0
 0x1 0x0 0x8 0x0 0x2
 0x0 0x4 0x0 0x0 0x0
 0x3
        Version    :1
        Hold Time    :180
        My ITAD      :1112
        TRIP ID      :101161129
                 Option Paramater #1
                 Param Type: Capability
                 Length 8
                         Cap Code :Send Receive Capability
                         Cap Len  :4
                                Send Rec Cap: RCV ONLY MODE
        -->All route types supported
------------------------ OPEN DUMP ENDS ------------------------
NBR:16.1.1.202 Starting keepalive timer after writing something
```

```
NBR:16.1.1.202 Re-starting hold timer after a message is read
Received a TGREP_UPD_TIMER timeout
The bulkSyncQ size is 0 at this time
The tgrepQ size is 0 at this time
Restarting the router UPNBR:16.1.1.202 Starting keepalive timer after writing so
mething
NBR:16.1.1.202 Re-starting hold timer after a message is read
D timer after expiry
Received a TGREP_UPD_TIMER timeout
The bulkSyncQ size is 0 at this time
The tgrepQ size is 0 at this time
Restarting the router UPD timer after expiry
Received a TGREP_UPD_TIMER timeout
The bulkSyncQ size is 0 at this time
The tgrepQ size is 0 at this time
Restarting the router UPD timer after expiry
Received a TGREP_UPD_TIMER timeout
The bulkSyncQ size is 0 at this time
The tgrepQ size is 0 at this time
Restarting the router UPD timer after expiry
Received a TGREP_UPD_TIMER timeout
The bulkSyncQ size is 0 at this time
The tgrepQ size is 0 at this time
Restarting the router UPD timer after expiry
Received a TGREP_UPD_TIMER timeout
The bulkSyncQ size is 0 at this time
The tgrepQ size is 0 at this time
Restarting the router UPD timer after expiry
Received a TGREP_UPD_TIMER timeout
The bulkSyncQ size is 0 at this time
The tgrepQ size is 0 at this time
Restarting the router UPD timer after expiry
Received a TGREP_UPD_TIIO_CONNECT TIMER for nbr NBR:16.1.1.202 has expired
NBR:16.1.1.202 -Stopping the connect timer, no need anymore
MER timeout
The bulkSyncQ size is 0 at this time
The tgrepQ size is 0 at this time
Restarting the router UPD timer after expiry
Received a TGREP_UPD_TIMER timeout
The bulkSyncQ size is 0 at this time
The tgrepQ size is 0 at this time
Restarting the router UPD timer after expiry
NBR:16.1.1.202 Re-starting hold timer after a message is read
Received a TGREP_UPD_TIMER timeout
The bulkSyncQ size is 0 at this time
The tgrepQ size is 0 at this time
Restarting the router UPD timer after expiry
The tgrepQ size is 0 at this time
Restarting the router UPD timer after expiry
```

Related Commands

Command	Description
debug tgrep error	Turns on debugging for any errors in functioning.
debug tgrep events	Turns on debugging for main events occurring throughout the subsystem.
debug tgrep fsm	Turns on debugging for FSM activity.
debug tgrep io	Turns on debugging for detailed socket level activities.
debug tgrep messages	Turns on debugging for the movement of TGREP messages.

Command	Description
debug tgrep msgdump	Turns on debugging for the dump of the details of TGREP messages.
debug tgrep timers	Turns on debugging for timer activity.
debug tgrep tripr	Turns on debugging for the TRIP Reporter.
debug voip eddri	Turns on debugging for the EDDRI.

debug tgrep timers

To turn on debugging for detailed socket level activities, use the **debug tgrep timers** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug tgrep timers

no debug tgrep timers

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(1)	This command was introduced.

Usage Guidelines

There is always a performance penalty when using **debug** commands.

Examples

The following shows sample output from the **debug tgrep timers** command:

```
tgrep-gw-1-02#Received a TGREP_UPD_TIMER timeout
The bulkSyncQ size is 0 at this time
The tgrepQ size is 0 at this time
Restarting the router UPD timer after expiry
```

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

Table 7: debug tgrep timers Field Descriptions

Field	Description
Received a TGREP_UPD_TIMER timeout	This indicates that a timeout was received.
The bulkSyncQ size is 0 at this time	This indicates the size of the bulk sync queue.
The tgrepQ size is 0 at this time	This indicates the size of the TGREP queue.
Restarting the router UPD timer after expiry	This indicates that the timer has been reset.

Related Commands

Command	Description
debug tgrep error	Turns on debugging for any errors in functioning.
debug tgrep events	Turns on debugging for main events occurring throughout the subsystem.
debug tgrep fsm	Turns on debugging for FSM activity.
debug tgrep io	Turns on debugging for detailed socket level activities.
debug tgrep messages	Turns on debugging for the movement of TGREP messages.
debug tgrep msgdump	Turns on debugging for the dump of the details of TGREP messages.
debug tgrep timer-event	Turns on debugging for events that are related to the timer.
debug tgrep tripr	Turns on debugging for the TRIP Reporter.
debug voip eddri	Turns on debugging for the EDDRI.

debug tgrep tripr

To turn on debugging from the Telephony Routing over IP (TRIP) Reporter (TRIPR), use the **debug tgrep tripr** command in privileged EXEC mode. To turn off debugging, use the **no** form of this command.

debug tgrep tripr

no debug tgrep tripr

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(1)	This command was introduced.

Usage Guidelines

There is always a performance penalty when using **debug** commands.

A watched queue is used to inform the TRIPR process about changes in any of the interesting attributes of dial peer that potentially could trigger TRIP update. A dial peer attribute change manifests into a prefix attribute change and is deposited into the watched queue of TRIPR by the Event Dispatcher. The trunk group system also does the same.

Examples

The following shows sample output from the **debug tgrep tripr** command:

```
20:51:11: tripr_build_triprtr_prefix_destination_ev : got the ev id 1 reason 64 num_prefix
 1 advertise 0x2prefix 1128 addrFam 4
20:51:11: tripr_build_triprtr_prefix_destination_ev ac 22 tc 23 ac_avg 22
20:51:11: tripr_build_triprtr_prefix_destination_ev csr success 0 total 0
20:51:11:
20:51:11: --------------------------------
20:51:11: attrib 0x4002
20:51:11: ******* REACHABLE ROUTE ******
20:51:11: TRIP_AF_E164 1128
20:51:11:  ac: 22
20:51:11:
20:51:11: ======================================
20:51:11: tripr_build_triprtr_prefix_destination_ev : got the ev id 1 reason 64 num_prefix
 1 advertise 0x27prefix 123456789 addrFam 4
20:51:11: tripr_build_triprtr_prefix_destination_ev ac 22 tc 23 ac_avg 22
20:51:11: tripr_build_triprtr_prefix_destination_ev csr success 117 total 120
20:51:11:  tg mci cc mci
20:51:11: tripr_build_triprtr_prefix_destination_ev tg mci cic 0 carrier mci
20:51:11:
20:51:11: --------------------------------
20:51:11: attrib 0x1C002
20:51:11: ******* REACHABLE ROUTE ******
```

```
20:51:11: TRIP_AF_E164 123456789
20:51:11:  csr: tot 120 succ 117
20:51:11:  ac: 22tc: 23
20:51:11:
20:51:11: ====================================
20:51:11: tripr_build_triprtr_prefix_destination_ev : got the ev id 1 reason 64 num_prefix
 1 advertise 0x27prefix 99999 addrFam 4
20:51:11: tripr_build_triprtr_prefix_destination_ev ac 22 tc 23 ac_avg 22
20:51:11: tripr_build_triprtr_prefix_destination_ev csr success 0 total 0
20:51:11:  tg mci cc mci
20:51:11: tripr_build_triprtr_prefix_destination_ev tg mci cic 0 carrier mci
20:51:11:
20:51:11: --------------------------------
20:51:11: attrib 0x1C002
20:51:11: ******* REACHABLE ROUTE ******
20:51:11: TRIP_AF_E164 99999
20:51:11:  csr: tot 0 succ 0
20:51:11:  ac: 22tc: 23
20:51:11:
20:51:11: ====================================
```

The table below describes the significant fields in the display.

Table 8: debug tgrep tripr Field Descriptions

Field	Description
ev id	This field can contain the following entries: • 1--Prefix regular event • 2--Trunk group regular event • 3--Carrier regular event • 4--Prefix sync event • 5--Trunk group sync event • 6--Carrier sync event • 7--Null sync event
reason: (for a prefix family event)	This field can contain the following entries: • 1--Prefix down • 2--Prefix up • 4--Prefix trunk group attribute changed • 8--Prefix available circuits changed • 16--Prefix total circuits changed • 32--Prefix CSR changed • 64--Prefix AC interesting point • 128--Prefix carrier attributes changed • 256--Prefix stop advertise configured • 512--Prefix start advertise configured

Field	Description
reason: (for a trunk group family event)	This field can contain the following entries: • 1--Trunk group down • 2--Trunk group up • 4--Trunk group prefix attribute changed • 8--Trunk group available circuits changed • 16--Trunk group total circuits changed • 32--Trunk group CSR changed • 64--Trunk group AC interesting point • 128--Trunk group stop advertise configured • 256--Trunk group start advertise configured
reason: (for a carrier family event)	This field can contain the following entries: • 1--Carrier down • 2--Carrier up • 4--Carrier prefix attribute changed • 8--Carrier available circuits changed • 16--Carrier total circuits changed • 32--Carrier CSR changed • 64--Carrier AC interesting point • 128--Carrier stop advertise configured • 256--Carrier start advertise configured

Related Commands

Command	Description
debug tgrep error	Turns on debugging for any errors in functioning.
debug tgrep events	Turns on debugging for main events occurring throughout the subsystem.
debug tgrep fsm	Turns on debugging for FSM activity.
debug tgrep io	Turns on debugging for detailed socket level activities.

Command	Description
debug tgrep messsages	Turns on debugging for the movement of TGREP messages.
debug tgrep msgdump	Turns on debugging for the dump of the details of TGREP messages.
debug tgrep timer-event	Turns on debugging for events that are related to the timer.
debug tgrep timers	Turns on debugging for timer activity.
debug voip eddri	Turns on debugging for the EDDRI.

debug tgrm

To display debugging messages for all trunk groups, use the **debug tgrm** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug tgrm [**all**| **default**| **detail**| **error** [**call [informational]**| **software [informational]**]| **function**| **inout**| **service**]

no debug tgrm

Syntax Description

all	(Optional) Displays all TGRM debugging messages.
default	(Optional) Displays detail, error, and inout information. This option also runs if no keywords are added.
detail	(Optional) Displays non-inout information related to call processing, such as call updates or call acceptance checking.
error	(Optional) Displays TGRM error messages.
call	(Optional) Displays call processing errors.
informational	(Optional) Displays minor errors and major errors. Without the **informational** keyword, only major errors are displayed.
software	(Optional) Displays software errors.
function	(Optional) Displays TGRM functions.
inout	(Optional) Displays information from the functions that form the external interfaces of TGRM to other modules or subsystems.
service	(Optional) Displays TGRM services.

Command Default

Debugging is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.1(3)T	This command was introduced.
12.2(2)XB1	This command was implemented on the Cisco AS5850 platform.
12.2(11)T	This command was implemented on the Cisco AS5850 platform.
12.3(8)T	The **all**, **default**, **detail**, **error**, **call**, **informational**, **software**, **function**, **inout**, and **service** keywords were added to this command.

Usage Guidelines

Because the **debug tgrm** command causes a large amount of messages to be generated, router performance can be affected.

Caution The **debug tgrm** command can impact the performance of your router. This command should only be used during low traffic periods.

Examples

The following is sample output from the **debug tgrm all** command for an incoming CAS call on a trunk group that is rejected because of the **max-calls** command:

```
Router# debug tgrm all

03:53:56: //-1/xxxxxxxxxxxx/TGRM/tgrm_accept_call:
    Timeslot=11, CallType=Voice, CallDirection=Incoming, Slot=2, SubUnit=1, Port=1,
DS0-Group=1
03:53:56: //-1/xxxxxxxxxxxx/TGRM/tgrm_tg_member_core:
03:53:56: //-1/xxxxxxxxxxxx/TGRM/tgrm_tg_trunk_member:
03:53:56: //-1/xxxxxxxxxxxx/TGRM/tgrm_trunk_equal:
03:53:56: //-1/xxxxxxxxxxxx/TGRM/tgrm_tg_trunk_member:
03:53:56: //-1/xxxxxxxxxxxx/TGRM/tgrm_trunk_equal:
03:53:56: //-1/xxxxxxxxxxxx/TGRM/tgrm_tg_member_trunk_internal:
03:53:56: //-1/xxxxxxxxxxxx/TGRM/tgrm_tg_member_core:
03:53:56: //-1/xxxxxxxxxxxx/TGRM/tgrm_tg_trunk_member:
03:53:56: //-1/xxxxxxxxxxxx/TGRM/tgrm_trunk_equal:
03:53:56: //-1/xxxxxxxxxxxx/TGRM/tgrm_tg_trunk_member:
03:53:56: //-1/xxxxxxxxxxxx/TGRM/tgrm_trunk_equal:
03:53:56: //-1/xxxxxxxxxxxx/TGRM/tgrm_trunk_equal:
03:53:56: //-1/xxxxxxxxxxxx/TGRM/tgrm_trunk_channel_active:
    Trunk=2/1:1 (TG 211), Timeslot=11, CallType=Voice,
    CallDirection=Incoming
03:53:56: //-1/xxxxxxxxxxxx/TGRM/tgrm_trunk_channel_delete:
03:53:56: //-1/xxxxxxxxxxxx/TGRM/tgrm_trunk_channel_delete_queue:
03:53:56: //-1/xxxxxxxxxxxx/TGRM/tgrm_tg_call_count_update:
    CallDirection=Incoming, Increment call count
03:53:56: //-1/xxxxxxxxxxxx/TGRM/tgrm_tg_call_count_update_no_crm:
03:53:56: //-1/xxxxxxxxxxxx/TGRM/tgrm_tg_call_count_update_no_crm:
    CountType=TGRM_COUNT_VOICE, CallDirection=Incoming, Increment call count
    Updated values: CallCount=1, FreeTimeslots=23
03:53:56: //-1/xxxxxxxxxxxx/TGRM/tgrm_tg_call_count_update_crm:
    CallType=Voice, CallDirection=Incoming, Increment the call count
03:53:56: //-1/xxxxxxxxxxxx/TGRM/tgrm_tg_info:
03:53:56: //-1/xxxxxxxxxxxx/TGRM/tgrm_tg_info_internal:
03:53:56: //-1/xxxxxxxxxxxx/TGRM/tgrm_tg_info_internal:
    TG 211 found
```

```
03:53:56: //-1/xxxxxxxxxxxx/TGRM/tgrm_tg_info:
    TG 211 found
03:53:56: //-1/xxxxxxxxxxxx/TGRM/tgrm_tg_status:
03:53:56: //-1/xxxxxxxxxxxx/TGRM/tgrm_tg_info_internal:
03:53:56: //-1/xxxxxxxxxxxx/TGRM/tgrm_tg_info_internal:
    TG 211 found
03:53:56: //-1/xxxxxxxxxxxx/TGRM/tgrm_allow_call:
03:53:56: //-1/xxxxxxxxxxxx/TGRM/tgrm_allow_call:
    TG 211; CallType=Voice CallDirection=Incoming
03:53:56: //-1/xxxxxxxxxxxx/TGRM/tgrm_allow_call:
    Call denied; CallType=Voice CallDirection=Incoming; MaxAllowed=0 Current=1
03:53:56: //-1/xxxxxxxxxxxx/TGRM/tgrm_accept_call:
    Call Rejected; Reason - Maximum voice calls exceeded
03:53:56: //-1/xxxxxxxxxxxx/TGRM/tgrm_tg_info:
03:53:56: //-1/xxxxxxxxxxxx/TGRM/tgrm_tg_info_internal:
03:53:56: //-1/xxxxxxxxxxxx/TGRM/tgrm_tg_info_internal:
    TG 211 found
03:53:56: //-1/xxxxxxxxxxxx/TGRM/tgrm_tg_info:
    TG 211 found
```

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

Table 9: debug tgrm all Field Descriptions

Field	Description
//-1/xxxxxxxxxxxx/TGRM/ tgrm_accept_call:	The format of this message is //callid/GUID/module name/function name: • CallEntry ID is -1. This indicates that a call leg has not been identified. • GUID is xxxxxxxxxxxx. This indicates that the GUID information is unavailable. • TGRMis the module name. • Thetgrm_accept_callfield shows that the trunk group is accepting a call.
Timeslot=11, CallType=Voice, CallDirection=Incoming, Slot=2, SubUnit=1, Port=1, DS0-Group=1	Shows information about the call, including timeslot, call type and direction, and port information.
tgrm_trunk_channel_active: **Trunk=2/1:1 (TG 211), Timeslot=11, CallType=Voice,** **CallDirection=Incoming**	Shows information for the active trunk group, including the port, timeslot, and call type and direction.
tgrm_tg_call_count_update: CallDirection=Incoming, Increment call count	Indicates that the call counter for the trunk group has been incremented.
tgrm_tg_call_count_update_no_crm: CountType=TGRM_COUNT_VOICE, CallDirection=Incoming, Increment call count Updated values: CallCount=1, FreeTimeslots=23	Indicates that the call counter for the trunk group has been updated outside of the Carrier Resource Manager (CRM). This field contains more data than a call counter increment message that uses the CRM.

Field	Description
tgrm_allow_call: **TG 211; CallType=Voice CallDirection=Incoming**	Shows that a call was allowed on the 2/1:1 trunk.
tgrm_allow_call: **Call denied; CallType=Voice CallDirection=Incoming; MaxAllowed=0 Current=1**	Shows that a call on the trunk group was denied.
tgrm_accept_call: **Call Rejected; Reason - Maximum voice calls exceeded**	Shows that a call was rejected on this trunk group due to a maximum number of voice calls being received.

debug tiff reader

To display output about the off-ramp TIFF reader, use the **debug tiff reader** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug tiff reader

no debug tiff reader

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(4)T	This command was introduced.

Examples

The following debug example displays information about the off-ramp TIFF reader.

```
Router# debug tiff reader
*Jan 1 18:59:13.683: tiff_reader_data_handler: new context
*Jan 1 18:59:13.683: tiff_reader_data_handler: resolution: standard
*Jan 1 18:59:13.683: tiff_reader_data_handler: buffer size: 1524i>> tiff_reader_engine()
ENGINE_START/DONE gggg(pl 616E9994)
*Jan 1 18:59:13.691: tiff_reader_data_handler: buffer size: 1524
*Jan 1 18:59:13.699: tiff_reader_data_handler: buffer size: 1524i>> tiff_reader_engine()
case FAX_EBUFFER pppp(pl 616E9994)
*Jan 1 18:59:13.703: tiff_reader_put_buffer: START_OF_FAX_PAGEi>> tiff_reader_engine() case
 FAX_EBUFFER gggg
*Jan 1 18:59:13.711: tiff_reader_data_handler: buffer size: 1524
*Jan 1 18:59:13.719: tiff_reader_data_handler: buffer size: 1524i>> tiff_reader_engine()
case FAX_EBUFFER pppp(pl 616E9994)
i>> tiff_reader_engine() case FAX_EBUFFER gggg
*Jan 1 18:59:13.727: tiff_reader_data_handler: buffer size: 1524i>> tiff_reader_engine()
case FAX_EBUFFER pppp(pl 616E9994)
i>> tiff_reader_engine() case FAX_EBUFFER gggg
*Jan 1 18:59:13.735: tiff_reader_data_handler: buffer size: 1524
*Jan 1 18:59:13.743: tiff_reader_data_handler: buffer size: 1524i>> tiff_reader_engine()
case FAX_EBUFFER pppp(pl 616E9994)
i>> tiff_reader_engine() case FAX_EBUFFER gggg
*Jan 1 18:59:13.751: tiff_reader_data_handler: buffer size: 1524
*Jan 1 18:59:13.759: tiff_reader_data_handler: buffer size: 1524i>> tiff_reader_engine()
case FAX_EBUFFER pppp(pl 616E9994)
i>> tiff_reader_engine() case FAX_EBUFFER gggg
*Jan 1 18:59:13.767: tiff_reader_data_handler: buffer size: 1524
*Jan 1 18:59:13.775: tiff_reader_data_handler: buffer size: 1524i>> tiff_reader_engine()
case FAX_EBUFFER pppp(pl 616E9994)
i>> tiff_reader_engine() case FAX_EBUFFER gggg
*Jan 1 18:59:13.787: tiff_reader_data_handler: buffer size: 1524
*Jan 1 18:59:13.795: tiff_reader_data_handler: buffer size: 1524i>> tiff_reader_engine()
case FAX_EBUFFER pppp(pl 616E9994)
```

```
i>> tiff_reader_engine() case FAX_EBUFFER gggg
*Jan 1 18:59:13.803: tiff_reader_data_handler: buffer size: 1524
*Jan 1 18:59:13.811: tiff_reader_data_handler: buffer size: 1524i>> tiff_reader_engine()
case FAX_EBUFFER pppp(pl 616E9994)
i>> tiff_reader_engine() case FAX_EBUFFER gggg
*Jan 1 18:59:13.819: tiff_reader_data_handler: buffer size: 1524
*Jan 1 18:59:13.827: tiff_reader_data_handler: buffer size: 1524i>> tiff_reader_engine()
case FAX_EBUFFER pppp(pl 616E9994)
i>> tiff_reader_engine() case FAX_EBUFFER gggg
*Jan 1 18:59:13.835: tiff_reader_data_handler: buffer size: 1524
*Jan 1 18:59:13.843: tiff_reader_data_handler: buffer size: 1524i>> tiff_reader_engine()
case FAX_EBUFFER pppp(pl 616E9994)
i>> tiff_reader_engine() case FAX_EBUFFER gggg
*Jan 1 18:59:13.851: tiff_reader_data_handler: buffer size: 1524i>> tiff_reader_engine()
case FAX_EBUFFER pppp(pl 616E9994)
i>> tiff_reader_engine() case FAX_EBUFFER gggg
*Jan 1 18:59:13.863: tiff_reader_data_handler: buffer size: 1524
*Jan 1 18:59:13.871: tiff_reader_data_handler: buffer size: 1524i>> tiff_reader_engine()
case FAX_EBUFFER pppp(pl 616E9994)
i>> tiff_reader_engine() case FAX_EBUFFER gggg
*Jan 1 18:59:13.879: tiff_reader_data_handler: buffer size: 1524
*Jan 1 18:59:13.887: tiff_reader_data_handler: buffer size: 1524i>> tiff_reader_engine()
case FAX_EBUFFER pppp(pl 616E9994)
i>> tiff_reader_engine() case FAX_EBUFFER gggg
*Jan 1 18:59:13.895: tiff_reader_data_handler: buffer size: 1524
*Jan 1 18:59:13.903: tiff_reader_data_handler: buffer size: 1524i>> tiff_reader_engine()
case FAX_EBUFFER pppp(pl 616E9994)
i>> tiff_reader_engine() case FAX_EBUFFER gggg
*Jan 1 18:59:13.907: tiff_reader_data_handler: buffer size: 311i>> tiff_r_finish()
END_OF_FAX_PAGE pppp
*Jan 1 18:59:13.907: tiff_reader_put_buffer: END_OF_FAX_PAGE. Dial now ...if not in progress
*Jan 1 18:59:13.907: tiff_reader_data_handler: END_OF_DATA
*Jan 1 18:59:13.907: tiff_reader_data_handler: BUFF_END_OF_PART
*Jan 1 18:59:13.907: tiff_reader_data_handler: Dispose context
```

Related Commands

Command	Description
debug tiff writer	Displays output about the on-ramp TIFF writer.

debug tiff writer

To display output about the on-ramp TIFF writer, use the **debug tiff writer** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug tiff writer

debug tiff writer

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(4)T	This command was introduced.

Examples

The following debug example shows information about the off-ramp TIFF writer.

```
Router# debug tiff writer
*Jan 1 18:54:59.419: tiff_writer_data_process: START_OF_CONNECTION
18:55:10: %FTSP-6-FAX_CONNECT: Reception
*Jan 1 18:55:14.903: tiff_writer_data_process: START_OF_FAX_PAGE
*Jan 1 18:55:14.903: tiff_writer_data_process: tiff file created = 2000:01:01 18:55:14
18:55:21: %FTSP-6-FAX_DISCONNECT: Reception
*Jan 1 18:55:19.039: tiff_writer_data_process: END_OF_CONNECTION or ABORT_CONNECTION
*Jan 1 18:55:19.039: tiff_writer_put_buffer: END_OF_FAX_PAGE
*Jan 1 18:55:19.039: send TIFF_PAGE_READY
*Jan 1 18:55:19.039: send TIFF_PAGE_READY
18:55:21: %LINK-3-UPDOWN: Interface Serial2:0, changed state to down
```

Related Commands

Command	Description
debug tiff reader	Displays output about the on-ramp TIFF reader.

debug time-range ipc

To enable debugging output for monitoring the time-range ipc messages between the Route Processor and the line card, use the **debug time-range ipc**command inprivileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug time-range ipc

no debug time-range ipc

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(2)T	This command was introduced.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.

Examples

The following is sample output from the **debug time-range ipc**command. In the following example, the time ranges sent to the line card are monitored:

```
Router# debug time-range ipc
00:14:19:TRANGE-IPC:Sent Time-range t1 ADD to all slots
00:15:22:TRANGE-IPC:Sent Time-range t1 ADD to all slots
```

In the following example, the time ranges deleted from the line card are monitored:

```
Router# debug time-range ipc
00:15:42:TRANGE-IPC:Sent Time-range t1 DEL to all slots
00:15:56:TRANGE-IPC:Sent Time-range t1 DEL to all slots
```

Related Commands

Command	Description
show time-range ipc	Displays the statistics about the time-range ipc messages between the Route Processor and line card.

debug token ring

To display messages about Token Ring interface activity, use the **debug token ring** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug token ring

no debug token ring

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

This command reports several lines of information for each packet sent or received and is intended for low traffic, detailed debugging.

The Token Ring interface records provide information regarding the current state of the ring. These messages are only displayed when the **debug token events** command is enabled.

The **debug token ring** command invokes verbose Token Ring hardware debugging. This includes detailed displays as traffic arrives and departs the unit.

Caution It is best to use this command only on routers and bridges with light loads.

Examples

The following is sample output from the **debug token ring** command:

```
Router# debug token ring
TR0: Interface is alive, phys. addr 5000.1234.5678
TR0:  in: MAC: acfc: 0x1105 Dst: c000.ffff.ffff Src: 5000.1234.5678 bf: 0x45
TR0:  in:   riflen 0, rd_offset 0, llc_offset 40
TR0: out: MAC: acfc: 0x0040 Dst: 5000.1234.5678 Src: 5000.1234.5678 bf: 0x00
TR0: out: LLC: AAAA0300 00009000 00000100 AAC00000 00000802 50001234 ln: 28
TR0:  in: MAC: acfc: 0x1140 Dst: 5000.1234.5678 Src: 5000.1234.5678 bf: 0x09
TR0:  in: LLC: AAAA0300 00009000 00000100 AAC0B24A 4B4A6768 74732072 ln: 28
TR0:   in:   riflen 0, rd_offset 0, llc_offset 14
TR0: out: MAC: acfc: 0x0040 Dst: 5000.1234.5678 Src: 5000.1234.5678 bf: 0x00
TR0: out: LLC: AAAA0300 00009000 00000100 D1D00000 FE11E636 96884006 ln: 28
TR0:  in: MAC: acfc: 0x1140 Dst: 5000.1234.5678 Src: 5000.1234.5678 bf: 0x09
TR0:  in: LLC: AAAA0300 00009000 00000100 D1D0774C 4DC2078B 3D000160 ln: 28
TR0:  in:   riflen 0, rd_offset 0, llc_offset 14
TR0: out: MAC: acfc: 0x0040 Dst: 5000.1234.5678 Src: 5000.1234.5678 bf: 0x00
TR0: out: LLC: AAAA0300 00009000 00000100 F8E00000 FE11E636 96884006 ln: 28
```

The table below describes the significant fields shown in the second line of output.

Table 10: debug token ring Field Descriptions

Message	Description
TR0:	Name of the interface associated with the Token Ring event.

Message	Description
in:	Indication of whether the packet was input to the interface (in) or output from the interface (out).
MAC:	Type of packet, as follows: • MAC--Media Access Control • LLC--Link Level Control
acfc: 0x1105	Access Control, Frame Control bytes, as defined by the IEEE 802.5 standard.
Dst: c000.ffff.ffff	Destination address of the frame.
Src: 5000.1234.5678	Source address of the frame.
bf: 0x45	Bridge flags for internal use by technical support staff.

The table below describes the significant fields shown in the third line of output.

Table 11: debug token ring Field Descriptions

Message	Description
TR0:	Name of the interface associated with the Token Ring event.
in:	Indication of whether the packet was input to the interface (in) or output from the interface (out).
riflen 0	Length of the routing information field (RIF) in bytes.
rd_offset 0	Offset (in bytes) of the frame pointing to the start of the RIF field.
llc_offset 40	Offset in the frame pointing to the start of the LLC field.

The table below describes the significant fields shown in the fifth line of output.

Table 12: debug token ring Field Descriptions

Message	Description
TR0:	Name of the interface associated with the Token Ring event.

Message	Description
out:	Indication of whether the packet was input to the interface (in) or output from the interface (out).
LLC:	Type of frame, as follows: • MAC--Media Access Control • LLC--Link Level Control
AAAA0300	This and the octets that follow it indicate the contents (hex) of the frame.
ln: 28	The length of the information field (in bytes).

debug topology

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

debug topology {**accounting**| **all**| **cache**| **db**| **event**| **ha**| **interface**| **locking**| **sb**| **state**| **topoid**| **vrf**}

no debug topology {**accounting**| **all**| **cache**| **db**| **event**| **ha**| **interface**| **locking**| **sb**| **state**| **topoid**| **vrf**}

Syntax Description

accounting	Enables debugging for topology accounting.
all	Enables debugging for all topology routing events.
cache	Enables debugging for topology ID cache activity.
db	Enables debugging for topology DB events.
event	Enables debugging for topology notification events.
ha	Enables debugging for topology High Availability (HA) events.
interface	Enables debugging for topology interface association.
locking	Enables debugging for topology client locking activity.
sb	Enables debugging for topology sub-block.
state	Enables debugging for topology state change events.
topoid	Enables debugging for topology ID management events.
vrf	Enables debugging for topology VRF association.

Command Default

Debugging output for topology related events is disabled.

Command Modes

Privileged EXEC (#)

Command History

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

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

Examples

The following example shows how to enable debugging for topology HA events:

```
Device# debug topology ha
```

Related Commands

Command	Description
show topology	Displays status and configuration information for topologies configured with MTR.

debug track

To display tracking activity for tracked objects, use the **debug track** command in privileged EXEC mode. To turn off output, use the **no** form of this command.

debug track

no debug track

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(15)T	This command was introduced.
12.3(8)T	The output was enhanced to include the track-list objects.
12.2(25)S	This command was integrated into Cisco IOS Release 12.2(25)S.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
Cisco IOS XE Release 2.1	This command was integrated into Cisco IOS XE Release 2.1.

Usage Guidelines

Use this command to display activity for objects being tracked by the tracking process. These objects can be the state of IP routing, the line-protocol state of an interface, the IP-route reachability, and the IP-route threshold metric.

Examples

The following example shows that object number 100 is being tracked and that the state of IP routing on Ethernet interface 0/2 is down:

```
Router# debug track
Feb 26 19:56:23.247:Track:100 Adding interface object
Feb 26 19:56:23.247:Track:Initialise
Feb 26 19:56:23.247:Track:100 New interface Et0/2, ip routing Down
Feb 26 19:56:23.247:Track:Starting process
```

The following example shows that object number 100 is being tracked and that the state of IP routing on Ethernet interface 0/2 has changed and is back up:

```
Router# debug track
Feb 26 19:56:41.247:Track:100 Change #2 interface Et0/2, ip routing Down->Up
00:15:07:%LINK-3-UPDOWN:Interface Ethernet0/2, changed state to up
00:15:08:%LINEPROTO-5-UPDOWN:Line protocol on Interface Ethernet0/2, changed state to up
```

Related Commands

Command	Description
show track	Displays tracking information.

debug trifecta

To enable debugging for errors that pertain to major, severe, or minor events in the processes of ASA SM, use the **debug trifecta** command in the global configuration mode. Run the command from SP of Supervisor. To disable the debugging, use the **no** form of this command.

debug trifecta {**all**| **major**| **minor**| **severe**}

no debug trifecta {**all**| **major**| **minor**| **severe**}

Syntax Description

all	Displays the output for major, minor, and severe events in the processes for ASA SM
major	Displays the output for major events in the processes for ASA SM
minor	Displays the output for minor events in the processes for ASA SM
severe	Displays the output for severe errors in ASA SM processes such as the inability to allocate memory, or create processes

Command Default

None

Command Modes

Global configuration

Command History

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

Usage Guidelines

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

Examples

The sample output for the command is as follows:

```
debug trifecta all
Router-sp#debug trifecta all
TRIFECTA severe debugging is on
TRIFECTA major debugging is on
TRIFECTA minor debugging is on
TRIFECTA debug debugging is on
Router-sp#
Jan 24 20:06:34.463 IST: SP: process_trifecta_msgs(): Polling TIMEINFO slot 2.
Jan 24 20:06:34.463 IST: SP: send_time_zone_info_to_x86(): sending time zone info to X86
slot 2
Jan 24 20:07:24.467 IST: SP: process_trifecta_msgs(): Polling TIMEINFO slot 2.
```

```
Jan 24 20:07:24.467 IST: SP: send_time_zone_info_to_x86(): sending time zone info to X86
slot 2
Jan 24 20:08:14.471 IST: SP: process_trifecta_msgs(): Polling TIMEINFO slot 2.
Jan 24 20:08:14.471 IST: SP: send_time_zone_info_to_x86(): sending time zone info to X86
slot 2
Jan 24 20:09:04.475 IST: SP: process_trifecta_msgs(): Polling TIMEINFO slot 2.
Jan 24 20:09:04.475 IST: SP: send_time_zone_info_to_x86(): sending time zone info to X86
slot 2
Jan 24 20:09:54.479 IST: SP: process_trifecta_msgs(): Polling TIMEINFO slot 2.
Jan 24 20:09:54.479 IST: SP: send_time_zone_info_to_x86(): sending time zone info to X86
slot 2
```

debug tsp

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

To display information about the telephony service provider (TSP), use the **debug tsp**command in privileged EXEC mode. To disable debugging output, use the no form of this command.

debug tsp {**all**| **call**| **error**| **port**}

no debug tsp {**all**| **call**| **error**| **port**}

Syntax Description

all	Enables all TSP debugging (except statistics).
call	Enables call debugging.
error	Error debugging.
port	Port debugging.

Command Default

Disabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(6)T	This command was introduced.
12.3(8)T	This command was replaced by the **debug voip tsp** command.

Examples

The following shows sample output from the **debug tsp all** command:

```
Router# debug tsp all

01:04:12:CDAPI TSP RX ===> callId=(32 ), Msg=(CDAPI_MSG_CONNECT_IND,1 )
Sub=(CDAPI_MSG_SUBTYPE_NULL,0 )cdapi_tsp_connect_ind
01:04:12:TSP CDAPI:cdapi_free_msg returns 1
01:04:13:tsp_process_event:[0:D, 0.1 , 3] tsp_cdapi_setup_ack tsp_alert
01:04:13:tsp_process_event:[0:D, 0.1 , 5] tsp_alert_ind
01:04:13:tsp_process_event:[0:D, 0.1 , 10]
01:04:14:tsp_process_event:[0:D, 0.1 , 10]
01:04:17:CDAPI TSP RX ===> callId=(32 ), Msg=(CDAPI_MSG_DISCONNECT_IND,7 )
Sub=(CDAPI_MSG_SUBTYPE_NULL,0 )cdapi_tsp_disc_ind
```

```
01:04:17:TSP CDAPI:cdapi_free_msg returns 1
01:04:17:tsp_process_event:[0:D, 0.1 , 27] cdapi_tsp_release_indtsp_disconnet_tdm
01:04:17:tsp_process_event:[0:D, 0.4 , 7] cdapi_tsp_release_comp
```

Related Commands

Command	Description
debug track	Displays information about the telephony service provider.
debug voip rawmsg	Displays the raw message owner, length, and pointer.

debug tunnel rbscp

To turn on the debugging output for Rate Based Satellite Control Protocol (RBSCP) tunnels, use the **debug tunnel rbscp** command in privileged EXEC mode. To turn off debugging output, use the **no** form of this command.

debug tunnel rbscp [**ack_split**| **detail**| **msg**| **rto**| **state**| **window**]

no debug tunnel rbscp [**ack_split**| **detail**| **msg**| **rto**| **state**| **window**]

Syntax Description

ack_split	(Optional) Displays debugging messages about RBSCP ACK splitting.
detail	(Optional) Displays detailed debugging messages about RBSCP.
msg	(Optional) Displays debugging messages about the RBSCP messages.
rto	(Optional) Displays debugging messages about RBSCP round-trip times (RTTs) and retransmission timeouts (RTOs).
state	(Optional) Displays debugging messages about the RBSCP states.
window	(Optional) Displays debugging messages about RBSCP window stuffing.

Command Modes

Privileged EXEC

Command History

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

Usage Guidelines

Use the **debug tunnel rbscp** command in privileged EXEC mode to troubleshoot RBSCP command operations.

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

Examples

The following example turns on debugging messages about RBSCP messages:

```
Router# debug tunnel rbscp msg
Tunnel RBSCP message debugging is on
router#
*Mar  1 05:36:01.435: handling FWD_TSN: sequence=20h, tsn=0h
*Mar  1 05:36:03.371: rbscp_output_a_fwdtsn: tsn=0h, seq=Dh, for_hb=1
*Mar  1 05:36:10.835: handling FWD_TSN: sequence=21h, tsn=0h
*Mar  1 05:36:12.771: rbscp_output_a_fwdtsn: tsn=0h, seq=Eh, for_hb=1
*Mar  1 05:36:20.235: handling FWD_TSN: sequence=22h, tsn=0h
*Mar  1 05:36:22.171: rbscp_output_a_fwdtsn: tsn=0h, seq=Fh, for_hb=1
```

Note The debug output will vary depending on what the router is configured to do after the debug command is entered.

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

Table 13: debug tunnel rbscp msg Field Descriptions

Field	Description
handling FWD_TSN	The router has received and is processing a FWD_TSN message from a peer with a sequence number of 20 hex and a Transport Sequence Number (TSN) of 0 hex.
rbscp_output_a_fwdtsn	The router is sending a FWD_TSN message to the peer with a TSN of 0 hex, a sequence number of 0D hex and it is for a heartbeat (equivalent of a keepalive).

The following example turns on debugging messages about RBSCP round-trip times and retransmission timeouts:

```
Router# debug tunnel rbscp rto
Tunnel RBSCP RTT/RTO debugging is on
router#
*Mar  1 05:36:50.927: update_rtt: cur_rtt:549 ms:548 delay:0
*Mar  1 05:36:50.927: New RTT est:549 RTO:703
*Mar  1 05:37:00.327: update_rtt: cur_rtt:549 ms:548 delay:0
*Mar  1 05:37:00.327: New RTT est:549 RTO:703
*Mar  1 05:37:09.727: update_rtt: cur_rtt:549 ms:548 delay:0
*Mar  1 05:37:09.727: New RTT est:549 RTO:703
```

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

Table 14: debug tunnel rbscp rto Field Descriptions

Field	Description
update rtt: curr rtt	Displays the updated, previous, and current RTT, in milliseconds, and a number that represents the amount of additional delay from queuing.

Field	Description
New RTT est	Displays the estimated new RTT, in milliseconds.
RTO	Displays the new retransmission timeout, in milliseconds.

Related Commands

Command	Description
show rbscp	Displays state and statistical information about RBSCP tunnels.

debug tunnel route-via

To display debugging information about the tunnel transport using a subset of the route table, use the **debug tunnel route-via**command in privileged EXEC mode. To disable this feature, use the no form of this command.

debug tunnel route-via

no debug tunnel route-via

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.4(11)T	This command was introduced.

Examples

The following sample output of **debug tunnel route-via**command displays the outgoing interface for the tunnel transport.

```
Router# debug tunnel route-via
Tunnel route-via debugging is on
*May 22 11:54:34.803: TUN-VIA: Tunnel0 candidate route-via Ethernet0/0, next hop
 10.73.2.1
*May 22 11:54:34.803: TUN-VIA: Tunnel0 route-via action is forward
Router# no debug tunnel route-via
undebug tunnel route-via
Tunnel route-via debugging is off
```

Related Commands

Command	Description
show interface tunnel	Displays information about the physical output tunnel interface.
tunnel route-via	Specifies the outgoing interface of the tunnel transport.

debug txconn all

To turn on all debug flags for Cisco Transaction Connection (CTRC) communications with the Customer Information Control System (CICS), use the **debug txconn all**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug txconn all

no debug txconn all

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging is not enabled for the txconn subsystem.

Command Modes

Privileged EXEC

Command History

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

Examples

The following example shows the immediate output of the **debug txconn all** command. For examples of specific debugging messages, see the examples provided for the **debug txconn appc**, **debug txconn config**, **debug txconn data**, **debug txconn event**, **debug txconn tcp**, and **debug txconn timer** commands.

```
Router# debug txconn all
All possible TXConn debugging has been turned on
```

Related Commands

Command	Description
debug snasw	Displays debugging information related to SNA Switching Services.
debug txconn appc	Displays APPC-related trace or error messages for communications with CICS.
debug txconn config	Displays trace or error messages for CTRC configuration and control blocks for CICS communications.
debug txconn data	Displays CICS client and host data being handled by CTRC, in hexadecimal notation.

Command	Description
debug txconn event	Displays trace or error messages for CTRC events related to CICS communications.
debug txconn tcp	Displays error messages or traces for TCP/IP communications with CICS.
debug txconn timer	Displays performance information related to CICS communications.
show debugging	Displays the state of each debugging option.

debug txconn appc

To display Advanced Program-to-Program Communication (APPC)-related trace or error messages for communications with the Customer Information Control System (CICS), use the **debug txconn appc** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug txconn appc

no debug txconn appc

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging is not enabled for the txconn subsystem.

Command Modes

Privileged EXEC

Command History

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

Examples

The following shows sample APPC debugging output from the **debug txconn appc** command:

```
Router# debug txconn appc
01:18:05: TXCONN-APPC-622ADF38: Verb block =
01:18:05: TXCONN-APPC-622ADF38:  0001 0200 0300 0000 0400 0000 0000 0000
01:18:05: TXCONN-APPC-622ADF38:  0000 00FC 0000 0000 0000 0000 0000 0000
01:18:05: TXCONN-APPC-622ADF38:  0000 0000 0840 0007 0000 0000 0000 0000
01:18:05: TXCONN-APPC-622ADF38:  7BC9 D5E3 C5D9 4040 07F6 C4C2 4040 4040
01:18:05: TXCONN-APPC-622ADF38:  4040 4040 4040 4040 4040 4040 4040 4040
01:18:05: TXCONN-APPC-622ADF38:  4040 4040 4040 4040 4040 4040 4040 4040
01:18:05: TXCONN-APPC-622ADF38:  4040 4040 4040 4040 4040 4040 4040 4040
01:18:05: TXCONN-APPC-622ADF38:  4040 4040 4040 4040 0000 0000 0000 0000
01:18:05: TXCONN-APPC-622ADF38:  0000 0000 0000 0000 0000 0000 0000 0000
01:18:05: TXCONN-APPC-622ADF38:  0000 0000 0000 0000 0000 0000 0000 0000
01:18:05: TXCONN-APPC-622ADF38:  00E2 E3C1 D9E6 4BC7 C1E9 C5D3 D3C5 4040
01:18:05: TXCONN-APPC-622ADF38:  4040 0000 0000 0000 0000 0000
01:18:05: TXCONN-APPC-621E5730: Verb block =
01:18:05: TXCONN-APPC-621E5730:  0001 0200 0300 0000 0400 0000 0000 0000
01:18:05: TXCONN-APPC-621E5730:  0000 00FD 0000 0000 0000 0000 0000 0000
01:18:05: TXCONN-APPC-621E5730:  0000 0000 0840 0007 0000 0000 0000 0000
01:18:05: TXCONN-APPC-621E5730:  C9C2 D4D9 C4C2 4040 07F6 C4C2 4040 4040
01:18:05: TXCONN-APPC-621E5730:  4040 4040 4040 4040 4040 4040 4040 4040
01:18:05: TXCONN-APPC-621E5730:  4040 4040 4040 4040 4040 4040 4040 4040
01:18:05: TXCONN-APPC-621E5730:  4040 4040 4040 4040 4040 4040 4040 4040
01:18:05: TXCONN-APPC-621E5730:  4040 4040 4040 4040 0000 0000 0000 0000
01:18:05: TXCONN-APPC-621E5730:  0000 0000 0000 0000 0000 0000 0000 0000
01:18:05: TXCONN-APPC-621E5730:  0000 0000 0000 0000 0000 0000 0000 0000
01:18:05: TXCONN-APPC-621E5730:  00E2 E3C1 D9E6 4BE2 E3C5 D3D3 C140 4040
01:18:05: TXCONN-APPC-621E5730:  4040 0000 0000 0000 0000 0000
```

Related Commands

Command	Description
debug snasw	Displays debugging information related to SNA Switching Services.
debug txconn all	Displays all CTRC debugging information related to communications with CICS.
debug txconn config	Displays trace or error messages for CTRC configuration and control blocks for CICS communications.
debug txconn data	Displays CICS client and host data being handled by CTRC, in hexadecimal notation.
debug txconn event	Displays trace or error messages for CTRC events related to CICS communications.
debug txconn tcp	Displays error messages or traces for TCP/IP communications with CICS.
debug txconn timer	Displays performance information related to CICS communications.
show debugging	Displays the state of each debugging option.

debug txconn config

To display trace or error messages for Cisco Transaction Connection (CTRC) configuration and control blocks for Customer Information Control System (CICS) communications, use the **debug txconn config**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug txconn config

no debug txconn config

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging is not enabled for the txconn subsystem.

Command Modes

Privileged EXEC

Command History

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

Examples

The following shows sample output from the **debug txconn config** command:

```
Router# debug txconn config
22:11:37: TXCONN-CONFIG: deleting transaction 61FCE414
22:11:37: TXCONN-CONFIG: deleting connection 61FB5CB0
22:11:37: TXCONN-CONFIG: server 62105D6C releases connection 61FB5CB0
22:11:44: TXCONN-CONFIG: new connection 61FB64A0
22:11:44: TXCONN-CONFIG: server 6210CEB4 takes connection 61FB64A0
22:11:44: TXCONN-CONFIG: new transaction 61E44B9C
22:11:48: TXCONN-CONFIG: deleting transaction 61E44B9C
22:11:53: TXCONN-CONFIG: new transaction 61E44B9C
22:11:54: TXCONN-CONFIG: deleting transaction 61E44B9C
```

Related Commands

Command	Description
debug snasw	Displays debugging information related to SNA Switching Services.
debug txconn all	Displays all CTRC debugging information related to communications with CICS.
debug txconn appc	Displays APPC-related trace or error messages for communications with CICS.

Command	Description
debug txconn data	Displays CICS client and host data being handled by CTRC, in hexadecimal notation.
debug txconn event	Displays trace or error messages for CTRC events related to CICS communications.
debug txconn tcp	Displays error messages or traces for TCP/IP communications with CICS.
debug txconn timer	Displays performance information related to CICS communications.
show debugging	Displays the state of each debugging option.

debug txconn data

To display a hexadecimal dump of Customer Information Control System (CICS) client and host data being handled by Cisco Transaction Connection (CTRC), plus information about certain CTRC internal operations, use the **debug txconn data** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug txconn data

no debug txconn data

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging is not enabled for the txconn subsystem.

Command Modes

Privileged EXEC

Command History

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

Examples

The following shows selected output from the **debug txconn data** command when a connection is established, data is received from the client via TCP/IP, data is sent to the client, and then the connection is closed.

```
Router# debug txconn data
TXConn DATA debugging is on
00:04:50: TXConn(62197464) Created
00:04:50: TXConn(62197464) State(0) MsgID(0) -> nextState(1)
00:04:50: TXConn(62197464) Client->0000 003A 0000 0002 000B 90A0
00:04:50: TXConn(62197464) Received LL 58 for session(0 0 2).
00:06:27: TXConn(62197464) Client<-0000 0036 0000 0003 000B 8001 0707 0864
00:06:53: TXConn(62175024) Deleted
```

The following lines show output when data is sent to the host:

```
00:04:50: TXTrans(id:62197910 conn:62197464 addr:2) LL(58) FMH5(0) CEBI(0)
00:04:50: TXTrans(id:62197910 conn:62197464 addr:2) State(0) MsgID(7844) -> nextState(1)
00:04:50: TXTrans(id:62197910 conn:62197464 addr:2) conversationType(mapped) syncLevel(1)
sec(0)
00:04:50: TXTrans(id:62197910 conn:62197464 addr:2) TPName CCIN
00:04:50: TXTrans(id:62197910 conn:62197464 addr:2) apDataLength(32) GDSID(12FF)
00:04:50: TXTrans(id:62197910 conn:62197464 addr:2) ->Host 0000 0008 03F4 F3F7 0000 0008
0401 0000
```

The following lines show output when data is received from the host:

```
00:05:01: TXTrans(id:62197910 conn:62197464 addr:2) <-Host  0092 12FF 0000 000C 0102 0000
0000 0002
```

The following lines show CTRC generating an FMH7 error message indicating that a CICS transaction has failed at the host or has been cleared by a router administrator:

```
00:06:27: TXTrans(id:6219853C conn:62197464 addr:3) Generating FMH7.
00:06:27: %TXCONN-3-TXEXCEPTION: Error occurred from transaction 3 of client 157.151.241.10
 connected to server CICSC, exception type is 9
```

The following line shows CTRC responding to an FMH7 error message sent by the CICS client program:

```
00:07:11: TXTrans(id:62197910 conn:62197464 addr:2) Generating FMH7 +RSP.
```

Related Commands

Command	Description
debug snasw	Displays debugging information related to SNA Switching Services.
debug txconn all	Displays all CTRC debugging information related to communications with CICS.
debug txconn appc	Displays APPC-related trace or error messages for communications with CICS.
debug txconn config	Displays trace or error messages for CTRC configuration and control blocks for CICS communications.
debug txconn event	Displays trace or error messages for CTRC events related to CICS communications.
debug txconn tcp	Displays error messages or traces for TCP/IP communications with CICS.
debug txconn timer	Displays performance information related to CICS communications.
show debugging	Displays the state of each debugging option.

debug txconn event

To display trace or error messages for Cisco Transaction Connection (CTRC) events related to Customer Information Control System (CICS) communications, use the **debug txconn event**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug txconn event

no debug txconn event

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging is not enabled for the txconn subsystem.

Command Modes

Privileged EXEC

Command History

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

Examples

The following shows sample output from the **debug txconn event** command:

```
Router# debug txconn event
TXConn event debugging is on
Router#
22:15:08: TXCONN-EVENT: [*] Post to 62146464(cn), from 6211E744(tc), msg
61FC6170, msgid 0x6372 'cr', buffer 6211289C.
22:15:08: TXCONN-EVENT: Dispatch to 62146464, from 6211E744, msg 61FC6170,
msgid 6372 'cr', buffer 6211289C.
22:15:08: TXCONN-EVENT: [*] Post to 61E44BA0(sn), from 62146464(cn), msg
621164D0, msgid 0x7844 'xD', buffer 0.
22:15:08: TXCONN-EVENT: [*] Post to 6211E744(tc), from 62146464(cn), msg
61FC6170, msgid 0x6347 'cG', buffer 0.
22:15:08: TXCONN-EVENT: Dispatch to 61E44BA0, from 62146464, msg 621164D0,
msgid 7844 'xD', buffer 0.
22:15:08: TXCONN-EVENT: Dispatch to 6211E744, from 62146464, msg 61FC6170,
msgid 6347 'cG', buffer 0.
22:15:08: TXCONN-EVENT: [*] Post to 62146464(cn), from 6211E744(tc), msg
61FC6170, msgid 0x6372 'cr', buffer 6211289C.
22:15:08: TXCONN-EVENT: Dispatch to 62146464, from 6211E744, msg 61FC6170,
msgid 6372 'cr', buffer 6211289C.
22:15:08: TXCONN-EVENT: [*] Post to 61E44BA0(sn), from 62146464(cn), msg
61FBFBF4, msgid 0x7844 'xD', buffer 0.
22:15:08: TXCONN-EVENT: [*] Post to 6211E744(tc), from 62146464(cn), msg
61FC6170, msgid 0x6347 'cG', buffer 0.
22:15:08: TXCONN-EVENT: Dispatch to 61E44BA0, from 62146464, msg 61FBFBF4,
msgid 7844 'xD', buffer 0.
22:15:08: TXCONN-EVENT: [*] Post to 61FC6394(ap), from 61E44BA0(sn), msg
621164D0, msgid 0x634F 'cO', buffer 0.
```

```
22:15:08: TXCONN-EVENT: Dispatch to 6211E744, from 62146464, msg 61FC6170,
msgid 6347 'cG', buffer 0.
```

Related Commands

Command	Description
debug snasw	Displays debugging information related to SNA Switching Services.
debug txconn all	Displays all CTRC debugging information related to communications with CICS.
debug txconn appc	Displays APPC-related trace or error messages for communications with CICS.
debug txconn config	Displays trace or error messages for CTRC configuration and control blocks for CICS communications.
debug txconn data	Displays CICS client and host data being handled by CTRC, in hexadecimal notation.
debug txconn tcp	Displays error messages or traces for TCP/IP communications with CICS.
debug txconn timer	Displays performance information related to CICS communications.
show debugging	Displays the state of each debugging option.

debug txconn tcp

To display error messages and traces for TCP, use the **debug txconn tcp**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug txconn tcp

no debug txconn tcp

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging is not enabled for the txconn subsystem.

Command Modes

Privileged EXEC

Command History

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

Examples

The following shows sample output from the **debug txconn tcp**command:

```
Router# debug txconn tcp
TXCONN-TCP-63528473: tcpdriver_passive_open returned NULL
TXCONN-TCP-63528473: (no memory) tcp_reset(63829482) returns 4
TXCONN-TCP: tcp_accept(74625348,&error) returns tcb 63829482, error 4
TXCONN-TCP: (no memory) tcp_reset(63829482) returns 4
TXCONN-TCP-63528473: (open) tcp_create returns 63829482, error = 4
TXCONN-TCP-63528473: tcb_connect(63829482,1.2.3.4,2010) returns 4
TXCONN-TCP-63528473: (open error) tcp_reset(63829482) returns 4
TXCONN-TCP-63528473: tcp_create returns 63829482, error = 4
TXCONN-TCP-63528473: tcb_bind(63829482,0.0.0.0,2001) returns 4
TXCONN-TCP-63528473: tcp_listen(63829482,,) returns 4
TXCONN-TCP-63528473: (errors) Calling tcp_close (63829482)
```

Related Commands

Command	Description
debug ip	Displays debugging information related to TCP/IP communications.
debug snasw	Displays debugging information related to SNA Switching Services.
debug txconn all	Displays all CTRC debugging information related to communications with CICS.

Command	Description
debug txconn appc	Displays APPC-related trace or error messages for communications with CICS.
debug txconn config	Displays trace or error messages for CTRC configuration and control blocks for CICS communications.
debug txconn data	Displays CICS client and host data being handled by CTRC, in hexadecimal notation.
debug txconn event	Displays trace or error messages for CTRC events related to CICS communications.
debug txconn timer	Displays performance information related to CICS communications.
show debugging	Displays the state of each debugging option.

debug txconn timer

To display performance information regarding Cisco Transaction Connection (CTRC) communications with Customer Information Control System (CICS), use the **debug txconn timer** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug txconn timer

no debug txconn timer

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging is not enabled for the txconn subsystem.

Command Modes

Privileged EXEC

Command History

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

Examples

The following example shows turnaround time and host response time in milliseconds for a CICS transaction requested through CTRC. Turnaround time is measured from when CTRC receives the first request packet for the transaction until CTRC sends the last response packet of the transaction to the client. Host response time is measured from when CTRC sends the last request packet for a transaction to the host until CTRC receives the first response packet from the host for that transaction.

```
Router# debug txconn timer
TXConn timer debugging is on
00:04:14: TXTrans(id:622F4350 conn:62175024 addr:1) Turnaround Time = 4536(msec)
HostResponseTime = 120(msec)
```

Related Commands

Command	Description
debug snasw	Displays debugging information related to SNA Switching Services.
debug txconn all	Displays all CTRC debugging information related to communications with CICS.
debug txconn appc	Displays APPC-related trace or error messages for communications with CICS.

Command	Description
debug txconn config	Displays trace or error messages for CTRC configuration and control blocks for CICS communications.
debug txconn data	Displays CICS client and host data being handled by CTRC, in hexadecimal notation.
debug txconn event	Displays trace or error messages for CTRC events related to CICS communications.
debug txconn tcp	Displays error messages or traces for TCP/IP communications with CICS.
show debugging	Displays the state of each debugging option.

debug udptn

To display debug messages for UDP Telnet (UDPTN) events, use the **debug udptn**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug udptn

no debug udptn

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled.

Command Modes

Privileged EXEC

Command History

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

Examples

The following is sample output from the **debug udptn** command:

```
terrapin# debug udptn
terrapin#
udptn 172.16.1.1
Trying 172.16.1.1 ... Open
*Mar 1 00:10:15.191:udptn0:adding multicast group.
*Mar 1 00:10:15.195:udptn0:open to 172.16.1.1:57 Loopback0jjaassdd
*Mar 1 00:10:18.083:udptn0:output packet w 1 bytes
*Mar 1 00:10:18.087:udptn0:Input packet w 1 bytes
terrapin#
disconnect
Closing connection to 172.16.1.1 [confirm] y
terrapin#
*Mar 1 00:11:03.139:udptn0:removing multicast group.
```

Related Commands

Command	Description
udptn	Enables transmission or reception of UDP packets.
transport output	Defines the protocol that can be used for outgoing connections from a line.

debug usb driver

To display debug messages about universal serial bus (USB) transfers, use the **debug usb driver**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug usb driver [**transfer** *transfer-method*]

no debug usb driver [**transfer** *transfer-method*]

Syntax Description

transfer	(Optional) Specifies the type of transfer method for which messages are to be displayed on the console.
transfer-method	One of the following options: **interrupt**, **bulk**, or **control**.

Command Default

None

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(14)T	This command was introduced.
12.4(11)T	This command was integrated into the Cisco 7200VXR NPE-G2 platform.

Usage Guidelines

The **debug usb driver** command produces a large amount of data that might slow down your system, so use this command with caution.

Examples

The following sample debug output is produced when the **debug usb driver**command with the **transfer** and **control** keywords is issued and when an eToken is unplugged and plugged back in:

```
Router# debug usb driver transfer bulk

USB Driver Bulk Transfer debugging is on
Router# debug usb driver transfer control
USB Driver Control Transfer debugging is on
Router# debug usb stack

Stack debugging is on
Router#
Router#
*Dec 22 06:18:29.399:%USB_HOST_STACK-6-USB_DEVICE_DISCONNECTED:A USB device has been removed
 from port 1.
```

```
*Dec 22 06:18:29.499:Detached:
*Dec 22 06:18:29.499:Host:         1
*Dec 22 06:18:29.499:Address:      18
*Dec 22 06:18:29.499:Manufacturer: AKS
*Dec 22 06:18:29.499:Product:      eToken Pro 4254
*Dec 22 06:18:29.499:Serial Number:
Router#
*Dec 22 06:18:29.499:%USB_TOKEN_FILESYS-6-USB_TOKEN_REMOVED:USB Token device
removed:usbtoken1.
*Dec 22 06:18:29.499:%CRYPTO-6-TOKENREMOVED:Cryptographic token eToken removed from usbtoken1
Router#
Router#
Router#
Router#
Router#
*Dec 22 06:18:38.063:%USB_HOST_STACK-6-USB_DEVICE_CONNECTED:A Low speed USB device has been
 inserted in port 1.
*Dec 22 06:18:38.683:ATTACHED===>Class-driver activated
*Dec 22 06:18:38.683:Host:         1
*Dec 22 06:18:38.683:Address:      19
*Dec 22 06:18:38.683:Manufacturer: AKS
*Dec 22 06:18:38.683:Product:      eToken Pro 4254
*Dec 22 06:18:38.683:Serial Number:
*Dec 22 06:18:39.383:Control Transfer
Device Handle:0x3010000
Direction:0x0
Request:0x1
Type:0x40
Recipient:0x0
ValueDesc:0x0
ValueIndex:0x0
Index:0x0
*Dec 22 06:18:39.383:Control Transfer
Device Handle:0x3010000
Direction:0x80
Request:0x81
Type:0x40
Recipient:0x0
ValueDesc:0x0
ValueIndex:0x0
Index:0x0
*Dec 22 06:18:39.407:Control Transfer
Device Handle:0x3010000
Direction:0x0
Request:0x3
Type:0x40
Recipient:0x0
ValueDesc:0x0
ValueIndex:0x0
Index:0x0
*Dec 22 06:18:39.407:Control Transfer
Device Handle:0x3010000
Direction:0x80
Request:0
my3825#x83
Type:0x40
Recipient:0x0
ValueDesc:0x0
ValueIndex:0x0
Index:0x0
*Dec 22 06:18:39.503:Control Transfer
Device Handle:0x3010000
Direction:0x0
Request:0x2
Type:0x40
Recipient:0x0
ValueDesc:0x0
ValueIndex:0x0
Index:0x0
*Dec 22 06:18:39.507:Control Transfer
Device Handle:0x3010000
Direction:0x80
Request:0x82
```

```
Type:0x40
Recipient:0x0
ValueDesc:0x0
ValueIndex:0x0
Index:0x0
*Dec 22 06:18:39.507:%USB_TOKEN_FILESYS-6-USB_TOKEN_INSERTED:USB Token device
inserted:usbtoken1.
*Dec 22 06:18:39.515:Control Transfer
Device Handle:0x3010000
Direction:0x0
Request:0x6
Type:0x40
Recipient:0x0
ValueDesc:0x0
ValueIndex:0x0
Index:0x0
*Dec 22 06:18:39.515:%USB_TOKEN_FILESYS-6-REGISTERING_WITH_IFS:Registering USB Token File
System usbtoken1:might take a while...
*Dec 22 06:18:39.515:Control Transfer
Device Handle:0x3010000
Direction:0x80
Request:0x86
Type:0x40
Recipient:0x0
ValueDesc:0x0
ValueIndex:0x0
Index:0x0
*Dec 22 06:18:39.543:Control Transfer
Device Handle:0x3010000
Direction:0x0
Request:0x6
Type:0x40
Recipient:0x0
ValueDesc:0x0
ValueIndex:0x0
Index:0x0
.
.
.
```

debug user-group

To display information about the user group, use the **debug user-group**command in privileged EXEC mode. To disable this feature, use the **no** form of this command.

debug user-group {**additions**| **all**| **api**| **database**| **deletions**}

no debug user-group {**additions**| **all**| **api**| **database**| **deletions**}

Syntax Description

additions	Displays debugging information about additions to the user-group.
all	Displays all debugging information about the user-group.
api	Displays debugging information about the user-group Application Programming Interface (API).
database	Displays debugging information about the user-group database of associated source IP addresses.
deletions	Displays debugging information about deletions from the user-group.

Command Modes

Privileged EXEC (#)

Command History

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

Usage Guidelines

To troubleshoot user-based firewall support, use the **debug user-group** command.

Examples

The following example configures debugging for user-group additions:

```
Router# debug user-group additions
Usergroup Additions debugging is on
Router
#
```

Related Commands

Command	Description
user-group	Defines the user-group associated with the identity policy.

debug v120 event

To display information on V.120 activity, use the **debug v120 event**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug v120 event

no debug v120 event

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

V.120 is an ITU specification that allows for reliable transport of synchronous, asynchronous, or bit transparent data over ISDN bearer channels.

For complete information on the V.120 process, use the **debug v120 packet** command along with the **debug v120 event** command. V.120 events are activity events rather than error conditions.

Examples

The following is sample output from the **debug v120 event**command of V.120 starting up and stopping. Also included is the interface that V.120 is running on (BR 0) and where the V.120 configuration parameters are obtained from (default).

```
Router# debug v120 event
0:01:47: BR0:1-v120 started - Setting default V.120 parameters
0:02:00: BR0:1:removing v120
```

Related Commands

Command	Description
debug v120 packet	Displays general information on all incoming and outgoing V.120 packets.

debug v120 packet

To display general information on all incoming and outgoing V.120 packets, use the **debug v120 packet**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug v120 packet

no debug v120 packet

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

The **debug v120 packet** command shows every packet on the V.120 session. You can use this information to determine whether incompatibilities exist between Cisco's V.120 implementation and other vendors' V.120 implementations.

V.120 is an ITU specification that allows for reliable transport of synchronous, asynchronous, or bit transparent data over ISDN bearer channels.

For complete information on the V.120 process, use the **debug v120 events** command along with the **debug v120 packet**command.

Examples

The following is sample output from the **debug v120 packet** command for a typical session startup:

```
Router# debug v120 packet
0:03:27: BR0:1: I SABME:lli 256 C/R 0 P/F=1
0:03:27: BR0:1: O UA:lli 256 C/R 1 P/F=1
0:03:27: BR0:1: O IFRAME:lli 256 C/R 0 N(R)=0 N(S)=0 P/F=0 len 43
0x83 0xD 0xA 0xD 0xA 0x55 0x73 0x65
0x72 0x20 0x41 0x63 0x63 0x65 0x73 0x73
0:03:27: BR0:1: I RR:lli 256 C/R 1 N(R)=1 P/F=0
0:03:28: BR0:1: I IFRAME:lli 256 C/R 0 N(R)=1 N(S)=0 P/F=0 len 2
0x83 0x63
0:03:28: BR0:1: O RR:lli 256 C/R 1 N(R)=1 P/F=0
0:03:29: BR0:1: I IFRAME:lli 256 C/R 0 N(R)=1 N(S)=1 P/F=0 len 2
0x83 0x31
0:03:29: BR0:1: O RR:lli 256 C/R 1 N(R)=2 P/F=0
%LINEPROTO-5-UPDOWN: Line protocol on Interface BRI0: B-Channel 1, changed state to up
0:03:31: BR0:1: I IFRAME:lli 256 C/R 0 N(R)=1 N(S)=2 P/F=0 len 2
0x83 0x55
0:03:32: BR0:1: I IFRAME:lli 256 C/R 0 N(R)=1 N(S)=3 P/F=0 len 3
0x83 0x31 0x6F
0:03:32: BR0:1: O RR:lli 256 C/R 1 N(R)=3 P/F=0
0:03:32: BR0:1: I IFRAME:lli 256 C/R 0 N(R)=1 N(S)=4 P/F=0 len 2
0x83 0x73
0:03:32: BR0:1: O RR:lli 256 C/R 1 N(R)=5 P/F=0
0:03:32: BR0:1: I IFRAME:lli 256 C/R 0 N(R)=1 N(S)=5 P/F=0 len 2
0x83 0xA
0:03:32: BR0:1: O IFRAME:lli 256 C/R 0 N(R)=6 N(S)=1 P/F=0 len 9
0x83 0xD 0xA 0x68 0x65 0x66 0x65 0x72 0x3E
```

The table below describes the significant fields in the display.

Table 15: debug v120 packet Field Descriptions

Field	Descriptions
BR0:1	Interface number associated with this debugging information.
I/O	Packet going into or out of the interface.
SABME, UA, IFRAME, RR	V120 packet type: • SABME--Set asynchronous balanced mode, extended • US--Unnumbered acknowledgment • IFRAME--Information frame • RR--Receive ready
lli 256	Logical link identifier number.
C/R 0	Command or response.
P/F=1	Poll final.
N(R)=0	Number received.
N(S)=0	Number sent.
len 43	Number of data bytes in the packet.
0x83	Up to 16 bytes of data.

Related Commands

Command	Description
debug tarp events	Displays information on TARP activity.

debug vfi checkpoint

To debug virtual forwarding instance (VFI) checkpointing events and errors, use the **debug vfi checkpoint** command in privileged EXEC mode. To disable debugging of VFI checkpointing events and errors, use the **no** form of this command.

debug vfi checkpoint

no debug vfi checkpoint

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(33)SRC	This command was introduced.
12.2(50)SY	This command was integrated into Cisco IOS Release12.2(50)SY .

Examples

The following is sample output from the **debug vfi checkpoint** command:

```
Router# debug vfi checkpoint

*Jun  5 22:37:17.268: AToM HA: CF status 3 not processed
*Jun  5 22:37:17.268: VFI HA: CF status 3 not processed
*Jun  5 22:37:17.296: AC HA RF: CId:83, Seq:228, Sta:RF_STATUS_PEER_COMM, Opr:0, St:ACTIVE,
 PSt:STANDBY HOT
*Jun  5 22:37:17.296: VFI HA: CID 145, Seq 229, Status RF_STATUS_PEER_COMM, Op 0, State
ACTIVE, Peer STANDBY HOT
*Jun  5 22:37:17.296: AToM HA: CID 84, Seq 230, Status RF_STATUS_PEER_COMM, Op 0, State
ACTIVE, Peer STANDBY HOT
*Jun  5 22:37:17.444: AToM HA: CF status 3 not processed
*Jun  5 22:37:17.444: VFI HA: CF status 3 not processed
*Jun  5 22:37:17.268: %OIR-SP-3-PWRCYCLE: Card in module 6, is being power-cycled (RF
request)
*Jun  5 22:37:17.792: AC HA RF: CId:83, Seq:228, Sta:RF_STATUS_PEER_PRESENCE, Opr:0,
St:ACTIVE, PSt:DISABLED
*Jun  5 22:37:17.792: VFI HA: CID 145, Seq 229, Status RF_STATUS_PEER_PRESENCE, Op 0, State
 ACTIVE, Peer DISABLED
*Jun  5 22:40:40.244: SP-STDBY: SP: Currently running ROMMON from S (Gold) region
*Jun  5 22:40:45.028: %DIAG-SP-STDBY-6-RUN_MINIMUM: Module 6: Running Minimal Diagnostics...
*Jun  5 22:40:56.492: %DIAG-SP-STDBY-6-DIAG_OK: Module 6: Passed Online Diagnostics
*Jun  5 22:41:53.436: %SYS-SP-STDBY-5-RESTART: System restarted
*Jun  5 22:42:12.760: VFI HA: CID 145 Seq 229 Event RF_PROG_STANDBY_BULK Op 0 State ACTIVE
 Peer STANDBY COLD-BULK
*Jun  5 22:42:12.764: VFI HA: Ignore RF progression event, VFI Mgr process is not running,
 skipped bulk sync
*Jun  5 22:42:16.948: %ISSU_PROCESS-SP-7-DEBUG: Peer state is [ STANDBY HOT ]; Please issue
 the runversion command
*Jun  5 22:42:15.928: %PFREDUN-SP-STDBY-6-STANDBY: Ready for SSO mode
*Jun  5 22:42:16.956: %RF-SP-5-RF_TERMINAL_STATE: Terminal state reached for (SSO)
*Jun  5 22:42:16.112: %SYS-SP-STDBY-3-LOGGER_FLUSHED: System was paused for 00:00:00 to
ensure console debugging output--
```

Related Commands

Command	Description
debug cwan atom	Enables debugging of AToM platform events.
debug cwan ltl	Enables debugging of LTL manager platform events.
debug issu client negotiation	Enables debugging of ISSU client negotiation events and errors concerning message versions or client capabilities.
debug issu client registration	Enables debugging of ISSU client registration events and errors concerning message versions or client capabilities.
debug issu client transform	Enables debugging of ISSU client transform events and errors.

debug vg-anylan

To monitor error information and 100VG-AnyLAN port adapter connection activity, use the **debug vg-anylan** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug vg-anylan

no debug vg-anylan

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

Thiscommand could create a substantial amount of command output.

Examples

The following is sample output from the **debug vg-anylan**command:

```
Router# debug vg-anylan
%HP100VG-5-LOSTCARR: HP100VG(2/0), lost carrier
```

The table below lists the messages that could be generated by this command.

Table 16: debug vg-anylan Message Descriptions

Message	Description	Action
%HP100VG-5-LOSTCARR: HP100VG(2/0), lost carrier	Lost carrier debug message. The VG controller detects that the link to the hub is down due to cable, hub, or VG controller problem.	Check, repair, or replace the cable or hub. If you determine that the cable and hub are functioning normally, repair or replace the 100VG-AnyLAN port adapter.
%HP100VG-5-CABLEERR: HP100VG(2/0), cable error, training failed	Bad cable error messages. Cable did not pass training.[1]	Check, repair, or replace the cable or hub. If you determine that the cable and hub are functioning normally, repair or replace the 100VG-AnyLAN port adapter.
%HP100VG-5-NOCABLE: HP100VG(2/0), no tone detected, check cable, hub	No cable attached error message. The VG MAC cannot hear tones from the hub.1	Check, repair, or replace the cable or hub. If you determine that the cable and hub are functioning normally, repair or replace the 100VG-AnyLAN port adapter.

Message	Description	Action
HP100VG-1-FAIL: HP100VG(2/0), Training Fail - unable to login to the hub	Training to the VG network failed. Login to the hub rejected by the hub.1	Take action based on the following error messages: • %HP100VG-1-DUPMAC: HP100VG(2/0), A duplicate MAC address has been detected. • HP100VG-1-LANCNF: HP100VG(2/0), Configuration is not compatible with the network. • %HP100VG-1-ACCESS: HP100VG(2/0), Access to network is not allowed.
%HP100VG-1-DUPMAC: HP100VG(2/0), A duplicate MAC address has been detected	Duplicate MAC address on the same VG network. Two VG devices on the same LAN segment have the same MAC address.	Check the router configuration to make sure that no duplicate MAC address is configured.
%HP100VG-1-LANCNF: HP100VG(2/0), Configuration is not compatible with the network	Configuration of the router is not compatible to the network.	Check that the configuration of the hub for Frame Format, Promiscuous, and Repeater bit indicates the proper configuration.
%HP100VG-1-ACCESS: HP100VG(2/0), Access to network is not allowed	Access to the VG network is denied by the hub.	Check the configuration of the hub.
%HP100VG-3-NOTHP100VG: Device reported 0x5101A	Could not find the 100VG PCI device on a 100VG-AnyLAN port adapter.	Make sure the 100VG-AnyLAN port adapter is properly seated in the slot. Otherwise repair or replace the 100VG-AnyLAN port adapter.
%HP100VG-1-DISCOVER: Only found 0 interfaces on bay 2, shutting down bay	No 100VG interface detected on a 100VG-AnyLAN port adapter in a slot.	Make sure the 100VG-AnyLAN port adapter is properly seated in the slot. Otherwise repair or replace the 100VG-AnyLAN port adapter.

1 This message might be displayed when the total load on the cascaded hub is high. Wait at least 20 seconds before checking to determine if the training really failed. Check if the protocol is up after 20 seconds before starting troubleshooting.

debug video vicm

To display debugging messages for the Video Call Manager (ViCM) that handles video calls, enter the **debug video vicm** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug video vicm

no debug video vicm

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(5)XK	This command was introduced.
12.0(6)T	This command was modified.

Examples

The following shows sample output when you use the **debug video vicm** command. Comments are enclosed in asterisks (*).

```
Router# debug video vicm
Video ViCM FSM debugging is on
***** Starting Video call *****
Router# SVC HANDLE in rcvd:0x80001B:
00:42:55:ViCM - current state = Idle, Codec Ready
00:42:55:ViCM - current event = SVC Setup
00:42:55:ViCM - new state = Call Connected
00:42:55:ViCM - current state = Call Connected
00:42:55:ViCM - current event = SVC Connect Ack
00:42:55:ViCM - new state = Call Connected
*****Video Call Disconnecting*****
Router#
00:43:54:ViCM - current state = Call Connected
00:43:54:ViCM - current event = SVC Release
00:43:54:ViCM - new state = Remote Hangup
00:43:54:ViCM - current state = Remote Hangup
00:43:54:ViCM - current event = SVC Release Complete
00:43:54:ViCM - new state = Remote Hangup
mc3810_video_lw_periodic:Codec is not ready
mc3810_video_lw_periodic:sending message
00:43:55:ViCM - current state = Remote Hangup
00:43:55:ViCM - current event = DTR Deasserted
00:43:55:ViCM - new state = Idle
mc3810_video_lw_periodic:Codec is ready
mc3810_video_lw_periodic:sending message
00:43:55:ViCM - current state = Idle
```

```
00:43:55:ViCM - current event = DTR Asserted
00:43:55:ViCM - new state = Idle, Codec Ready
```

debug vlan packet

To display general information on virtual LAN (VLAN) packets that the router received but is not configured to support, use the **debug vlan packet**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug vlan packet

no debug vlan packet

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

The **debug vlan packet** command displays only packets with a VLAN identifier that the router is not configured to support. This command allows you to identify other VLAN traffic on the network. Virtual LAN packets that the router is configured to route or switch are counted and indicated when you use the **show vlans** command.

Examples

The following is sample output from the **debug vlan packet** output. In this example, a VLAN packet with a VLAN ID of 1000 was received on FDDI interface 0 and this interface was not configured to route or switch this VLAN packet:

```
Router# debug vlan packet
vLAN: IEEE 802.10 packet bearing vLAN ID 1000 received on interface
   Fddi0 which is not configured to route/switch ID 1000.
```

debug voice aaa asnl

To display debugging messages for gateway authentication, authorization, and accounting (AAA) Application Subscribe/Notify Layer (ASNL), use the **debug voice aaa asnl** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voice aaa asnl

no debug voice aaa asnl

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging of AAA ASNL is not enabled.

Command Modes

Privileged EXEC

Command History

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

Usage Guidelines

The **debug voice aaa asnl** command is a subset of the **debug voice aaa** command. It displays only events and error information related to the Accounting Server Connectivity Failure and Recovery Detection feature.

Examples

The following output is displayed when the **debug voice aaa asnl** command is entered:

```
Router# debug voice aaa asnl
01:39:15:voip_aaa_accounting_pthru_send:Method List Name:ml1, aaa_av_list 0x62D69FCC,
acct_rec_type 3
01:39:15:voip_aaa_search_mlist_node_by_name:Method List Name:ml15
01:39:15:voip_aaa_accounting_pthru_send:Accounting Probe UID=1, adb = 629977A0
01:39:15:voip_aaa_accounting_pthru_send(1):increment num_acct_sent counter
```

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

Table 17: debug voice aaa asnl Field Descriptions

Field	Description
acct_rec_type	Accounting record type: START (1), UPDATE (2), STOP (3), ACCT_ON (4).
Accounting Probe UID	ID of the accounting probe record.
Method List Name	Method list name.

debug voice all

To display debugging information for all components of the Voice Call Manager, use the **debug voice all** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voice all[*slot*/*port*]

no debug voice all[*slot*/*port*]

Syntax Description

slot / *port*	(Optional) The slot and port number of the voice port. If the *slot* and *port* arguments are entered, only debugging information for that voice port is displayed. If the *slot* and *port* are not entered, debugging information for all voice ports is displayed.

Command Modes

Privileged EXEC

Usage Guidelines

This command is valid on the Cisco MC3810 only.

Examples

The **debug voice all** command output provides debug output for all the **debug** commands for the Voice Call Manager compiled into one display. For sample output of the individual commands, see the sample displays for the **debug voice cp, debug voice eecm, debug voice protocol, debug voice signaling**, and **debug voice tdsm**commands.

Related Commands

Command	Description
debug voice eecm	Displays debugging information for the Voice End-to-End Call Manager.
debug voice protocol	Displays debugging information for the Voice Line Protocol State machine.
debug voice signaling	Displays debugging information for the voice port signaling.
debug voice tdsm	Displays debugging information for the voice tandem switch.
debug voice ccapi	Debugs the call control API.

debug voice cp

To display debugging information for the Voice Call Processing State Machine, use the **debug voice cp** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voice cp[*slot*/*port*]

no debug voice cp[*slot*/*port*]

Syntax Description

slot/port	(Optional) The slot and port number of the voice port. If the slot and port arguments are entered, only debugging information for that voice port is displayed.

Command Modes

Privileged EXEC

Usage Guidelines

This command is valid on the Cisco MC3810 only.

Examples

The following is sample output from the **debug voice cp** command:

```
Router# debug voice cp 1/1
Voice Call Processing State Machine debugging is on
1/1: CPD( ), idle gets event seize_ind
1/1: CPD( ), idle gets event dsp_ready
1/1: CPD( ), idle ==> collect
1/1: CPD(in), collect gets event digit
1/1: CPD(in), collect gets event digit
1/1: CPD(in), collect gets event digit
1/1: CPD(in), collect gets event digit
1/1: CPD(in), collect gets event addr_done
1/1: CPD(in), collect ==> request
1/1: CPD(in), request gets event call_proceeding
1/1: CPD(in), request ==> in_wait_answer
1/1: CPD(in), in_wait_answer gets event call_accept
1/1: CPD(in), in_wait_answer gets event call_answered
1/1: CPD(in), in_wait_answer ==> connected
1/1: CPD(in), connected gets event peer_onhook
1/1: CPD(in), connected ==> disconnect_wait
1/1: CPD(in), disconnect_wait gets event idle_ind
1/1: CPD(in), disconnect_wait ==> idle
```

Related Commands

Command	Description
debug voice all	Displays debugging information for all components of the Voice Call Manager.
debug voice eecm	Displays debugging information for the Voice End-to-End Call Manager.

Command	Description
debug voice protocol	Displays debugging information for the Voice Line protocol State machine.
debug voice signaling	Displays debugging information for the voice port signaling.
debug voice tdsm	Displays debugging information for the voice tandem switch.

debug voice dsp crash-dump

To display debugging information for the crash dump feature, use the **debug voice dsp crash-dump** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voice dsp crash-dump [**details**| **keepalives**]

no debug voice dsp crash-dump [**details**| **keepalives**]

Syntax Description

details	(Optional) Displays debugging information for the crash dump feature details. There is no debug output until there is one DSP crash. When the crash dump feature is turned on, the detailed debug messages are displayed.
keepalives	(Optional) Displays debugging information for the crash dump feature keepalives. Confirms that a crash dump file has been written to the specified destination.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

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

Usage Guidelines

DSP resource management (DSPRM) sends a debug message to the console that confirms that a crash dump file has been written to the specified destination.

Caution Enabling this debug feature adds extra time for the DSP to recover. The **keepalive** and **detail** keywords produce large volumes of output and should not be used except under the direction of a Cisco engineer.

You can also use the **undebug all** command to disable debugging output.

Examples

The following example shows a debug message that confirms that a crash dump file has been written to the specified destination. The stack is displayed on the console, and the DSPware version, complexity (image set), and Cisco IOS software version is also displayed.

```
Router# debug voice dsp crash-dump keepalives
```

```
*Mar  8 03:42:19.505:Got back DSP status 0x12 0x0 for dsp 9 slot 1
*Mar  8 03:42:19.509:Got back DSP status 0x12 0x0 for dsp 10 slot 1
*Mar  8 03:42:19.509:Got back DSP status 0x12 0x0 for dsp 11 slot 1
*Mar  8 03:42:19.509:Got back DSP status 0x12 0x0 for dsp 12 slot 1
*Mar  8 03:42:19.509:Got back DSP status 0x12 0x0 for dsp 13 slot 1
*Mar  8 03:42:19.509:Got back DSP status 0x12 0x0 for dsp 14 slot 1
*Mar  8 03:42:21.509:status cleared done, dsp 9 slot 1
*Mar  8 03:42:21.513:status cleared done, dsp 10 slot 1
*Mar  8 03:42:21.513:status cleared done, dsp 11 slot 1
*Mar  8 03:42:21.513:status cleared done, dsp 12 slot 1
*Mar  8 03:42:21.513:status cleared done, dsp 13 slot 1
*Mar 8 03:42:21.513:status cleared done, dsp 14 slot 1
```

The following command disables all the debugging output on the screen to stop the output from the **debug voice dsp crash-dump keepalives** command:

```
Router# undebug all
```

The following example shows the **debug voice dsp crash-dump details** command entry when no DSP crash is present. There is no debugging output until there is one DSP crash. When the crash dump feature is turned on, the detailed debug messages are displayed.

```
Router# debug voice dsp crash-dump details
```

Related Commands

Command	Description
show voice dsp crash-dump	Displays voice dsp crash dump information.
voice dsp crash-dump	Enables the crash dump feature and specifies the destination file and the file limit.

debug voice eecm

To display debugging information for the Voice End-to-End Call Manager, use the **debug voice eecm** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voice eecm[*slot*/*port*]

no debug voice eecm[*slot*/*port*]

Syntax Description

slot / *port*	(Optional) Slot and port number of the voice port. If the slot and port arguments are entered, only debugging information for that voice port is displayed.

Command Modes

Privileged EXEC

Usage Guidelines

This command is valid on the Cisco MC3810 only.

Examples

The following is sample output from the **debug voice eecm** command:

```
Router# debug voice eecm
1/1: EECM(in), ST_NULL       EV_ALLOC_DSP
1/1: EECM(in), ST_DIGIT_COLLECT  EV_PARSE_DIGIT  3
1/1: EECM(in), ST_DIGIT_COLLECT  EV_PARSE_DIGIT  7
1/1: EECM(in), ST_DIGIT_COLLECT  EV_PARSE_DIGIT  0
1/1: EECM(in), ST_DIGIT_COLLECT  EV_PARSE_DIGIT  2
1/1: EECM(in), ST_ADDRESS_DONE   EV_OUT_SETUP
-1/-1: EECM(out), ST_NULL     EV_IN_SETUP
1/1: EECM(in), ST_OUT_REQUEST   EV_IN_PROCEED
1/2: EECM(out), ST_SEIZE      EV_ALLOC_DSP
1/2: EECM(out), ST_SEIZE      EV_OUT_ALERT
1/1: EECM(in), ST_OUT_REQUEST   EV_IN_ALERT
1/1: EECM(in), ST_OUT_REQUEST   EV_OUT_ALERT_ACK
1/2 EECM(out), ST_IN_PENDING   EV_OUT_CONNECT
1/1: EECM(in), ST_WAIT_FOR_ANSWER EV_IN_CONNECT
1/2: EECM(out), ST_ACTIVE     EV_OUT_REL
1/1: EECM(in), ST_ACTIVE      EV_IN_REL
1/1: EECM(in), ST_DISCONN_PENDING EV_OUT_REL_ACK
```

Related Commands

Command	Description
debug voice all	Displays debugging information for all components of the Voice Call Manager.
debug voice protocol	Displays debugging information for the Voice Line protocol State machine.
debug voice signaling	Displays debugging information for the voice port signaling.

Command	Description
debug voice tdsm	Displays debugging information for the voice tandem switch.
debug voice ccapi	Debugs the call control API.

debug voice enum

To view voice telephone number mapping (ENUM) information, use the **debug voice enum** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voice enum {**detail| summary**}

no debug voice enum {**detail| summary**}

Syntax Description

detail	Displays detailed output.
summary	Displays summary output.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(11)T	This command was introduced.

Usage Guidelines

Disable console logging and use buffered logging before using the **debug voice enum** command. Using the **debug voice enum** command generates a large volume of debugs, which can affect router performance.

Examples

The following is sample output from the **debug voice enum detail** command. The output shows the match number as 5108891234, ENUM table as 10. Rule 1 in table 10 matched the pattern and after applying the replacement rule, the resulting string is 5108891234. The ENUM query is sent out for the domain 4.3.2.1.9.8.8.0.1.5.e164.cisco.com. The output then shows the matching Naming Authority Pointer (NAPTR) records obtained in the response. The records are then processed and the final URLs (contact lists) are shown toward the end.

```
Router# debug voice enum detail
enum_resolve_domain:match_num 5108891234 table_indx 10
enum_resolve_domain:rule 1 result string 5108891234
generate_enum_search_string :search string 4.3.2.1.9.8.8.0.1.5.e164.cisco.com
enum_dns_query:name = 4.3.2.1.9.8.8.0.1.5.e164.cisco.com type = 35, ns_server = 0
order 100 pref 10 service sip+E2U flag U
regexp /^.*$/sip:5108891234@1.8.50.14/ replacement
order 200 pref 10 service h323+E2U flag U
regexp /^.*$/h323:5555@1.5.1.1/ replacement
num_elem = 2
NAPTR Record :order 100 pref 10 service sip+E2U
              flags U regexp /^.*$/sip:5108891234@1.8.50.14/
              replacement
NAPTR Record :order 200 pref 10 service h323+E2U
```

```
               flags U regexp /^.*$/h323:5555@1.5.1.1/
               replacement
decode_naptr_record :re_string ^.*$
decode_naptr_record :re_substitution_string sip:5108891234@1.8.50.14
decode_naptr_record :re_flags_string
U_FLAG case, stopping query
new_e164_user sip:5108891234@1.8.50.14
decode_naptr_record :re_string ^.*$
decode_naptr_re
tahoe13#cord :re_substitution_string h323:5555@1.5.1.1
decode_naptr_record :re_flags_string
U_FLAG case, stopping query
new_e164_user h323:5555@1.5.1.1
contact_list :
              sip:5108891234@1.8.50.14
contact_list :
              h323:5555@1.5.1.1
enum_resolve_domain:contact_list 64558450
```

A sample output of the **debug voice enum summary** command is shown below.

The output shows the matching number, the ENUM table used and the rule in the table that matched the number along with the resulting string. Note that this output is a subset of the output from **debug voice enum detail** command.

```
Router# debug voice enum summary
enum_resolve_domain:match_num 5108891234 table_indx 10
enum_resolve_domain:rule 1 result string 5108891234
```

The table below provides an alphabetical listing of the **debug voice enum** command fields and a description of each field.

Table 18: debug voice enum Field Descriptions

Field	Description
contact_list	Final list of URLs that the gateway will try to contact as an attempt to place the call.
flag	Flag value of a NAPTR record as defined in RFC 2915.
match_num	Number to be used for matching against the ENUM match table.
name	Fully qualified domain name sent out to DNS server.
ns_server	Address of the DNS server. If 0, the Domain Name System (DNS) server configured on the gateway is used.
num_elem	Number of records received in the response.
order	Order in the record, as defined in RFC 2915.
pref	Preference of the record, as defined in RFC 2915.
regexp	Regular expression of the record, as defined in RFC 2915.

Field	Description
replacement	Replacement string of the record, as defined in RFC 2915.
re_flags_string	Flag indicating whether matching and replacement should be case sensitive: • i = Case insensitive • otherwise = Case sensitive
re_string	The first part of the regexp, delimited by "/". This is used to match the incoming string. Refer to RFC 2915.
re_substitution_string	The second part of regexp, delimited by "/".
result string	String that results when match_num is taken through the ENUM match table for a match. This string will be used to form a fully qualified domain name (FQDN).
rule	Rule number that matched match_num in the enum match table.
search string	String sent out to the DNS server.
service	Service field of the NAPTR record. Refer to RFC 2915.
table_indx	Index of the ENUM match table picked for this call.
type	Type of record requested in the query: 35 = NAPTR 33 = DNS Service (SRV)

Related Commands

Command	Description
rule (ENUM configuration)	Defines the rule pattern for an ENUM match table.
show voice enum-match-table	Displays the ENUM match table rules.
test enum	Tests the ENUM match table rules.
voice enum-match-table	Initiates the ENUM match table definition.

debug voice fastpath

To turn on debugging to monitor voice fastpath activity, use the **debug voice fastpath**command in privileged EXEC mode. To turn off voice fastpath debugging, use the **no** form of this command.

debug voice fastpath[**invalidate**][*slot*/*port*]

no debug voice fastpath[**invalidate**][*slot*/*port*]

Syntax Description

invalidate	(Optional) Turns on debugging for fastpath cache invalidation.
slot / *port*	(Optional) Slot and port to be debugged. Slash mark is required.

Command Default

Voice fastpath debugging does not occur.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.4(21)	This command was introduced on the Cisco AS5400XM and AS5350XM.

Usage Guidelines

The **debug voice fastpath** command displays the details on every packet that is being switched via fastpath. The **debug voice fastpath invalidate** command displays the details of cache invalidation and cache update. The **debug voice fastpath** command and its options are interchangeable with the **debug voip fastpath** command.

Voice fastpath is enabled by default. In order to disable it, issue the **no voice-fastpath enable** command in global configuration mode.

When voice fastpath is enabled, the IP address and User Datagram Protocol (UDP) port number information for the logical channel that is opened for a specific call are cached. Voice fastpath prevents the RTP stream from reaching the application layer. Instead, the packets are forwarded at a lower layer to help reduce CPU utilization in high call-volume scenarios.

When supplementary services such as hold or transfer are used, voice fastpath causes the router to stream the audio to the cached IP address and UDP port. The new logical channel information (generated after a call on hold is resumed or after a transfer is completed) is disregarded. Traffic must go to the application layer constantly so that redefinition of the logical channel is considered and audio is streamed to the new IP address and UDP port pair. Therefore, be sure to disable voice-fastpath in order to support supplementary services.

Note The **debug voice fastpath** command should be enabled only when there is light traffic on the gateway. Enabling this command can affect the functionality of the gateway.

Examples

The following example shows how to turn on voice fastpath debugging, shows how to use the **show debug** command to display what debugging functions are enabled, and provides sample output for the debugging function:

```
Router# debug voice fastpath
Fastpath related debugging is on
Router# show debug
 fastpath:
  Fastpath related debugging is on
Router#
*Nov 14 08:22:35.971: NP VPD(2/01): pak sent via fastpath,part=0x652DEE80 ret=0x000003
len=32
*Nov 14 08:22:35.987: NP VPD(2/01): pak sent via fastpath,part=0x652DEEC0 ret=0x000003
len=32
*Nov 14 08:22:36.011: NP VPD(2/01): pak sent via fastpath,part=0x652DEF00 ret=0x000003
len=32
*Nov 14 08:22:36.031: NP VPD(2/01): pak sent via fastpath,part=0x652DEF40 ret=0x000003
len=32
*Nov 14 08:22:36.051: NP VPD(2/01): pak sent via fastpath,part=0x652DEF80 ret=0x000003
len=32
*Nov 14 08:22:36.071: NP VPD(2/01): pak sent via fastpath,part=0x652DEFC0 ret=0x000003
len=32
*Nov 14 08:22:36.095: NP VPD(2/01): pak sent via fastpath,part=0x652DF000 ret=0x000003
len=32
*Nov 14 08:22:36.111: NP VPD(2/01): pak sent via fastpath,part=0x652DF040 ret=0x000003
len=32
*Nov 14 08:22:36.131: NP VPD(2/01): pak sent via fastpath,part=0x652DF080 ret=0x000003
len=32
*Nov 14 08:22:36.151: NP VPD(2/01): pak sent via fastpath,part=0x652DF0C0 ret=0x000003
len=32
*Nov 14 08:22:36.171: NP VPD(2/01): pak sent via fastpath,part=0x652DF100 ret=0x000003
len=32
*Nov 14 08:22:36.195: NP VPD(2/01): pak sent via fastpath,part=0x652DF140 ret=0x000003
len=32
*Nov 14 08:22:36.207: NP VPD(2/01): pak sent via fastpath,part=0x652DF180 ret=0x000003
len=32
*Nov 14 08:22:36.231: NP VPD(2/01): pak sent via fastpath,part=0x652DF1C0 ret=0x000003
len=32
*Nov 14 08:22:36.251: NP VPD(2/01): pak sent via fastpath,part=0x652DF200 ret=0x000003
len=32
*Nov 14 08:22:36.271: NP VPD(2/01): pak sent via fastpath,part=0x652DF240 ret=0x000003
len=32
*Nov 14 08:22:36.291: NP VPD(2/01): pak sent via fastpath,part=0x652DF280 ret=0x000003
len=32
*Nov 14 08:22:36.315: NP VPD(2/01): pak sent via fastpath,part=0x652DF2C0 ret=0x000003
len=32
*Nov 14 08:22:36.331: NP VPD(2/01): pak sent via fastpath,part=0x652DF300 ret=0x000003
len=32
*Nov 14 08:22:36.351: NP VPD(2/01): pak sent via fastpath,part=0x652DF340 ret=0x000003
len=32
*Nov 14 08:22:36.371: NP VPD(2/01): pak sent via fastpath,part=0x652DF380 ret=0x000003
len=32
*Nov 14 08:22:36.391: NP VPD(2/01): pak sent via fastpath,part=0x652DF3C0 ret=0x000003
len=32
```

The following example shows how to use the **debug voice fastpath**command *slot*/*port* command to debug slot 2, port 13 on the router:

```
Router# debug voice fastpath 2/013
Fastpath related debugging is on
```

```
*Nov 14 08:28:00.623: NP VPD(2/13): pak sent via fastpath,part=0x652DFFC0 ret=0x000003
len=32
*Nov 14 08:28:00.643: NP VPD(2/13): pak sent via fastpath,part=0x652E0000 ret=0x000003
len=32
*Nov 14 08:28:00.659: NP VPD(2/13): pak sent via fastpath,part=0x652E0080 ret=0x000003
len=32
*Nov 14 08:28:00.831: NP VPD(2/13): pak sent via fastpath,part=0x652E0280 ret=0x000003
len=32
*Nov 14 08:28:00.855: NP VPD(2/13): pak sent via fastpath,part=0x652E0300 ret=0x000003
len=32
*Nov 14 08:28:00.867: NP VPD(2/13): pak sent via fastpath,part=0x652E0380 ret=0x000003
len=32
*Nov 14 08:28:01.031: NP VPD(2/13): pak sent via fastpath,part=0x652E0540 ret=0x000003
len=32
*Nov 14 08:28:01.051: NP VPD(2/13): pak sent via fastpath,part=0x652E0580 ret=0x000003
len=32
*Nov 14 08:28:01.075: NP VPD(2/13): pak sent via fastpath,part=0x652E0640 ret=0x000003
len=32
*Nov 14 08:28:01.231: NP VPD(2/13): pak sent via fastpath,part=0x652E0840 ret=0x000003
len=32
*Nov 14 08:28:01.251: NP VPD(2/13): pak sent via fastpath,part=0x652E07C0 ret=0x000003
len=32
*Nov 14 08:28:01.271: NP VPD(2/13): pak sent via fastpath,part=0x652E0900 ret=0x000003
len=32
*Nov 14 08:28:01.439: NP VPD(2/13): pak sent via fastpath,part=0x652E0AC0 ret=0x000003
len=32
*Nov 14 08:28:01.463: NP VPD(2/13): pak sent via fastpath,part=0x652E0B40 ret=0x000003
len=32
*Nov 14 08:28:01.483: NP VPD(2/13): pak sent via fastpath,part=0x652E0BC0 ret=0x000003
len=32
```

The following example shows how to enable debugging for fastpath cache invalidation on slot 2, port 17, and shows how to display sample output for the debugging function:

```
Router# debug voice fastpath invalidate 2/17

 Fastpath cache invalidation related  debugging is on
Router# show voice call summary
PORT            CODEC    VAD VTSP STATE             VPM STATE
=============== ======== === =================== ======================
6/4:0.20         g729r8    y  S_CONNECT             CSM_OC6_CONNECTED
6/4:0.21         g729r8    y  S_CONNECT             CSM_OC6_CONNECTED
Router# show spe | i a
Country code config : default T1 (u Law)
Country code setting: e1-default
Port state: (s)shutdown   (r)recovery  (t)test  (a)active call
            (b)busiedout  (d)download  (B)bad   (p)busyout pending
Call type : (m)modem      (d)digital   (v)voice (f)fax-relay       (_)not in use
Summary   :
Ports     : Total  540  In-use     2  Free   514  Disabled    24
Calls     : Modem    0  Digital    0  Voice    2  Fax-relay    0
                      SPE          SPE     SPE  SPE   Port         Call
SPE#    Port #        State        Busyout Shut Crash State        Type
2/02    0012-0017     ACTIVE             0    0     0 _____a       _____v
2/03    0018-0023     ACTIVE             0    0     0 a_____       v_____
Router# show logging

Syslog logging: enabled (274 messages dropped, 20 messages rate-limited,
                0 flushes, 0 overruns, xml disabled, filtering disabled)
    Console logging: disabled
    Monitor logging: level debugging, 0 messages logged, xml disabled,
                     filtering disabled
    Buffer logging: level debugging, 1018 messages logged, xml disabled,
                    filtering disabled
    Logging Exception size (8192 bytes)
    Count and timestamp logging messages: disabled
    Trap logging: level informational, 133 message lines logged

Log Buffer (1000000 bytes):
*Nov 14 08:40:36.499:  NP VPD (2/17): Cached header parameter values: header size : 28,
payload size : 13, ssrc : 0x24DB1F03, udp chksum : 0x0
*Nov 14 08:40:36.499:  NP VPD (2/17): Cached IP/UDP pkt details: dest-ip: 31.31.31.1
```

```
src-ip: 31.31.31.3   dport: 0x4070  sport: 0x43A6
*Nov 14 08:40:40.851:  NP VPD (2/17): Cached header parameter values: header size : 28,
payload size : 32, ssrc : 0x24DB1F03, udp chksum : 0x0
*Nov 14 08:40:40.851:  NP VPD (2/17): Cached IP/UDP pkt details: dest-ip: 31.31.31.1
src-ip: 31.31.31.3   dport: 0x4070  sport: 0x43A6
*Nov 14 08:40:40.939:  NP VPD (2/17): Cache being cleared due to change in payload size old
 payload size : 32  new rx payload size : 13 cached ssrc : 24DB1F03
```

Related Commands

Command	Description
debug voip fastpath	Turns on debugging to monitor VoIP fastpath packets.
show voice call	Displays the call status information for voice ports.
voice fastpath enable	Turns on voice fastpath.

debug voice h221

To debug telephony call control information, use the **debug voice h221**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voice h221 [**all**| **default**| **error** [**call [informational]**| **software [informational]**]| **function**| **individual**| **inout**| **raw [decode]**]

no debug voice h221

Syntax Description

all	(Optional) Enables all H.221 debugging, except the raw option.
default	(Optional) Activates function, inout, error call, and software debugging.
error	(Optional) Enables H.221 call error and software error debugging.
error [call]	(Optional) Enables H.221 major call processing error debugs related to the H.221 subsystem.
error [call [informational]]	(Optional) Enables H.221 major and informational call processing error debugs related to the H.221 subsystem.
error [software]	(Optional) Enables H.221 major software error debugs related to the H.221 subsystem.
error [software [informational]]	(Optional) Enables H.221 major and informational software error debugs related to the H.221 subsystem.
function	(Optional) Enables procedure tracing.
individual	(Optional) Activates individual H.221 debugging.
inout	(Optional) Enables subsystem inout debugging.
raw	(Optional) Displays raw BAS messages.
raw [decode]	(Optional) Decodes raw BAS data.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.4(11)T	This command was introduced.

Usage Guidelines

This command enables debugging for H.221 message events (voice telephony call control information).

Note This command provides the same results as the **debug voip h221** command.

Caution We recommend that you log the output from the **debug voice h221 all** command to a buffer, rather than sending the output to the console; otherwise, the size of the output could severely impact the performance of the gateway.

Use the **debug voice h221 individual** *x* command, (where *x* is an index number for a debug category), to activate a single debug, selected by index number instead of entering a group of debug commands. See the table below for a list of debug categories and corresponding index numbers.

Table 19: Indexes and Categories for the debug voice h221 individual command

Index Number	Debug Category
1, 2, 30, 31, 32	Secondary number exchange
5, 6, 14, 15, 16, 22	Audio mode/caps
7, 10, 12, 13, 17, 28	Video mode/caps
8, 9, 23	B-channel mode/caps
11, 24, 33	Miscellaneous command exchange
18	Bandwidth calculations
19, 20, 21	DSP configuration
3, 4, 25, 27, 42, 43	General caps/internal
26	Non-standard caps/command
29	Loop request
34, 35, 36, 37, 38, 39, 40, 41	BAS squelch

Examples

The raw keyword displays the raw BAS information coming from or to the DSP. It is displayed in a hexadecimal octet format. The **decode** option decodes the BAS information into a readable English format.

The following is sample output from the **debug voice h221 raw** decode command:

```
BAS=81:1 0 0 0 0 0 0 1: AUDIO CAPS=g711 a-law
BAS=82:1 0 0 0 0 0 1 0: AUDIO CAPS=g711 u-law
BAS=84:1 0 0 0 0 1 0 0: AUDIO CAPS=g722 48k
BAS=85:1 0 0 0 0 1 0 1: AUDIO CAPS=g728
BAS=F9:1 1 1 1 1 0 0 1: H.242 MBE start indication
BAS=02:0 0 0 0 0 0 1 0: H.242 MBE length=2
BAS=0A:0 0 0 0 1 0 1 0: H.242 MBE type=H.263 caps
BAS=8A:1 - - - - - - -: Always 1
BAS=8A:- 0 0 0 1 - - -: H.263 MPI=1
BAS=8A:- - - - - 0 1 -: H.263 FORMAT=h.263_cif
BAS=8A:- - - - - - - 0: No additional options
```

Related Commands

Command	Description
debug voip ccapi	Enables debugging for the call control application programming interface (CCAPI) contents.
debug voip rtp	Enables debugging for Real-Time Transport Protocol (RTP) named event packets.

debug voice h324

To debug video call control information, use the **debug voice h324**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voice h324 [**all**| **function**| **inout**| **default**| **individual** [*number*]| **message**| **error** [**software [informational]**| **call [informational]**]]

Syntax Description

all	(Optional) Enables all H.324 debugging except raw and raw decode.
default	(Optional) Activates function, inout, error call, and software debugging.
error	(Optional) Enables H.324 call error and software error debugging.
error [call]	(Optional) Enables H.324 major call processing error debugs related to the H.324 subsystem.
error [call [informational]]	(Optional) Enables H.324 major and informational call processing error debugs related to the H.324 subsystem.
error [software]	(Optional) Enables H.324 major software error debugs related to the H.324 subsystem.
error [software [informational]]	(Optional) Enables H.324 major and informational software error debugs related to the H.324 subsystem.
function	(Optional) Enables procedure tracing.
individual	(Optional) Activates individual H.324 debugging.
inout	(Optional) Enables subsystem inout debugging.
message	(Optional) Enables H.245 message display to/from H.324. Only displays message types, for message detail, use debug h245 asn1.
number	Index number. Number of debug category. See the table below.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.4(22)T	This command was introduced.

Usage Guidelines

This command enables debugging for H.324 message events (video call control information).

Note This command is the same as the **debug voip h324**command**.**

Caution We recommend that you log the output from the **debug voice h324 all** command to a buffer, rather than sending the output to the console; otherwise, the size of the output could severely impact the performance of the gateway.

Use the **debug voice h324 individual** *index-number* command, where *index number is*a debug category, to activate a single debug.

This is helpful when trying to see a specific problem, without having a large number of debug output being generated. For example, the user could select the command **debug voice h324 individual 4** to see calls where no video caps arrived from the IP side of the call (SIP to H.324 direction). Multiple debug output can be activated using this command, one at a time. These are not additional debug output to the ones enabled by the command **debug voice h324 all**, just another way to selectively see specific information, without generating large amounts of debug output.

Table 20: Index Numbers and Descriptions for the debug voice h324 Command

Index Number	Description
1	Shows incoming H.245 message type
2	Shows MSD master/slave determination upon receiving MSD from peer
3	Warns that no audio caps were found from IP leg (not necessarily an error).
4	Warns that no video caps were found from IP leg (not necessarily an error).
5	Shows MSD master/slave determination when sending MSDack.
6	Displays media type being sent (audio/video), when sending MES message.
7	Displays H.223 parameters when sending TCS.

Index Number	Description
8	Displays OLC information, when sending audio OLC.
9	Displays OLC information, when sending video OLC.
10	Displays OLCack information, when sending OLCack.
11	Displays OLCrej information, when sending OLCrej.
12	Displays digit begin sent, when sending USER INPUT message.
13-15	Displays internal status bits of h245 messages sent/received in the h324 subsystem. No user data is provided.
16	Displays master/slave determination when MSDack is received.
17	Displays media type when MESack is received.
18	Displays media type when MESrej is received.
19	Displays OLC information, when receiving audio OLC.
20	Displays OLC information, when receiving video OLC.
21	Displays media type when OLCack is received.
22	Displays media type when OLCrej is received.
23	Displays message type, when an H.245 miscellaneous message is received (for example FastVideoUpdate).
24	Displays digit begin received, when receiving USER INPUT message.
25	Displays message type, when an H.245 miscellaneous message is sent (for example FastVideoUpdate).
26	Displays outgoing message command type. No user data provided with this debug.
27	Displays the initial H.223 mux level received from the peer, reported by the DSP.

Index Number	Description
28	Displays information about either OLCack or OLCrej being sent in response to an OLC request.
29	Displays the audio codec being opened with the IP leg.
30	Displays the video codec being opened with the IP leg. Should always be the same as the video codec with the H.324 leg.
31	Displays when IOS is sending the DSP either the H.223 mux table, or AL information. No user data is provided.
32	Indicates the digit being sent to the IP leg, through the RFC 2833 procedure.
33-34	Displays the parameters being sent to the DSP to configure either audio or video.
35	Displays information about the H.223 multiplex table being sent to the DSP.
36	Displays information about the H.223 AL configuration being sent to the DSP.
37-38	Indicates message arriving from IP leg. No user data is provided.
39	Displays information when receiving VENDOR ID message. This may show the type of equipment being connected to on the H.324 leg, if the peer adds the information to the message.
40	Displays the new H.223 multiplex level being configured.
41	Displays the new H.223 maximum PDU size being configured.
42	Indicates when the internal video capability memory has been released. No user data is provided.
43	Indicates when an empty capability set (ECS) has arrived from the IP leg of the call.
44	Indicates when a new capability set has arrived from the IP leg after an ECS has arrived.

Index Number	Description
45	Displays the dynamic payload number from the IP leg (H.324 to IP direction).

debug voice mlpp

To display debugging information for the Multilevel Precedence and Preemption (MLPP) service, use the **debug voice mlpp**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voice mlpp [**all**| **default**| **detail**| **error**| **function**| **inout**]

no debug voice mlpp [**all**| **default**| **detail**| **error**| **function**| **inout**]

Syntax Description

all	(Optional) Enables all MLPP debugging.
default	(Optional) Enables error, function, and inout debugging. This is the default option if no keywords are used.
detail	(Optional) Displays detailed trace messages of the MLPP subsystem.
error	(Optional) Enables MLPP call error debugging.
function	(Optional) Enables tracing of the functions called by the MLPP subsystem.
inout	(Optional) Enables function in and out debugging.

Command Modes

Privileged EXEC (#)

Command History

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

Usage Guidelines

This command enables debugging for MLPP events.

Examples

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

- Ephone 1 is connected to ephone 3 (nonMLPP call).
- Ephone 4 makes an MLPP call to ephone 3. The preemption tone is played to both ephone 1 and 3.

- Ephone 3 is disconnected after the preemption tone timeout and precedence ringing.
- Ephone 3 answers the MLPP call and is connected to ephone 4.

```
Router# debug voice mlpp

Sep  5 14:24:49.492: //299/E4F8A0AE814C/VOIP-MLPP/voice_mlpp_call_update:
Sep  5 14:24:49.492: //299/E4F8A0AE814C/VOIP-MLPP/voice_mlpp_get_preemptInfo:
   Peer=20005
Sep  5 14:24:49.492: //299/E4F8A0AE814C/VOIP-MLPP/voice_mlpp_get_preemptInfo:
   mlpp_ephone_find_call is successful
Sep  5 14:24:49.492: //299/E4F8A0AE814C/VOIP-MLPP/voice_mlpp_get_preemptInfo:
   A mlpp channel is selected
 PeerTag[20005] preemptorCallID[299] preemptCallID[297]
Sep  5 14:24:49.496: //299/E4F8A0AE814C/VOIP-MLPP/voice_mlpp_call_add:
Sep  5 14:24:49.496: //299/E4F8A0AE814C/VOIP-MLPP/voice_mlpp_call_add:
   Unsupported Voice Interface Type; Interface Type=26DtPreemptionTone
Sep  5 14:24:54.500: //296/DD8862EE8146/VOIP-MLPP/voice_mlpp_resource_reserve_req:
    Call not preempted, No reservation necessary
Sep  5 14:24:54.500: //296/DD8862EE8146/VOIP-MLPP/voice_mlpp_call_delete:
   Unsupported Voice Interface Type; Interface Type=26
Sep  5 14:24:54.500: //297/DD8862EE8146/VOIP-MLPP/voice_mlpp_call_delete:
   Unsupported Voice Interface Type; Interface Type=26
Sep  5 14:24:54.508: //301/E4F8A0AE814C/VOIP-MLPP/voice_mlpp_call_add:
Sep  5 14:24:54.508: //301/E4F8A0AE814C/VOIP-MLPP/voice_mlpp_call_add:
   Unsupported Voice Interface Type; Interface Type=26
Sep  5 14:24:59.947: //301/E4F8A0AE814C/VOIP-MLPP/voice_mlpp_call_add:
Sep  5 14:24:59.947: //301/E4F8A0AE814C/VOIP-MLPP/voice_mlpp_call_add:
   Unsupported Voice Interface Type; Interface Type=26
Sep  5 14:24:59.951: //299/E4F8A0AE814C/VOIP-MLPP/voice_mlpp_call_add:
Sep  5 14:24:59.951: //299/E4F8A0AE814C/VOIP-MLPP/voice_mlpp_call_add:
   Unsupported Voice Interface Type; Interface Type=26
```

Related Commands

Command	Description
debug ephone mlpp	Displays debugging information for MLPP calls to phones in a Cisco Unified CME system.
mlpp indication	Enables MLPP indication on an SCCP phone or analog FXS port.
mlpp max-precedence	Sets the maximum precedence (priority) level that a phone user can specify when making an MLPP call.
mlpp preemption	Enables preemption capability on an SCCP phone or analog FXS port.

debug voice protocol

To display debugging information for the Voice Line protocol State machine, use the **debug voice protocol** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voice protocol[*slot*/*port*]

no debug voice protocol[*slot*/*port*]

Syntax Description

slot / *port*	(Optional) Slot and port number of the voice port. If the slot and port arguments are entered, only debugging information for that voice port is displayed.

Command Modes

Privileged EXEC

This command is valid on the Cisco MC3810 only.

Usage Guidelines

In the debugging display, the following abbreviations are used for the different signaling protocols:

- LFXS--FXS trunk loop start protocol
- LFXO--FXO trunk loop start protocol
- GFXS--FXS trunk ground start protocol
- GFXO--FXO trunk ground start protocol
- E&M--E&M trunk protocol

Examples

The following is sample output from the **debug voice protocol** command:

```
Router# debug voice protocol
Voice Line protocol State machine debugging is on
1/1: LFXS( ), idle gets event offhook
1/1: LFXS( ), idle ==> seize
1/1: LFXS(in), seize gets event ready
1/1: LFXS(in), seize ==> dial_tone
1/1: LFXS(in), dial_tone gets event digit
1/1: LFXS(in), dial_tone ==> collect
1/1: LFXS(in), collect gets event digit
1/1: LFXS(in), collect gets event digit
1/1: LFXS(in), collect gets event digit
1/1: LFXS(in), collect gets event addr_done
1/1: LFXS(in), collect ==> call_progress
1/2: LFXS( ), idle gets event seize
1/2: LFXS( ), idle ==> ringing
1/2: LFXS(out), ringing gets event dial_tone
1/2: LFXS(out), ringing gets event offhook
1/2: LFXS(out), ringing ==> connected
1/1: LFXS(in), call_progress gets event answer
1/1: LFXS(in), call_progress ==> connected
1/2: LFXS(out), connected gets event onhook
1/2: LFXS(out), connected ==> disconnect_wait
```

```
1/2: LFXS(out), disconnected_wait gets event disconnect
1/2: LFXS(out), disconnect_wait ==> cpc
1/1: LFXS(in), connected gets event disconnect
1/2: LFXS(out), connected ==> cpc
1/2: LFXS(out), cpc gets event offhook
1/2: LFXS(out), cpc gets event timer1
1/2: LFXS(out), cpc ==> cpc_recover
1/2: LFXS(out), cpc gets event timer1
1/2: LFXS(out), cpc_recover ==> offhook_wait
1/1: LFXS(in), offhook_wait gets event onhook
1/1: LFXS(in), offhook_wait ==> idle
1/2: LFXS(out), offhook_wait gets event onhook
1/2: LFXS(out), offhook_wait ==> idle
```

Related Commands

Command	Description
debug voice all	Displays debugging information for the voice tandem switch.
debug voice eecm	Displays debugging information for the Voice End-to-End Call Manager.
debug voice signaling	Displays debugging information for the voice port signaling.
debug voice tdsm	Displays debugging information for the voice tandem switch.
debug voice ccapi	Debugs the call control API.

debug voice register errors

To display debug information on voice register module errors during registration in a Cisco Unified CallManager Express (Cisco Unified CME) or Cisco Unified Session Initiation Protocol (SIP) Survivable Remote Site Telephony (SRST) environment, use the **debug voice register errors**command in privileged EXEC mode. To disable debugging, use the **no** form of the command.

debug voice register errors

no debug voice register errors

Syntax Description

This command has no arguments or keywords

Command Default

Disabled

Command Modes

Privileged EXEC mode

Command History

Cisco IOS Release	Modification
12.2(15)ZJ	This command was introduced for Cisco SIP SRST 3.0
12.3(4)T	This command was integrated into Cisco IOS Release 12.3(4)T for Cisco SIP SRST 3.0.
12.4(4)T	This command was added to Cisco Unified CME 3.4 and Cisco SIP SRST 3.4.

Usage Guidelines

Registration errors include failure to match pools or any internal errors that happen during registration.

Examples

Examples

The following is sample output for this command for a registration request with authentication enabled:

```
...
*May 6 18:07:26.971: VOICE_REG_POOL: Register request for (4901) from (10.5.49.83)
*May 6 18:07:26.971: VOICE_REG_POOL: key(9499C07A000036A3) added to nonce table
*May 6 18:07:26.975: VOICE_REG_POOL: Contact doesn't match any pools
*May 6 18:07:26.975: //4/89D7750A8005/SIP/Error/ccsip_spi_register_incoming_registration:
Registration Authorization failed with authorization header=
...
```

If there are no voice register pools configured for a particular registration request, the message “Contact doesn’t match any pools” is displayed.

When authentication is enabled and if the phone requesting registration cannot be authenticated, the message “Registration Authorization failed with authorization header” is displayed.

Examples

The following is sample output from this command:

```
Router# debug voice register errors
*Apr 22 11:52:54.523 PDT: VOICE_REG_POOL: Contact doesn't match any pools
*Apr 22 11:52:54.539 PDT: VOICE_REG_POOL: Register request for (33015) from (10.2.152.39)
*Apr 22 11:52:54.539 PDT: VOICE_REG_POOL: Contact doesn't match any pools.
*Apr 22 11:52:54.559 PDT: VOICE_REG_POOL: Register request for (33017) from (10.2.152.39)
*Apr 22 11:53:04.559 PDT: VOICE_REG_POOL: Maximum registration threshold for pool(3) hit
```

If there are no voice register pools configured for a particular registration request, the message "Contact doesn't match any pools" is displayed.

If the **max registrations** command is configured, when registration requests reach the maximum limit, the "Maximum registration threshold for pool (x) hit" message is displayed for the particular pool.

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

Table 21: debug voice register errors Field Descriptions

Field	Description
Contact (doesn't match any pools)	Contact refers to the location of the SIP devices and the IP address.
key (*MAC address*)	Unique MAC address of a locally available individual SIP phone used to support a degree of authentication in Cisco Unified CME.
Register request for (*telephone number*) from (*IP address*).	The unique key for each registration is the telephone number.
Registration Authorization (failed with authorization header)	Registration Authorization message is displayed when **authenticate** command is configured in Cisco Unified CME.

Related Commands

Command	Description
debug voice register events	Displays debug information on voice register module events during SIP phone registrations in a Cisco Unified CME or Cisco Unified SIP SRST environment.

debug voice register events

To display debug information on voice register module events during Session Initiation Protocol (SIP) phone registrations in a Cisco Unified CallManager Express (Cisco Unified CME) or Cisco Unified SIP Survivable Remote Site Telephony (SRST) environment, use the **debug voice register events** command in privileged EXEC mode. To disable debugging, use the **no** form of this command.

debug voice register events

no debug voice register events

Syntax Description

This command has no arguments or keywords

Command Default

Disabled

Command Modes

Privileged EXEC mode

Command History

Cisco IOS Release	Modification
12.2(15)ZJ	This command was introduced for Cisco SIP SRST 3.0
12.3(4)T	This command was integrated into Cisco IOS Release 12.3(4)T for Cisco SIP SRST 3.0.
12.4(4)T	This command was added to Cisco CME 3.4 and Cisco SIP SRST 3.4.

Usage Guidelines

Using the debug voice register events command should suffice to view registration activity.

Registration activity includes matching of pools, registration creation, and automatic creation of dial peers. For more details and error conditions, you can use the debug voice register errors command.

Cisco Unified CME

The following example shows output from this command:

```
*May 6 18:07:27.223: VOICE_REG_POOL: Register request for (4901) from (1.5.49.83)
*May 6 18:07:27.223: VOICE_REG_POOL: Contact matches pool 1 number list 1
*May 6 18:07:27.223: VOICE_REG_POOL: key(4901) contact(10.5.49.83) add to contact table
*May 6 18:07:27.223: VOICE_REG_POOL: No entry for (4901) found in contact table
*May 6 18:07:27.223: VOICE_REG_POOL: key(4901) contact(10.5.49.83) added to contact
tableVOICE_REG_POOL pool->tag(1), dn->tag(1), submask(1)
*May 6 18:07:27.223: VOICE_REG_POOL: Creating param container for dial-peer 40001.
*May 6 18:07:27.223: VOICE_REG_POOL: Created dial-peer entry of type 0
*May 6 18:07:27.223: VOICE_REG_POOL: Registration successful for 4901, registration id is
2
...
```

The phone number 4901 associated with voice register pool 1, voice register dn 1, registered successfully. A dynamic normal (type 0) VoIP dial peer has been created for entry 4901. The dial peer can be verified using the **show voice register dial-peers** and **show sip-ua status registrar**commands.

Cisco Unified SIP SRST

The following is sample output from this command:

```
Router# debug voice register events
Apr 22 10:50:21.731 PDT: VOICE_REG_POOL: Contact matches pool 1
Apr 22 10:50:21.731 PDT: VOICE_REG_POOL: key(91011) contact(192.168.0.2) add to contact
table
Apr 22 10:50:21.731 PDT: VOICE_REG_POOL: key(91011) exists in contact table
Apr 22 10:50:21.731 PDT: VOICE_REG_POOL: contact(192.168.0.2) exists in contact table, ref
 updated
Apr 22 10:50:21.731 PDT: VOICE_REG_POOL: Created dial-peer entry of type 1
Apr 22 10:50:21.731 PDT: VOICE_REG_POOL: Registration successful for 91011, registration
id is 257
```

The phone number 91011 registered successfully, and *type 1* is reported in the debug, which means that there is a preexisting VoIP dial peer.

```
Apr 22 10:50:38.119 PDT: VOICE_REG_POOL: Register request for (91021) from (192.168.0.3)
Apr 22 10:50:38.119 PDT: VOICE_REG_POOL: Contact matches pool 2
Apr 22 10:50:38.123 PDT: VOICE_REG_POOL: key(91021) contact(192.168.0.3) add to contact
table
Apr 22 10:50:38.123 PDT: VOICE_REG_POOL: key(91021) exists in contact table
Apr 22 10:50:38.123 PDT: VOICE_REG_POOL: contact(192.168.0.3) exists in contact table, ref
 updated
Apr 22 10:50:38.123 PDT: VOICE_REG_POOL: Created dial-peer entry of type 1
Apr 22 10:50:38.123 PDT: VOICE_REG_POOL: Registration successful for 91021, registration
id is 258
```

A dynamic VoIP dial peer has been created for entry 91021. The dial peer can be verified using the **show voice register dial-peers** and **show sip-ua status registrar**commands.

```
Apr 22 10:51:08.971 PDT: VOICE_REG_POOL: Register request for (95021) from (10.2.161.50)
Apr 22 10:51:08.971 PDT: VOICE_REG_POOL: Contact matches pool 3
Apr 22 10:51:08.971 PDT: VOICE_REG_POOL: key(95021) contact(10.2.161.50) add to contact
table
Apr 22 10:51:08.971 PDT: VOICE_REG_POOL: No entry for (95021) found in contact table
Apr 22 10:51:08.975 PDT: VOICE_REG_POOL: key(95021) contact(10.2.161.50) added to contact
table
Apr 22 10:51:08.979 PDT: VOICE_REG_POOL: Created dial-peer entry of type 0
Apr 22 10:51:08.979 PDT: VOICE_REG_POOL: Registration successful for 95021, registration
id is 259
Apr 22 10:51:09.019 PDT: VOICE_REG_POOL: Register request for (95012) from (10.2.161.50)
Apr 22 10:51:09.019 PDT: VOICE_REG_POOL: Contact matches pool 3
Apr 22 10:51:09.019 PDT: VOICE_REG_POOL: key(95012) contact(10.2.161.50) add to contact
table
Apr 22 10:51:09.019 PDT: VOICE_REG_POOL: No entry for (95012) found in contact table
Apr 22 10:51:09.023 PDT: VOICE_REG_POOL: key(95012) contact(10.2.161.50) added to contact
table
Apr 22 10:51:09.027 PDT: VOICE_REG_POOL: Created dial-peer entry of type 0
Apr 22 10:51:09.027 PDT: VOICE_REG_POOL: Registration successful for 95012, registration
id is 260
Apr 22 10:51:09.071 PDT: VOICE_REG_POOL: Register request for (95011) from (10.2.161.50)
Apr 22 10:51:09.071 PDT: VOICE_REG_POOL: Contact matches pool 3
Apr 22 10:51:09.071 PDT: VOICE_REG_POOL: key(95011) contact(10.2.161.50) add to contact
table
Apr 22 10:51:09.071 PDT: VOICE_REG_POOL: No entry for (95011) found in contact table
Apr 22 10:51:09.075 PDT: VOICE_REG_POOL: key(95011) contact(10.2.161.50) added to contact
table
Apr 22 10:51:09.079 PDT: VOICE_REG_POOL: Created dial-peer entry of type 0
Apr 22 10:51:09.079 PDT: VOICE_REG_POOL: Registration successful for 95011, registration
id is 261
Apr 22 10:51:09.123 PDT: VOICE_REG_POOL: Register request for (95500) from (10.2.161.50)
Apr 22 10:51:09.123 PDT: VOICE_REG_POOL: Contact matches pool 3
Apr 22 10:51:09.123 PDT: VOICE_REG_POOL: key(95500) contact(10.2.161.50) add to contact
table
```

```
Apr 22 10:51:09.123 PDT: VOICE_REG_POOL: No entry for (95500) found in contact table
Apr 22 10:51:09.127 PDT: VOICE_REG_POOL: key(95500) contact(10.2.161.50) added to contact
table
Apr 22 10:51:09.131 PDT: VOICE_REG_POOL: Created dial-peer entry of type 0
Apr 22 10:51:09.131 PDT: VOICE_REG_POOL: Registration successful for 95500, registration
id is 262
*Apr 22 11:52:54.523 PDT: VOICE_REG_POOL: Contact doesn't match any pools
*Apr 22 11:52:54.539 PDT: VOICE_REG_POOL: Register request for (33015) from (10.2.152.39)
*Apr 22 11:52:54.539 PDT: VOICE_REG_POOL: Contact doesn't match any pools
*Apr 22 11:52:54.559 PDT: VOICE_REG_POOL: Register request for (33017) from (10.2.152.39)
```

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

Table 22: debug voice register events Field Descriptions

Field	Description
Contact	Indicates the location of the SIP devices and may indicate the IP address.
contact table	The table that maintains the location of the SIP devices.
key	The phone number is used as the unique key to maintain registrations of SIP devices.
multiple contact	More than one registration matches the same phone number.
no entry	The incoming registration was not found.
type 0	Normal dial peer.
type 1	Existing normal dial peer.
type 2	Proxy dial peer.
type 3	Existing proxy dial peer.
type 4	Dial-plan dial peer.
type 5	Existing dial-plan dial peer.
type 6	Alias dial peer.
type 7	Existing alias dial peer.
un-registration successful	The incoming unregister was successful.
Register request/registration id *number*	The internal unique number for each registration; useful for debugging particular registrations.

Related Commands

Command	Description
debug voice register errors	Displays debug information on voice register module errors during registration in a Cisco Unified CME or Cisco Unified SIP SRST environment.
show sip-ua status registrar	Displays all the SIP endpoints that are currently registered with the contact address.
show voice register dial-peers	Displays details of Cisco Unified SIP SRST configuration and of all dynamically created VoIP dial peers.

debug voice signaling

To display debugging information for the voice port signaling, use the **debug voice signaling** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voice signalling[*slot*/*port*]

no debug voice signalling[*slot*/*port*]

Syntax Description

slot /*port*	(Optional) Slot and port number of the voice port. If the slot and port arguments are entered, only debugging information for that voice port is displayed.

Command Modes

Privileged EXEC

Usage Guidelines

This command is valid on the Cisco MC3810 only.

Examples

The following is sample output from the **debug voice signaling** command:

```
Router# debug voice signaling
1/1: TIU, report_local_hook=1
1/2: TIU, set ring cadence=1
1/2: TIU, ringer on
1/2: TIU, ringer off
1/2: TIU, ringer on
1/2: TIU, report_local_hook=1
1/2: TIU, turning off ringer due to SW ringtrip
1/2: TIU, ringer off
1/2: TIU, set ring cadence=0
1/2: TIU, ringer off
1/2: TIU, set reverse battery=1
1/2: TIU, set reverse battery=1
1/1: TIU, report_local_hook=0
1/2: TIU, set reverse battery=0
1/2: TIU, set loop disabled=1
1/1: TIU, set reverse battery=0
1/1: TIU, set loop disabled=1
1/2: TIU, report_local_hook=1
1/1: TIU, report_lead_gnd grounded=1
1/1: TIU, report_lead_gnd grounded=0
1/2: TIU, set loop disabled=0
1/1: TIU, set loop disabled=0
1/1: TIU, report_local_hook=0
1/2: TIU, report_local_hook=0
1/1: TIU, report_local_hook=1
1/2: TIU, report_local_hook=1
1/1: TIU, report_local_hook=0
1/2: TIU, report_local_hook=0
1/1: TIU, set reverse battery=0
1/2: TIU, set reverse battery=0
```

Related Commands

Command	Description
debug voice all	Displays debugging information for all components of the Voice Call Manager.
debug voice eecm	Displays debugging information for the Voice End-to-End Call Manager.
debug voice protocol	Displays debugging information for the Voice Line protocol State machine.
debug voice tdsm	Display debugging information for the voice tandem switch.
debug voice ccapi	Debugs the call control API.

debug voice source-group

To view voice source group information, use the **debug voice source-group** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voice source-group

no debug voice source-group

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(11)T	This command was introduced.

Usage Guidelines

Disable console logging and use buffered logging before using the **debug**voice source-group command. Using the **debug**voice source-group command generates a large volume of debugs, which can affect router performance.

Examples

A sample output of the **debug voice source-group** command is shown below.

The output shows that the hash table key for source ip group is 1.

```
00:30:49:SIPG:sipg_get() - idString=0x63BE1C28, hashkey=1
00:30:49:SIPG:sipg_find_key - hashkey=1,idstring=0x63BE1C28
```

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

Table 23: debug voice source-group Field Descriptions

Field	Description
hashkey	Hash table index of the source IP group.
idString	Value of the pointer to the source IP group name, which is used to make sure that it is not null.

Related Commands

Command	Description
carrier-id (voice source group)	Specifies the carrier handling incoming source VoIP calls (for carrier ID routing).
show voice source-group	Displays the details of one or more source IP groups.
test source-group	Tests the definition of a source IP group.
translation-profile (source group)	Associates a translation profile with the source IP group.
trunk-group-label (voice source group)	Specifies the trunk group handling incoming source VoIP calls (for trunk group label routing).
voice source-group	Initiates the source IP group definition.

debug voice statistics

To enable debugging of voice statistics, use the **debug voice statistics**command in privileged EXEC mode. To disable the debugging, use the **no** form of this command.

debug voice statistics {**csr**| **core**| **accounting**}

no debug voice statistics {**csr**| **core**| **accounting**}

Syntax Description

csr	Signaling voice call statistics records collection is debugged.
core	Generic statistics collection is debugged.
accounting	Voice accounting CSR collection is debugged.

Command Modes

Privileged EXEC

Command History

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

Examples

The following example shows the collection of records that is occurring in between intervals:

```
Router# debug voice statistics accounting

vstats_timer_handle_interval_event():Between Intervals!
04:52:37: vstats_acct_interval_end: interval_tag = 4
04:52:37: vstats_acct_interval_end: pushing out, tag=3
04:52:37: vstats_acct_clean_history_stats:
04:52:37: vstats_acct_clean_history_stats: stats (tag=3) not to be deleted
04:52:37: vstats_acct_clean_history_stats: stats (tag=2) not to be deleted
04:52:37: vstats_acct_create_empty_stats:
04:52:37: vstats_acct_create_new_rec_list:
04:52:37: vstats_acct_create_new_rec_list: add acct rec: methodlist=h323, acct-criteria=2
04:52:37: vstats_acct_create_new_rec:
04:52:37: vstats_acct_add_rec_entry:
04:52:37: vstats_acct_add_stats_entry:
04:52:37: vstat_push_driver_file_open():Cannot open
ftp://sgcp:sgcp@abc-pc:21//ftp_files/vstats.5400-GW.2003-02-13T162000Z.
errno=65540=Unknown error 65540
vstat_push_drv_activate_ftp_file_tx():open file
(ftp://sgcp:sgcp@jeremy-pc:21//ftp_files/vstats.5400-GW.2003-02-13T162000Z)=(ftp://sgcp:sg
cp@abc-pc:21//ftp_files/vstats.5400-GW.2003-02-13T162000Z)failed!
vstats_push_api_push_formatted_text():Start CMD error!
```

The following example shows a voice call going through the gateway:

```
Router# debug voice statistics csr
04:55:07: EM: Notify the producer not to produce
```

```
04:55:07: RADIUS(00000019): Storing nasport 0 in rad_db
04:55:07: RADIUS(00000019): Config NAS IP: 0.0.0.0
04:55:07: RADIUS(00000019): sending
04:55:07: RADIUS/ENCODE: Best Local IP-Address 1.6.43.101 for Radius-Server 1.6.10.203
04:55:07: RADIUS(00000019): Send Accounting-Request to 1.6.10.203:1646 id 21645/49,len 496
04:55:07: RADIUS:  authenticator C5 B8 AA 2E C3 AF 02 93 - 45 0B AE E5 B6 B2 99 1F
04:55:07: RADIUS:  Acct-Session-Id     [44]  10 "00000020"
04:55:07: RADIUS:  Vendor, Cisco       [26]  57
04:55:07: RADIUS:   h323-setup-time    [25]  51  "h323-setup-time=*16:22:30.994 UTC
          Thu Feb 13 2003"
04:55:07: RADIUS:  Vendor, Cisco       [26]  27
04:55:07: RADIUS:   h323-gw-id         [33]  21  "h323-gw-id=5400-GW."
04:55:07: RADIUS:  Vendor, Cisco       [26]  56
04:55:07: RADIUS:  Conf-Id             [24]  50  "h323-conf-id=2F4ED2E3 3EA611D7 800E0002
          B935C142"
04:55:07: RADIUS:  Vendor, Cisco       [26]  31
04:55:07: RADIUS:  h323-call-origin    [26]  25  "h323-call-origin=answer"
04:55:07: RADIUS:  Vendor, Cisco       [26]  32
04:55:07: RADIUS:   h323-call-type     [27]  26  "h323-call-type=Telephony"
04:55:07: RADIUS:  Vendor, Cisco       [26]  65
04:55:07: RADIUS:   Cisco AVpair       [1]   59  "h323-incoming-conf-id=2F4ED2E3 3EA611D7
          800E0002 B935C142"
04:55:07: RADIUS:  Vendor, Cisco       [26]  30
04:55:07: RADIUS:   Cisco AVpair       [1]   24  "subscriber=RegularLine"
04:55:07: RADIUS:  Vendor, Cisco       [26]  35
04:55:07: RADIUS:   Cisco AVpair       [1]   29  "gw-rxd-cdn=ton:0,npi:0,#:11"
04:55:07: RADIUS:  Vendor, Cisco       [26]  32
04:55:07: RADIUS:   Cisco AVpair       [1]   26  "calling-party-category=9"
04:55:07: RADIUS:  Vendor, Cisco       [26]  33
04:55:07: RADIUS:   Cisco AVpair       [1]   27  "transmission-medium-req=0"
04:55:07: RADIUS:  User-Name           [1]   4   "22"
04:55:07: RADIUS:  Acct-Status-Type    [40]  6   Start                     [1]
04:55:07: RADIUS:  NAS-Port-Type       [61]  6   Async                     [0]
04:55:07: RADIUS:  Vendor, Cisco       [26]  20
04:55:07: RADIUS:   cisco-nas-port     [2]   14  "ISDN 6/0:D:1"
04:55:07: RADIUS:  NAS-Port            [5]   6   0
04:55:07: RADIUS:  Calling-Station-Id  [31]  4   "22"
04:55:07: RADIUS:  Called-Station-Id   [30]  4   "11"
04:55:07: RADIUS:  Service-Type        [6]   6   Login                     [1]
04:55:07: RADIUS:  NAS-IP-Address      [4]   6   1.6.43.101
04:55:07: RADIUS:  Acct-Delay-Time     [41]  6   0
04:55:07: RADIUS(0000001A): Config NAS IP: 0.0.0.0
04:55:07: RADIUS(0000001A): sending
04:55:07: RADIUS/ENCODE: Best Local IP-Address 1.6.43.101 for Radius-Server 1.6.10.203
04:55:07: RADIUS(0000001A): Send Accounting-Request to 1.6.10.203:1646 id 21645/50, len427
04:55:07: RADIUS:  authenticator E4 98 06 8C 48 63 4F AA - 56 4F 40 12 33 F0 F5 99
04:55:07: RADIUS:  Acct-Session-Id     [44]  10  "00000021"
04:55:07: RADIUS:  Vendor, Cisco       [26]  57
04:55:07: RADIUS:   h323-setup-time    [25]  51  "h323-setup-time=*16:22:31.006 UTC
          Thu Feb 13 2003"
04:55:07: RADIUS:  Vendor, Cisco       [26]  27
04:55:07: RADIUS:   h323-gw-id         [33]  21  "h323-gw-id=5400-GW."
04:55:07: RADIUS:  Vendor, Cisco       [26]  56
04:55:07: RADIUS:   Conf-Id            [24]  50  "h323-conf-id=2F4ED2E3 3EA611D7 800E0002
          B935C142"
04:55:07: RADIUS:  Vendor, Cisco       [26]  34
04:55:07: RADIUS:   h323-call-origin   [26]  28 "h323-call-origin=originate"
04:55:07: RADIUS:  Vendor, Cisco       [26]  27
04:55:07: RADIUS:   h323-call-type     [27]  21  "h323-call-type=VoIP"
04:55:07: RADIUS:  Vendor, Cisco       [26]  65
04:55:07: RADIUS:   Cisco AVpair       [1]   59  "h323-incoming-conf-id=2F4ED2E3 3EA611D7
          800E0002 B935C142"
04:55:07: RADIUS:  Vendor, Cisco       [26]  30
04:55:07: RADIUS:   Cisco AVpair       [1]   24  "subscriber=RegularLine"
04:55:07: RADIUS:  Vendor, Cisco       [26]  30
04:55:07: RADIUS: Cisco AVpair       [1]   24  "session-protocol=cisco"
04:55:07: RADIUS:  Vendor, Cisco       [26]  35
04:55:07: RADIUS:   Cisco AVpair       [1]   29  "gw-rxd-cdn=ton:0,npi:0,#:11"
04:55:07: RADIUS:  User-Name           [1]   4   "22"
04:55:07: RADIUS:  Acct-Status-Type    [40]  6   Start                     [1]
04:55:07: RADIUS:  Calling-Station-Id  [31]  4   "22"
04:55:07: RADIUS:  Called-Station-Id   [30]  4   "11"
04:55:07: RADIUS:  Service-Type        [6]   6   Login                     [1]
```

```
04:55:07: RADIUS:  NAS-IP-Address      [4]   6   1.6.43.101
04:55:07: RADIUS:  Acct-Delay-Time     [41]  6   0
04:55:07: EM: No consumer registered for event type NEWINFO
04:55:07: EM: Notify the producer not to produce
04:55:07: EM: No consumer registered for event type NEWINFO
04:55:07: EM: Notify the producer not to produce
04:55:08: RADIUS: no sg in radius-timers: ctx 0x65BAB1BC sg 0x0000
04:55:08: RADIUS: Retransmit to (1.6.10.203:1645,1646) for id 21645/50
04:55:08: RADIUS: acct-delay-time for 403963FC (at 403965A1) now 1
04:55:09: RADIUS: no sg in radius-timers: ctx 0x65ADB8EC sg 0x0000
04:55:09: RADIUS: Retransmit to (1.6.10.203:1645,1646) for id 21645/49
04:55:09: RADIUS: acct-delay-time for 40389BFC (at 40389DE6) now 1
04:55:10: RADIUS: no sg in radius-timers: ctx 0x65BAB1BC sg 0x0000
04:55:10: RADIUS: Fail-over to (1.8.159.105:1645,1645) for id 21645/51
04:55:10: RADIUS: acct-delay-time for 403963FC (at 403965A1) now 2
04:55:10: RADIUS/ENCODE: Best Local IP-Address 1.6.43.101 for Radius-Server 1.8.159.105
04:55:10: RADIUS: Received from id 21645/53 1.8.159.105:1645, Accounting-response, len 20
04:55:10: RADIUS:  authenticator 57 EF DD 90 0F 88 76 EA - A5 3D A7 44 0D 90 66 16
04:55:10: vstats_acct_rsp_handler:   methodlist=h323, rsp_type=0x1
04:55:10:    acct_rsp_status=1 callid= 26, incoming=0, leg=2
04:55:10: vstats_acct_rsp_handler: last acct msg not sent yet. methodlist: h323
04:55:10: RADIUS: no sg in radius-timers: ctx 0x65ADB8EC sg 0x0000
04:55:10: RADIUS: Fail-over to (1.8.159.105:1645,1645) for id 21645/52
04:55:10: RADIUS: acct-delay-time for 40389BFC (at 40389DE6) now 2
04:55:10: RADIUS/ENCODE: Best Local IP-Address 1.6.43.101 for Radius-Server 1.8.159.105
04:55:10: RADIUS: Received from id 21645/54 1.8.159.105:1645, Accounting-response, len 20
04:55:10: RADIUS:  authenticator 97 88 6C BA DA 22 E7 5E - 73 EC 21 C6 36 1B 93 18
04:55:10: vstats_acct_rsp_handler:   methodlist=h323, rsp_type=0x1
04:55:10:    acct_rsp_status=callid= 25, incoming=1, leg=1
04:55:10: vstats_acct_rsp_handler: last acct msg not sent yet. methodlist: h323
04:55:13: RADIUS(0000001A): Config NAS IP: 0.0.0.0
04:55:13: RADIUS(0000001A): sending
04:55:13: RADIUS/ENCODE: Best Local IP-Address 1.6.43.101 for Radius-Server 1.6.10.203
04:55:13: RADIUS(0000001A): Send Accounting-Request to 1.6.10.203:1646 id 21645/55, len885
04:55:13: RADIUS:  authenticator F8 4F F1 30 7E 8B 5B 46 - EF AE 17 2D 5C BA 36 E5
04:55:13: RADIUS:  Acct-Session-Id     [44]  10  "00000021"
04:55:13: RADIUS:  Vendor, Cisco       [26]  57
04:55:13: RADIUS:   h323-setup-time    [25]  51  "h323-setup-time=*16:22:31.006 UTC
          Thu Feb 13 2003"
04:55:13: RADIUS:  Vendor, Cisco       [26]  27
04:55:13: RADIUS:   h323-gw-id         [33]  21  "h323-gw-id=5400-GW."
04:55:13: RADIUS:  Vendor, Cisco       [26]  56
04:55:13: RADIUS:   Conf-Id            [24]  50  "h323-conf-id=2F4ED2E3 3EA611D7 800E0002
          B935C142"
04:55:13: RADIUS:  Vendor, Cisco       [26]  34
04:55:13: RADIUS:   h323-call-origin   [26]  28  "h323-call-origin=originate"
04:55:13: RADIUS:  Vendor, Cisco       [26]  27
04:55:13: RADIUS:   h323-call-type     [27]  21  "h323-call-type=VoIP"
04:55:13: RADIUS:  Vendor, Cisco       [26]  65
04:55:13: RADIUS:   Cisco AVpair       [1]   59  "h323-incoming-conf-id=2F4ED2E3 3EA611D7
          800E0002 B935C142"
04:55:13: RADIUS:  Vendor, Cisco       [26]  30
04:55:13: RADIUS:   Cisco AVpair       [1]   24  "subscriber=RegularLine"
04:55:13: RADIUS:  Vendor, Cisco       [26]  30
04:55:13: RADIUS:   Cisco AVpair       [1]   24  "session-protocol=cisco"
04:55:13: RADIUS:  Vendor, Cisco       [26]  35
04:55:13: RADIUS:   Cisco AVpair       [1]   29  "gw-rxd-cdn=ton:0,npi:0,#:11"
04:55:13: RADIUS:  Vendor, Cisco       [26]  59
04:55:13: RADIUS:   h323-connect-time  [28]  53  "h323-connect-time=*16:22:31.046 UTC
          Thu Feb 13 2003"
04:55:13: RADIUS:  Acct-Input-Octets   [42]  6   2241
04:55:13: RADIUS:  Acct-Output-Octets  [43]  6   81
04:55:13: RADIUS:  Acct-Input-Packets  [47]  6   113
04:55:13: RADIUS:  Acct-Output-Packets [48]  6   5
04:55:13: RADIUS:  Acct-Session-Time   [46]  6   5
04:55:13: RADIUS:  Vendor, Cisco       [26]  62
04:55:13: RADIUS:   h323-disconnect-tim[29]  56  "h323-disconnect-time=*16:22:36.070 UTC
          Thu Feb 13 2003"
04:55:13: RADIUS:  Vendor, Cisco       [26]  32
04:55:13: RADIUS:   h323-disconnect-cau[30]  26  "h323-disconnect-cause=10"
04:55:13: RADIUS:  Vendor, Cisco       [26]  38
04:55:13: RADIUS:   h323-remote-address[23]  32  "h323-remote-address=14.0.0.110"
04:55:13: RADIUS:  Vendor, Cisco       [26]  24
```

```
04:55:13: RADIUS:   Cisco AVpair       [1]   18  "release-source=1"
04:55:13: RADIUS:  Vendor, Cisco       [26]  29
04:55:13: RADIUS:   h323-voice-quality [31]  23  "h323-voice-quality=-1"
04:55:13: RADIUS:  Vendor, Cisco       [26]  57
04:55:13: RADIUS:   Cisco AVpair       [1]   51  "alert-timepoint=*16:22:31.030 UTC
          Thu Feb 13 2003"
04:55:13: RADIUS:  Vendor, Cisco       [26]  39
04:55:13: RADIUS:   Cisco AVpair       [1]   33  "remote-media-address=14.0.0.110"
04:55:13: RADIUS:  Vendor, Cisco       [26]  44
04:55:13: RADIUS:   Cisco AVpair       [1]   38  "gw-final-xlated-cdn=ton:0,npi:0,#:11"
04:55:13: RADIUS:  Vendor, Cisco       [26]  44
04:55:13: RADIUS:   Cisco AVpair       [1]   38  "gw-final-xlated-cgn=ton:0,npi:1,#:22"
04:55:13: RADIUS:  User-Name           [1]   4   "22"
04:55:13: RADIUS:  Acct-Status-Type    [40]  6   Stop                       [2]
04:55:13: RADIUS:  Calling-Station-Id  [31]  4   "22"
04:55:13: RADIUS:  Called-Station-Id   [30]  4   "11"
04:55:13: RADIUS:  Service-Type        [6]   6   Login                      [1]
04:55:13: RADIUS:  NAS-IP-Address      [4]   6   1.6.43.101
04:55:13: RADIUS:  Acct-Delay-Time     [41]  6   0
04:55:13: RADIUS(00000019): Using existing nas_port 0
04:55:13: RADIUS(00000019):Config NAS IP: 0.0.0.0
04:55:13: RADIUS(00000019):sending
04:55:13: RADIUS/ENCODE: Best Local IP-Address 1.6.43.101 for Radius-Server 1.6.10.203
04:55:13: RADIUS(00000019): Send Accounting-Request to 1.6.10.203:1646 id 21645/56, len766
04:55:13: RADIUS:  authenticator 61 60 EB 92 29 5C DE B4 - CE 40 1C AB E3 A1 C8 F7
04:55:13: RADIUS:  Acct-Session-Id     [44]  10  "00000020"
04:55:13: RADIUS:  Vendor, Cisco       [26]  57
04:55:13: RADIUS:   h323-setup-time    [25]  51  "h323-setup-time=*16:22:30.994 UTC Thu
          Feb 13 2003"
04:55:13: RADIUS:  Vendor, Cisco       [26]  27
04:55:13: RADIUS:   h323-gw-id         [33]  21  "h323-gw-id=5400-GW."
04:55:13: RADIUS:  Vendor, Cisco       [26]  56
04:55:13: RADIUS:   Conf-Id            [24]  50  "h323-conf-id=2F4ED2E3 3EA611D7 800E0002
          B935C142"
04:55:13: RADIUS:  Vendor, Cisco       [26]  31
04:55:13: RADIUS:   h323-call-origin   [26]  25  "h323-call-origin=answer"
04:55:13: RADIUS:  Vendor, Cisco       [26]  32
04:55:13: RADIUS:   h323-call-type     [27]  26  "h323-call-type=Telephony"
04:55:13: RADIUS:  Vendor, Cisco       [26]  65
04:55:13: RADIUS:   Cisco AVpair       [1]   59  "h323-incoming-conf-id=2F4ED2E3 3EA611D7
          800E0002 B935C142"
04:55:13: RADIUS:  Vendor, Cisco       [26]  30
04:55:13: RADIUS:   Cisco AVpair       [1]   24  "subscriber=RegularLine"
04:55:13: RADIUS:  Vendor, Cisco       [26]  35
04:55:13: RADIUS:   Cisco AVpair       [1]   29  "gw-rxd-cdn=ton:0,npi:0,#:11"
04:55:13: RADIUS:  Vendor, Cisco       [26]  32
04:55:13: RADIUS:   Cisco AVpair       [1]   26  "calling-party-category=9"
04:55:13: RADIUS:  Vendor, Cisco       [26]  33
04:55:13: RADIUS:   Cisco AVpair       [1]   27  "transmission-medium-req=0"
04:55:13: RADIUS:  Vendor, Cisco       [26]  59
04:55:13: RADIUS:   h323-connect-time  [28]  53  "h323-connect-time=*16:22:31.046 UTC Thu
          Feb 13 2003"
04:55:13: RADIUS:  Acct-Input-Octets   [42]  6   81
04:55:13: RADIUS:  Acct-Output-Octets  [43]  6   2241
04:55:13: RADIUS:  Acct-Input-Packets  [47]  6   5
04:55:13: RADIUS:  Acct-Output-Packets [48]  6   113
04:55:13: RADIUS:  Acct-Session-Time   [46]  6   5
04:55:13: RADIUS:  Vendor, Cisco       [26]  62
04:55:13: RADIUS:   h323-disconnect-tim[29]  56  "h323-disconnect-time=*16:22:36.064 UTC
          Thu Feb 13 2003"
04:55:13: RADIUS:  Vendor, Cisco       [26]  32
04:55:13: RADIUS:   h323-disconnect-cau[30]  26  "h323-disconnect-cause=10"
04:55:13: RADIUS:  Vendor, Cisco       [26]  35
04:55:13: RADIUS:   Cisco AVpair       [1]   29  "h323-ivr-out=Tariff:Unknown"
04:55:13: RADIUS:  Vendor, Cisco       [26]  24
04:55:13: RADIUS:   Cisco AVpair       [1]   18  "release-source=1"
04:55:13: RADIUS:  Vendor, Cisco       [26]  28
04:55:13: RADIUS:   h323-voice-quality [31]  22  "h323-voice-quality=0"
04:55:13: RADIUS:  User-Name           [1]   4   "22"
04:55:13: RADIUS:  Acct-Status-Type    [40]  6   Stop [2]
04:55:13: RADIUS:  NAS-Port-Type       [61]  6   Async                      [0]
04:55:13: RADIUS:  Vendor, Cisco       [26]  20
04:55:13: RADIUS:   cisco-nas-port     [2]   14  "ISDN 6/0:D:1"
```

```
04:55:13: RADIUS:  NAS-Port            [5]   6   0
04:55:13: RADIUS:  Calling-Station-Id  [31]  4   "22"
04:55:13: RADIUS:  Called-Station-Id   [30]  4   "11"
04:55:13: RADIUS:  Service-Type        [6]   6   Login                        [1]
04:55:13: RADIUS:  NAS-IP-Addres       [4]   6   1.6.43.101
04:55:13: RADIUS:  Acct-Delay-Time     [41]  6   0
04:55:14: RADIUS: no sg in radius-timers: ctx 0x65BAB070 sg 0x0000
04:55:14: RADIUS: Retransmit to (1.6.10.203:1645,1646) for id 21645/55
04:55:14: RADIUS: acct-delay-time for 40553934 (at 40553CA3) now 1
04:55:14: RADIUS: no sg in radius-timers: ctx 0x65BA8284 sg 0x0000
04:55:14: RADIUS: Retransmit to (1.6.10.203:1645,1646) for id 21645/56
04:55:14: RADIUS: acct-delay-time for 405546C4 (at 405549BC) now 1
04:55:15: RADIUS: no sg in radius-timers: ctx 0x65BAB070 sg 0x0000
04:55:15: RADIUS: Fail-over to (1.8.159.105:1645,1645) for id 21645/57
04:55:15: RADIUS: acct-delay-time for 40553934 (at 40553CA3) now 2
04:55:15: RADIUS/ENCODE: Best Local IP-Address 1.6.43.101 for Radius-Server 1.8.159.105
04:55:15: RADIUS: no sg in radius-timers: ctx 0x65BA8284 sg 0x0000
04:55:15: RADIUS: Fail-over to (1.8.159.105:1645,1645) for id 21645/58
04:55:15: RADIUS: acct-delay-time for 405546C4 (at 405549BC) now 2
04:55:15: RADIUS/ENCODE: Best Local IP-Address 1.6.43.101 for Radius-Server 10.8.159.105
04:55:15: RADIUS: Received from id 21645/59 1.8.159.105:1645, Accounting-response, len 20
04:55:15: RADIUS:  authenticator B1 C4 5E FC DB FA 74 A4 - 05 E2 34 52 1A 11 26 06
04:55:15: vstats_acct_rsp_handler:   methodlist=h323, rsp_type=0x4
04:55:15: acct_rsp_status=1 callid= 26, incoming=0, leg=2
04:55:15: vstats_acct_rsp_handler: increment since-reset counter
04:55:15: vstats_acct_rsp_handler: increment interval counter
04:55:15: RADIUS: Received from id 21645/60 10.8.159.105:1645, Accounting-response, len 20
04:55:15: RADIUS:  authenticator 0E 70 74 2F E5 D8 EE 98 - B9 C0 DA 66 74 ED 84 77
04:55:15: vstats_acct_rsp_handler:   methodlist=h323, rsp_type=0x4
04:55:15: acct_rsp_status=1 callid= 25, incoming=1, leg=1
04:55:15: vstats_acct_rsp_handler: increment since-reset counter
04:55:15: vstats_acct_rsp_handler: increment interval counter
```

The following example shows the collection of records that is in between intervals:

```
Router# debug voice statistics accounting
Translating "abc-pc"...domain server (255.255.255.255)
vstats_timer_handle_interval_event():Between Intervals!
04:57:37: vstats_acct_interval_end: interval_tag = 5
04:57:37: vstats_acct_interval_end: pushing out, tag=4
04:57:37: vstats_acct_clean_history_stats:
04:57:37: vstats_acct_clean_history_stats: stats (tag=4) not to be deleted
04:57:37: vstats_acct_clean_history_stats: stats (tag=3) not to be deleted
04:57:37: vstats_acct_clean_history_stats: stats (tag=2) not to be deleted
04:57:37: vstats_acct_create_empty_stats:
04:57:37: vstats_acct_create_new_rec_list:
04:57:37: vstats_acct_create_new_rec_list: add acct rec: methodlist=h323, acct-criteria=2
04:57:37: vstats_acct_create_new_rec:
04:57:37: vstats_acct_add_rec_entry:
04:57:37: vstats_acct_add_stats_entry:
04:57:37: vstat_push_driver_file_open():Can not open
ftp://sgcp:sgcp@abc-pc:21//ftp_files/vstats.5400-GW.2003-02-13T162500Z.
errno=65540=Unknown error 65540
vstat_push_drv_activate_ftp_file_tx():open file
(ftp://sgcp:sgcp@abc-pc:21//ftp_files/vstats.5400-GW.2003-02-13T162500Z)=(ftp://sgcp:sgcp@
abc-pc:21//ftp_files/vstats.5400-GW.2003-02-13T162500Z) failed!
vstats_push_api_push_formatted_text():Start CMD error!
```

Related Commands

Command	Description
debug event-manager	Enables debugging of the event manager.

debug voice tdsm

To display debugging information for the voice tandem switch, use the **debug voice tdsm** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voice tdsm[*slot*/*port*]

no debug voice tdsm[*slot*/*port*]

Syntax Description

slot / *port*	(Optional) Slot and port number of the voice port. If the *slot* and *port* arguments are entered, only debugging information for that voice port is displayed.

Command Modes

Privileged EXEC

Usage Guidelines

This command is valid on the Cisco MC3810 only.

Examples

The following is sample output from the **debug voice tdsm** command:

```
Router# debug voice tdsm
Voice tandem switch debugging is on
-1/-1: TDSM(out), ref= -1, state NULL gets event OUT_SETUP
1/1: TDSM(in), ref=6, state CALL_INITIATED gets event IN_CALLPROC
1/1: TDSM(in), ref=6, state OUTG_CALLPROC gets event IN_ALERTING
1/1: TDSM(in), ref=6, state CALL_DELIVERED gets event IN_CONNECT
1/1: TDSM(out),ref=6, state CALL_ACTIVE send out conn. ack
1/1: TDSM(out),ref=6, state CALL_ACTIVE send out release, cause LOCAL_ONHOOK
1/1: TDSM(in), ref=6, state RELEASE_REQ gets event IN_REL_COMP, cause REMOTE_ONHOOK
-1/-1: TDSM(in), ref=-1, state NULL gets event IN_SETUP
-1/-1: TDSM(out), ref=6, state INC_CALLPROC gets event OUT_ALERTING
1/1: TDSM(out),ref=6, state CALL_RECEIVED gets event OUT_CONNECT
1/1: TDSM(in), ref-6, state CONNECT_REQ gets event IN_CONN_ACK
1/1: TDSM(out),ref-6, state CALL_ACTIVE send out release, cause LOCAL_ONHOOK
1/1: TDSM(in), ref=6, state RELEASE_REQ gets event IN_REL_COMP, cause REMOTE_ONHOOK
-1/-1:TDSM(out), ref=-1, state NULL gets event OUT_SETUP
1/1: TDSM(in), ref=7, state CALL_INITIATED gets event IN_CALLPROC
1/1: TDSM(in), ref=7, state OUTG_CALLPROC gets event IN_ALERTING
1/1: TDSM(in), ref=7, state CALL_DELIVERED gets event IN_CONNECT
1/1: TDSM(out),ref=7, state CALL_ACTIVE send out conn.ack
1/1: TDSM(out),ref=7, state CALL_ACTIVE send out release, cause LOCAL_ONHOOK
-1/-1: TDSM(in), ref=-1, state NULL gets event IN_SETUP
-1/-1: TDSM(out), ref=7, state INC_CALLPROC gets event OUT_ALERTING
1/1: TDSM(out),ref=7. state CALL_RECEIVED gets event OUT_CONNECT
1/1: TDSM(in), ref=7, state CONNECT_REQ gets event IN_CONN_ACK
1/1: TDSM(in), ref=7, state CALL_ACTIVE send out release, cause LOCAL_ONHOOK
1/1: TDSM(in), ref=7, state RELEASE_REQ gets event IN_REL_COMP, cause REMOTE_ONHOOK
-1/-1: TDSM(out), ref=-1, state NULL gets event OUT_SETUP
1/1: TDSM(in), ref=8, state CALL_INITIATED gets event IN_CALLPROC
1/1: TDSM(in), ref=8, state OUTG_CALLPROC gets event IN_ALERTINGbug all
```

Related Commands

Command	Description
debug voice all	Displays debugging information for all components of the Voice Call Manager.
debug voice eecm	Displays debugging information for the Voice End-to-End Call Manager.
debug voice protocol	Displays debugging information for the Voice Line protocol State machine.
debug voice signaling	Displays debugging information for the voice port signaling.
debug voice ccapi	Debugs the call control API.

debug voice translation

To view voice translation rule information, use the **debug voice translation** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voice translation

no debug voice translation

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(11)T	This command was introduced.

Usage Guidelines

Disable console logging and use buffered logging before using the **debug**voice translation command. Using the **debug**voice translation command generates a large volume of debugs, which can affect router performance.

Examples

Sample output from the **debug voice translation** command is shown below. The output shows the details of the original number following “regxrule_profile_translate”.

Following “regxrule_profile_match”, the output shows that rule 1 in the translation rule 1001 was a match; then the details of the SED substitution are shown.

Then the output shows the details of the translated number following “regxrule_profile_translate”.

In this example, because there was no called number or redirect number translation configured on the translation profile, corresponding errors were generated with a message that no match was found.

Following “regxrule_dp_translate”, the output indicates that there is no translation profile for outgoing direction, then it prints the numbers sent to the outgoing SPI.

```
Router#
00:51:56:regxrule_get_profile_from_trunkgroup:Voice port 0x64143DA8 does not belong to any
 trunk group
00:51:56:regxrule_get_profile_from_trunkgroup:Voice port 0x64143DA8 does not belong to any
 trunk group
00:51:56:regxrule_stack_pop_RegXruleNumInfo:stack=0x63DECAF4; count=1
00:51:56:regxrule_stack_push_RegXruleNumInfo:stack=0x63DECAF4; count=0
00:51:56:regxrule_profile_translate:number=4088880101 type=unknown plan=unknown
numbertype=calling
00:51:56:regxrule_profile_match:Matched with rule 1 in ruleset 1001
00:51:56:regxrule_profile_match:Matched with rule 1 in ruleset 1001
00:51:56:sed_subst:Successful substitution; pattern=4088880101 matchPattern=^.*
replacePattern=5551212 replaced pattern=5551212
```

```
00:51:56:regxrule_subst_num_type:Match Type = none, Replace Type = none Input Type = unknown
00:51:56:regxrule_subst_num_plan:Match Plan = none, Replace Plan = none Input Plan = unknown
00:51:56:regxrule_profile_translate:xlt_number=5551212 xlt_type=unknown xlt_plan=unknown
00:51:56:regxrule_profile_translate:number= type=UNKNOWN plan=UNKNOWN
numbertype=redirect-called
00:51:56:regxrule_get_RegXrule:Invalid translation ruleset tag=0
00:51:56:regxrule_profile_match:Error:ruleset for redirect-called number not found
00:51:56:regxrule_profile_translate:No match:number= type=UNKNOWN plan=UNKNOWN
00:51:56:regxrule_profile_translate:number=5108880101 type=unknown plan=unknown
numbertype=called
00:51:56:regxrule_get_RegXrule:Invalid translation ruleset tag=0
00:51:56:regxrule_profile_match:Error:ruleset for called number not found
00:51:56:regxrule_profile_translate:No match:number=5108880101 type=unknown plan=unknown
00:51:56:regxrule_stack_push_RegXruleNumInfo:stack=0x63DECAF4; count=1
00:51:56:regxrule_dp_translate:No profile found in peer 5108888 for outgoing direction
00:51:56:regxrule_dp_translate:calling_number=5551212 calling_octet=0x0
        called_number=5108880101 called_octet=0x80
        redirect_number= redirect_type=4294967295 redirect_plan=4294967295
00:51:56:regxrule_stack_pop_RegXruleNumInfo:stack=0x63DECAF4; count=2
00:51:56:regxrule_stack_push_RegXruleNumInfo:stack=0x63DECAF4; count=1
```

The table below provides an alphabetical listing of the **debug voice translation** command fields and a description of each field.

Table 24: debug voice translation Field Descriptions

Field	Description
called_number	Called number dialed number identification service (DNIS).
called_octet	Octect3 of called IE.
calling_number	Calling number automatic number identifier (ANI).
calling_octect	Octect3 of calling IE.
count	Number of elements in the translation stack.
Input Plan	Numbering plan of the input.
Input Type	Numbering type of the input.
matchPattern	Regular exp used for matching.
Match Plan	Numbering plan in the translation rule.
Match Type	Numbering type in the translation rule.
number	Incoming number for translation.
numbertype	Type of number: calling, called, or redirect.
pattern	Input string to the regular expression for matching.
plan	Numbering plan.
redirect_number	Redirect number.

Field	Description
redirect_plan	Numbering plan in the redirect number.
redirect_type	Numbering type in the redirect number.
replaced pattern	Final string after applying replacement rule of translation rule.
replacePattern	Replacement pattern in the translation rule.
Replace Plan	Replacement numbering plan in the translation rule.
Replace Type	Replacement numbering type in the translation rule.
stack	Value of the translation rule stack.
tag	Tag of the translation rule.
type	Numbering type in the translation rule.
xlt_number	Number after translation.
xlt_plan	Numbering plan after translation.
xlt_type	Numbering type after translation.

Related Commands

Command	Description
rule (voice translation-rule)	Defines the translation rule parameters for matching and replacing call number patterns.
show voice translation-rule	Displays a voice translation rule.
test voice translation-rule	Tests a voice translation rule.
voice translation-rule	Initiates the translation rule definition.

debug voice uri

To display debugging messages for uniform resource identifier (URI) voice classes, use the **debug voice uri** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voice uri

no debug voice uri

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

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

Usage Guidelines

Use this command to see which URI voice class and dial peer is matched for a Session Initiation Protocol (SIP) or telephone (TEL) URI.

Examples

The following examples show output from the **debug voice uri** command. Comments are imbedded in the examples.

```
Router# debug voice uri
```

Examples

The following output displays when an outbound dial peer match fails for the URL sip:9991234@sip.tgw.com?Subject=sip_e164_headers_plus.tcl&AccountInfo=12345&Priority=Urgent

```
*Jul 11 05:20:44.759:vuri_match_class:tag (767)
```

The first dial peer in the list is 767, which contains the **destination uri 767** command. 767 is a TELURI class, so it does not match the above URL.

```
*Jul 11 05:20:44.759:vuri_match_class:tag (766)
```

The next dial peer, 766, contains the **destination uri 766** command.

```
*Jul 11
05:20:44.759:vuri_match_class_sip:sip:9991234@sip.tgw.com?Subject=sip_e164_headers_plus.tcl&AccountInfo=12345&Priority=Urgent
 did not match pattern
```

766 is a SIP URI class and contains only the **pattern** command. The regex does not match the pattern, so there is no match.

```
*Jul 11 05:20:44.759:vuri_match_class:tag (999)
```
The next dial peer, 999, contains the **destination uri 999** command.

```
*Jul 11 05:20:44.759:vuri_match_class_sip:Match with phone context
*Jul 11 05:20:44.759:vuri_match_class_sip:input ()
```
If the **phone context** command is not present in the URI class, it is skipped.

```
*Jul 11 05:20:44.759:vuri_match_class_sip:Match with host
*Jul 11 05:20:44.759:vuri_match_class_sip:input (sip.tgw.com)
```
If the **host** command is not present in the URI class, it is skipped.

```
*Jul 11 05:20:44.759:vuri_match_class_sip:Match with user-id
*Jul 11 05:20:44.759:vuri_match_class_sip:input (9991234)
```
Try to match with the **user-id** portion of the URL, which is 9991234.

```
*Jul 11 05:20:44.759:vuri_match_class_sip:Match failed
```
The configured pattern, however, is “driver” and it is not a match.

Examples

The following debugging output is displayed when an outbound dial peer match is done for the URL sip:driver@cisco.com:

```
*Jul 11 06:06:30.119:vuri_match_class:tag (767)
```
The first dial peer in the list is 767 and it contains the **destination uri 767** command. 767 is a TEL URI class, so it does not match with the above URL.

```
*Jul 11 06:06:30.119:vuri_match_class:tag (766)
```
The next dial peer, 766, contains the **destination uri 766** command. Verify if the URL matches URI class 766.

```
*Jul 11 06:06:30.119:vuri_match_class_sip:sip:driver@cisco.com did not match pattern
```
The URL does not match with the **pattern** command.

```
*Jul 11 06:06:30.119:vuri_match_class:tag (999)
```
The next dial peer, 999, contains the **destination uri 999** command.

```
*Jul 11 06:06:30.119:vuri_match_class_sip:Match with phone context
*Jul 11 06:06:30.119:vuri_match_class_sip:input ()
```
If the **phone context** command is not configured, it is skipped.

```
*Jul 11 06:06:30.119:vuri_match_class_sip:Match with host
*Jul 11 06:06:30.119:vuri_match_class_sip:input (cisco.com)
```
If the **host** command is not configured under the class, it is skipped.

```
*Jul 11 06:06:30.119:vuri_match_class_sip:Match with user-id
*Jul 11 06:06:30.119:vuri_match_class_sip:input (driver)
```
If the **user-id** command is not configured, the user-id portion from the URL is “driver.”

```
*Jul 11 06:06:30.119:vuri_match_class_sip:driver matched; match length (6)
```
There is a match with the configured pattern. The number of characters that matched is 6.

Related Commands

Command	Description
destination uri	Specifies the voice class used to match the dial peer to the destination URI for an outgoing call.
incoming uri	Specifies the voice class that a VoIP dial peer uses to match the URI of an incoming call.
show dialplan incall uri	Displays which dial peer is matched for a specific URI in an incoming call.
show dialplan uri	Displays which outbound dial peer is matched for a specific destination URI.
voice class uri	Creates or modifies a voice class for matching dial peers to a SIP or TEL URI.

debug voice vofr

To show Cisco trunk and FRF.11 trunk call setup attempts and to show which dial peer is used in the call setup, use the **debug voice vofr**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voice vofr

no debug voice vofr

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(3)XG	This command was introduced.

Usage Guidelines

This command applies to Cisco trunks and FRF.11 trunks only; it does not apply to switched calls.

This command applies to VoFR, VoATM, and VoHDLC dial peers on the Cisco MC3810 device.

Examples

The following example shows sample output from the **debug voice vofr** command for a Cisco trunk:

```
Router# debug voice vofr
1d05h: 1/1:VOFR, unconf ==> pending_start
1d05h: 1/1:VOFR,create VOFR
1d05h: 1/1:VOFR,search dial-peer 7100 preference 0
1d05h: 1/1:VOFR, pending_start ==> start
1d05h: 1/1:VOFR,
1d05h:voice_configure_perm_svc:
1d05h:dial-peer 7100 codec = G729A payload size = 30 vad = off dtmf relay = on
   seq num = off
1d05h:voice-port 1/1 codec = G729A payload size = 30 vad = off dtmf relay = on
   seq num = off
1d05h: 1/1:VOFR,SIGNAL-TYPE = cept
1d05h:init_frf11 tcid 0 master 0 signaltype 2
1d05h:Going Out Of Service on tcid 0 with sig state 0001
1d05h: 1/1:VOFR, start get event idle
1d05h: 1/1:VOFR, start get event
1d05h: 1/1:VOFR, start get event set up
1d05h: 1/1:VOFR, start ==> pending_connect
1d05h: 1/1:VOFR, pending_connect get event connect
1d05h: 1/1:VOFR, pending_connect ==> connect
1d05h: 1/1:VOFR,SIGNAL-TYPE = cept
1d05h:init_frf11 tcid 0 master 1 signaltype 2
1d05h:start_vofr_polling on port 0 signaltype 2
```

The following example shows sample output from the **debug voice vofr** command for an FRF.11 trunk:

```
Router# debug voice vofr
1d05h: 1/1:VOFR,search dial-peer 7200 preference 2
1d05h: 1/1:VOFR,SIGNAL-TYPE = cept
```

```
1d05h:Launch Voice Trunk:signal-type 2
1d05h:calculated bandwidth = 10, coding = 6, size = 30
1d05h:%Voice-port 1/1 is down.
1d05h: 1/1:VOFR, pending_start get event idle
1d05h:Codec Type = 6 Payload Size = 30 Seq# off
1d05h:%Voice-port 1/1 is up.
1d05h:init_frf11 tcid 0 master 1 signaltype 2
1d05h:status OK :cid = 100
1d05h: 1/1:VOFR,
1d05h:start FRF11
1d05h: 1/1:VOFR, pending_start ==> frf11
1d05h: 1/1:VOFR,SIGNAL-TYPE = cept
```

Related Commands

Command	Description
debug ccfrf11 session	Displays the ccfrf11 function calls during call setup and teardown.
debug ccsip all	Displays the ccswvoice function calls during call setup and teardown.
debug ccswvoice vofr-session	Displays the ccswvoice function calls during call setup and teardown.
debug frame-relay fragment	Displays information related to Frame Relay fragmentation on a PVC.
debug vpm error	Displays the behavior of the Holst state machine.
debug vtsp port	Displays the behavior of the VTSP state machine.
debug vtsp vofr subframe	Displays the first 10 bytes (including header) of selected VoFR subframes for the interface.

debug voip aaa

To enable debugging messages for gateway authentication, authorization, and accounting (AAA) to be sent to the system console, use the **debug voip aaa** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voip aaa

no debug voip aaa

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
11.3(6)NA2	This command was introduced.
12.2(2)XB	This command was implemented on the Cisco AS5850 in the Cisco IOS Release 12.2(2)XB.
12.2(11)T	This command was integrated into Cisco IOS Release 12.2(11)T.

Examples

The following is sample output from the **debug voip aaa** command:

```
Router# debug voip aaa
VoIP AAA debugging is enabled
Router# show debug
voip aaa:
  voip aaa debugging is on
```

debug voip ais

To enable debugging of the application information system (AIS) database, use the **debug voip ais** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voip ais

no debug voip ais

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

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

Examples

The following is sample output from the **debug voip ais** command:

```
Router# debug voip ais

voip AIS debugging is on
Router#
*Jul 18 22:18:30.947: ais_appinst_create_record: new app inst record is created for sid=10,
 app_name=generic, stats:avail, elog:avail
*Jul 18 22:18:30.947: ais_appinst_insert_record_to_active: app inst sid=A is inserted to
active tree
*Jul 18 22:18:30.963: ais_be_server_get_record_from_url: Incoming url =
tftp://172.19.139.245/audio/ch_welcome.au
*Jul 18 22:18:30.963: ais_be_server_get_record_from_url: Found server name or ip =
172.19.139.245
*Jul 18 22:18:30.963: ais_be_server_get_record_from_url: AIS BE server record located
(6644ECCC)
Router#
*Jul 18 22:18:46.468: ais_be_server_get_record_from_url: Incoming url =
tftp://172.19.139.245/audio/ch_welcome.au
*Jul 18 22:18:46.468: ais_be_server_get_record_from_url: Found server name or ip =
172.19.139.245
*Jul 18 22:18:46.468: ais_be_server_get_record_from_url: AIS BE server record located
(6644ECCC)
Router#
*Jul 18 22:18:51.520: ais_be_server_get_record_from_url: Incoming url =
tftp://172.19.139.245/audio/ch_welcome.au
*Jul 18 22:18:51.520: ais_be_server_get_record_from_url: Found server name or ip =
172.19.139.245
*Jul 18 22:18:51.520: ais_be_server_get_record_from_url: AIS BE server record located
(6644ECCC)
Router#
*Jul 18 22:18:56.573: ais_be_server_get_record_from_url: Incoming url =
tftp://172.19.139.245/audio/ch_welcome.au
*Jul 18 22:18:56.573: ais_be_server_get_record_from_url: Found server name or ip =
172.19.139.245
*Jul 18 22:18:56.573: ais_be_server_get_record_from_url: AIS BE server record located
(6644ECCC)
Router#
```

```
*Jul 18 22:19:01.625: ais_be_server_get_record_from_url: Incoming url =
tftp://172.19.139.245/audio/ch_welcome.au
*Jul 18 22:19:01.625: ais_be_server_get_record_from_url: Found server name or ip =
172.19.139.245
*Jul 18 22:19:01.625: ais_be_server_get_record_from_url: AIS BE server record located
(6644ECCC)
*Jul 18 22:19:01.949: propagate_history_stats: stats for app inst 10 is propagated to
application (generic) and gateway level
*Jul 18 22:19:01.949: ais_appinst_move_record_active_to_history: session record (sid=A) is
 moved to history repository
```

Related Commands

Command	Description
call application event-log	Enables event logging for voice application instances.
call application stats	Enables statistics collection for voice applications.
debug voip event-log	Enables debugging of the event log module.

debug voip application

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

debug voip application [**accounting**| **all**| **callfeature**| **callsetup**| **core**| **datastruct**| **digitcollect**| **error**| **linking**| **media** [**packet**| **state**]| **oodrefer**| **redirect**| **script**| **session**| **settlement**| **states**| **supplementary-service**| **tclcommands**]

no debug voip application [**accounting**| **all**| **callfeature**| **callsetup**| **core**| **datastruct**| **digitcollect**| **error**| **linking**| **media** [**packet**| **state**]| **oodrefer**| **redirect**| **script**| **session**| **settlement**| **states**| **supplementary-service**| **tclcommands**]

Syntax Description

accounting	(Optional) Displays Voice over IP (VoIP) accounting messages.
all	(Optional) Displays all application debug messages.
callfeature	(Optional) Displays call feature debugs.
callsetup	(Optional) Displays the call setup being processed.
core	(Optional) Displays debug messages for the Application Framework (AFW) core libraries.
datastruct	(Optional) Displays debug messages for AFW data structures.
digitcollect	(Optional) Displays digits collected during the call.
error	(Optional) Displays application errors.
linking	(Optional) Displays script linking debugs.
media	(Optional) Displays debug traces for application media events.
oodrefer	(Optional) Displays debug messages for the Out-of-Dialog REFER (OOD-R) feature.
redirect	(Optional) Displays call redirection handler debugs.
script	(Optional) Displays script debugs.
session	(Optional) Displays default session application debugs.

settlement	(Optional) Displays debug messages for application settlement activities.
states	(Optional) Displays debug traces for application states.
supplementary-service	(Optional) Provides application layer tracing related to the processing of supplementary services requests.
tclcommands	(Optional) Displays debug messages for Tool Command Language (Tcl) commands used in application scripts.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(15)ZJT	This command was introduced.
12.3(4)T	This command was integrated into Cisco IOS Release 12.3(4)T. This command replaces the **debug voip ivr applib , debug voip ivr callsetup, debug voip ivr digitcollect, debug voip ivr redirect,**and **debug voip ivr supplementary-service** commands.
12.3(7)T	Reporting of H.450 capabilities was introduced.
12.4(4)T	The **callfeature** keyword was added.
12.4(4)XC	The **accounting** keyword was added.
12.4(9)T	The **accounting** keyword was integrated into Cisco IOS Release 12.4(9)T
12.4(11)XJ	The **oodrefer** keyword was added.
12.4(15)T	The **oodrefer** keyword was integrated into Cisco IOS Release 12.4(15)T.

Usage Guidelines

If you do not use any keywords, the **debug voip application** command displays application programming interface (API) libraries being processed.

The **debug voip application all** command differs from the **debug voip ivr all** command. The **debug voip application all**command enables all application framework debugs. The **debug voip ivr all**command enables both Application Framework Session debugs and interactive voice response (IVR) debugs.

Examples

The following is sample output from the **debug voip appli cation callsetup** command:

```
Router# debug voip application callsetup
ivr call setup debugging is on
Router#
*Mar  7 22:08:40.032://7//APPL:/afsSettlementValidateCall:target=, tokenp=0x0
*Mar  7 22:08:41.864://-1//PCM :LP:HN23A698CC:HN23A691A4:/InitiateCallSetup:Mode 1
RedirectMode 6 Incoming leg[-1] AlertTime -1 Destinations(1) [ 405  ]
*Mar  7 22:08:41.868://-1//PCM :HN23A698D0:/InitiateCallSetup:Destination 0 guid
:231D511B.1A5F11CC.800BB191.E9DE175D
*Mar  7 22:08:41.868: incoming_guid :00000000.00000000.00000000.00000000
*Mar  7 22:08:41.868://-1//PCM :HN23A698D0:/DNInitiate:Destination[405]
*Mar  7 22:08:41.868://-1//PCM :HN23A698D0:/DNMatchDialPeer:
*Mar  7 22:08:41.868: src carrier id:, tgt carrier id:
*Mar  7 22:08:41.868://-1//PCM :HN23A698D0:/DNQueuePeers:Matched peers(1)
*Mar  7 22:08:41.868://-1//PCM :HN23A698D0:/DNSetupPeer: Destination 0x6221092C
*Mar  7 22:08:41.872://-1//PCM :HN23A698D0:/DNSetupPeer:dialpeer tags for Rotary =  400
*Mar  7 22:08:41.872://-1//PCM :HN23A698D0:/DNSetupPeer:
*Mar  7 22:08:41.872:Destination SetupPeer cid(-1), destPat(405), match(2), prefix(),
peer(630D95B0)
*Mar  7 22:08:41.872://-1//PCM :HN23A698D0:/DNSettlementMatrixCheck:retcode=1 cid(-1)
trans=0x0, provider=0 No settle-call present
*Mar  7 22:08:41.940://8//PCM
:/DNHandler:(DN_SETTING[2])--(CC_EV_CALL_PROCEEDING[25])--IGNORED-->>(DN_SETTING[2])
*Mar  7 22:08:41.940://8//PCM :/CS_Setting_PROCEED:
*Mar  7 22:08:41.940://8//PCM :/CSPopLegAndWait:
*Mar  7 22:08:41.940://8//PCM :/CallSetupHandler:(CS_SETTING[0])
-----(CS_EV_PROCEEDING[3])------->>>(CS_SETTING[0])
*Mar  7 22:08:41.948://-1//PCM :HN23A698CC:/CSInterceptEvent:CallSetup[0x6320B998]
handlercount=1 Waits=1 #Objects=2 (CS_SETTING)
*Mar  7 22:08:41.948://8//PCM :/CSInterceptEvent:(CS_SETTING[0]) intercepting CS_EV_PROGRESS
 leg 8 (Mask=12)
*Mar  7 22:08:41.948://-1//PCM :HN23A698CC:/CSInterceptEvent:CallSetup[0x6320B998]
handlercount=1 Waits=1 #Objects=2 (CS_SETTING)
*Mar  7 22:08:41.952://-1//PCM :HN23A698CC:/CallSetupContinueEvent:CallSetup[0x6320B998]
handlercount=1 Waits=1 #Objects=2 (CS_SETTING)
*Mar  7 22:08:41.956://8//PCM :/CS_CutProgress:
*Mar  7 22:08:41.956://8//PCM :/CSPopLegAndWait:
*Mar  7 22:08:41.956://8//PCM :/CallSetupContinueEvent:(CS_SETTING[0])
-----(CS_EV_PROGRESS[15])------->>>(CS_CONFEDALERT[5])
*Mar  7 22:08:41.956://-1//PCM :HN23A698CC:/CallSetupHandleQueueEvents:
*Mar  7 22:08:41.956://8//PCM :/CallSetupContinueEvent: ***  Leaving function
***CallSetup[0x6320B998] handlercount=1 Waits=1 #Objects=2 (CS_CONFEDALERT)
*Mar  7 22:08:43.864://8//PCM :/CS_ConfedAlert_CONNECTED:no of Destinations:1
*Mar  7 22:08:43.864://-1//PCM :HN23A698CC:/CSDiscReturnAndEmptyLegALL:
*Mar  7 22:08:43.864://8//PCM :/CSPopLegAndWait:
*Mar  7 22:08:43.864://-1//PCM :HN23A698CC:/CSReturnIFDone:CallSetup[0x6320B998]
handlercount=1 Waits=1 #Objects=2 (CS_CONFED)
*Mar  7 22:08:43.864:CallSetupDump:CallSetup[0x6320B998] State:CS_CONFED[3] #Handler=1
#Waits=1#Objects=2
*Mar  7 22:08:43.864:All Destinations:
*Mar  7 22:08:43.864:DestinationDump:Destination[0x6221092C]:DN_SETTING[2] Holding[0] Leg[8]
*Mar  7 22:08:43.864:settlement_in_use:0 settlement_transaction:0x0 settlement_provider:0
settlement_type:0 settlement_callvalid:1 busyRotary:0
*Mar  7 22:08:43.864:WaitList of Destinations:{HAN[DN_HAND ][CS_HAND ]         ( )}
*Mar  7 22:08:43.868:Handler Tree{HAN[CS_HAND ][AFS_HAND]        ( HAN[DN_HAND][CS_HAND
][FALSE] [UC=1 ]LEG[8      ][CS_HAND ][LEG_OUTCONNECTED(8)][Cause(0)][UC=1 ])}
*Mar  7 22:08:43.868:{HAN[DN_HAND ][CS_HAND ]       ( )}
*Mar  7 22:08:43.868:
*Mar  7 22:08:43.868:Handler Tree Trace
*Mar  7 22:08:43.868://-1//PCM :HN23A698D0:/DNCleanup:Terminate=TRUE Status DN_SUCCESS
Leg[8]
*Mar  7 22:08:43.868://-1//PCM :HN23A698D0:/DNSettlementCleanup:cid(-1) trans=0x0, provider=0
*Mar  7 22:08:43.868://-1//PCM :HN23A698D0:/DNSetFree:
*Mar  7 22:08:43.868://-1//PCM :HN23A698CC:/CSReturnIFDone:CallSetup[0x6320B998]
handlercount=0 #Waits=0 #Objects=1 (CS_CONFED)
*Mar  7 22:08:43.872://-1//PCM :HN23A698CC:/CSReturnIFDone: decoupled extern connection
*Mar  7 22:08:43.872://-1//PCM :HN23A698CC:/CSReturnIFDone:CallSetup Returning(ls_000 Status
 CS_ACTIVE)
```

```
*Mar  7 22:08:43.872://8//PCM :/CallSetupHandler:(CS_CONFEDALERT[5])
-----(CS_EV_CONNECTED[5])------->>>(CS_CONFED[3])
*Mar  7 22:08:43.872://-1//PCM :HN23A698CC:/CallSetupCleanup:Terminate=FALSE
*Mar  7 22:08:43.876://-1//PCM :HN23A698CC:/CallSetupCleanup:State CS_CONFED
```

The following is sample output from the **debug voip application digitcollect** command:

```
Router# debug voip application digitcollect
ivr digit collect debugging is on
Router#
*Mar  7 22:09:08.108://9//DCM :/DigitCollect:DialPlan=TRUE AbortKey= TermKey=# NumPatts=0
             Enable=FALSE InterruptPrompt=FALSE maxDigits=0 DialPlanTerm=FALSE
*Mar  7 22:09:08.108://9//APPL:/AppTypeAheadGetDigit:no chars in buffer.
*Mar  7 22:09:08.112://9//DCM :/act_DCRunning_RDone:callid=9 Enable succeeded.enable=0
matchDialplan=1 numPatterns=0matchDialplanTerm=0
*Mar  7 22:09:11.428://9//APPL:/AppVcrControlEvent:VCR Control, not enabled.---
*Mar  7 22:09:11.428://9//APPL:/AppTypeAheadEvent:Passing, not enabled.---
*Mar  7 22:09:11.428://9//DCM :/act_DCRunning_Digit::pLeg 9 Digit 4 Tone Mode 0
*Mar  7 22:09:11.428://9//DCM :/DCTreatDigit:
*Mar  7 22:09:11.428://-1//DCM :HN23A6FF50:/DCTreatDigit:
*Mar  7 22:09:11.428: src carrier id:, tgt carrier id:
*Mar  7 22:09:11.428://-1//DCM :HN23A6FF50:/DCTreatDigit:Match single infotype
*Mar  7 22:09:11.676://9//APPL:/AppVcrControlEvent:VCR Control, not enabled.---
*Mar  7 22:09:11.676://9//APPL:/AppTypeAheadEvent:Passing, not enabled.---
*Mar  7 22:09:11.676://9//DCM :/act_DCRunning_Digit::pLeg 9 Digit 0 Tone Mode 0
*Mar  7 22:09:11.676://9//DCM :/DCTreatDigit:
*Mar  7 22:09:11.680://-1//DCM :HN23A6FF50:/DCTreatDigit:
*Mar  7 22:09:11.680: src carrier id:, tgt carrier id:
*Mar  7 22:09:11.680://-1//DCM :HN23A6FF50:/DCTreatDigit:Match single infotype
*Mar  7 22:09:11.908://9//APPL:/AppVcrControlEvent:VCR Control, not enabled.---
*Mar  7 22:09:11.908://9//APPL:/AppTypeAheadEvent:Passing, not enabled.---
*Mar  7 22:09:11.908://9//DCM :/act_DCRunning_Digit::pLeg 9 Digit 5 Tone Mode 0
*Mar  7 22:09:11.908://9//DCM :/DCTreatDigit:
*Mar  7 22:09:11.908://-1//DCM :HN23A6FF50:/DCTreatDigit:
*Mar  7 22:09:11.908: src carrier id:, tgt carrier id:
*Mar  7 22:09:11.908://-1//DCM :HN23A6FF50:/DCTreatDigit:Match single infotype
*Mar  7 22:09:11.912://9//DCM :/act_DCRunning_RDone:callid=9 Reporting disabled.
*Mar  7 22:09:11.912://-1//DCM :HN23A6FF50:/DigitCollectComplete:Status 4=DC_MATCHED_DIALPLAN.
 Digits=405
*Mar  7 22:09:11.916://-1//DCM :HN23A6FF50:/DCHandlerCleanup:
```

The following is sample output from the **debug voip application session**command:

```
Router# debug voip application session

applib session debugging is on
*Apr  4 23:57:08.054://-1//APPL:HN04B2BC78:LG35:/AFS_CALLSETUPIND:Calling #(4155550154),
Called #(52984), peer_tag(1)
*Apr  4 23:57:08.054://-1//APPL:HN04B2BC78:LG35:/afsSetupCall:Called #(52984)
*Apr  4
23:57:08.058://-1//APPL:HN04B2BC78:LG35:/afsMsgHandler:(CALLINIT)--(CC_EV_CALL_SETUP_IND)-->>(CONTACTINGDEST)
*Apr  4 23:57:08.466://-1//APPL:HN04B2BC78:LG36:/AFS_ContactingDest_ALERT:
*Apr  4 23:57:08.470://-1//APPL:HN04B2BC78:LG36:/AFS_ContactingDest_ALERT:inID(35), outID(36),
 outbnd peer_tag(6), prog_ind(8)
*Apr  4
23:57:08.470://-1//APPL:HN04B2BC78:LG36:/afsMsgHandler:(CONTACTINGDEST)--(CC_EV_CALL_ALERT)-->>(CONFINGALERT)
*Apr  4 23:57:08.470://-1//APPL:HN04B2BC78:CN11:/AFS_ConfingAlert_CREATEDONE:
{HAN[AFS_HAND][NULL    ]     ( LEG[35     ][AFS_HAND][LEG_INCALERTING(4)][Cause(0)][UC=1
]HAN[CS_HAND][AFS_HAND][FALSE] [UC=1 ]LEG[36     ][CS_HAND ][LEG_OUTINIT(6)][Cause(0)][UC=1
 ]CON[11     ][AFS_HAND][CONNECTION_CONFED(2)] [UC=1 ])}
*Apr  4
23:57:08.470://-1//APPL:HN04B2BC78:CN11:/afsMsgHandler:(CONFINGALERT)--(CC_EV_CONF_CREATE_DONE)-->>(CONFEDALERT)
*Apr  4
23:57:08.478://-1//APPL:HN04B2BC78:LG35:/afsMsgHandler:(CONFEDALERT)--(CC_EV_VOICE_MODE_DONE)-->>(CONFEDALERT)
*Apr  4 23:57:24.162://-1//APPL:HN04B2BC78:HN04B2BC78:/AFS_ConfedAlert_SETUPDONE:
*Apr  4 23:57:24.162://-1//APPL:HN04B2BC78:HN04B2BC78:/afsAppHandlerCleanup:CS_HAND
*Apr  4
23:57:24.162://-1//APPL:HN04B2BC78:/afsMsgHandler:(CONFEDALERT)--(APP_EV_CALLSETUP_DONE)-->>(CALLACTIVE)
*Apr  4
23:57:24.182://-1//APPL:HN04B2BC78:LG35:/afsMsgHandler:(CALLACTIVE)--(CC_EV_VOICE_MODE_DONE)-->>(CALLACTIVE)
*Apr  4 23:57:34.838://-1//APPL:HN04B2BC78:LG35:/AFS_DISCONNECT:
*Apr  4
23:57:34.838://-1//APPL:HN04B2BC78:LG35:/afsMsgHandler:(CALLACTIVE)--(CC_EV_CALL_DISCONNECTED)-->>(CALLDISCONNECT)
```

```
*Apr  4 23:57:34.838://-1//APPL:/afsCallProcess: [HANDLERDONE_EVENT_END]
*Apr  4 23:57:34.838://-1//APPL:HN04B2BC78:/afsHNDCleanup:Terminate TRUE Terminated
FALSE{HAN[AFS_HAND][NULL    ]     ( LEG[35
][AFS_HAND][LEG_INCCONNECTED(5)][Cause(16)][UC=1 ]LEG[36
][AFS_HAND][LEG_OUTCONNECTED(8)][Cause(0)][UC=1 ]CON[11    ][AFS_HAND][CONNECTION_CONFED(2)]
 [UC=1 ])}
*Apr  4 23:57:34.838://-1//APPL:HN04B2BC78:CN11:/afsMsgHandler:(CC_EV_CONF_DESTROY_DONE)
*Apr  4 23:57:34.854://-1//APPL:HN04B2BC78:LG35:/afsMsgHandler:(CC_EV_CALL_DISCONNECT_DONE)
*Apr  4 23:57:34.862://-1//APPL:HN04B2BC78:LG36:/afsMsgHandler:(CC_EV_CALL_DISCONNECT_DONE)
*Apr  4 23:57:34.862://-1//APPL:/afsCallProcess: [HANDLERDONE_EVENT_END]
*Apr  4 23:57:34.862://-1//APPL:HN04B2BC78:/afsHNDCleanup:Terminate TRUE Terminated
TRUE{HAN[AFS_HAND][NULL    ]    ( )}
*Apr  4 23:57:34.862://-1//APPL:HN04B32530:/afsFreeHND:Hndlr returned to the free queue
```

The following sample output shows an inbound call on a system with H.450.2, H.450.3, and H.450.12 capabilities enabled:

```
Router# debug voip application supplementary-service

supplementary service debugging is on
Jan 21 01:12:21.433://-1//APPL:/SSProcessH450CommonInfoEvent: CI_INFORM featureList=0xC0000000
 featureValue[0][0] featureControl=0x0
Jan 21 01:12:21.433://-1//APPL:/AppStoreCommonInfoToLeg:Leg peer_tag=8100
Jan 21 01:12:21.433://-1//APPL:/AppStoreCommonInfoToLeg:Received ciInform, store
ss_support=0xE000 to leg.
Jan 21 01:12:21.433://-1//APPL:/AppPrepareCommonInfo:Not voip dialpeer, no common info sent.
Jan 21 01:12:21.437://-1//APPL:/AppPrepareCommonInfoRequestReceived:Leg peer_tag=8100
Jan 21 01:12:21.437://-1//APPL:/AppPrepareCommonInfo:Global H450_2=1 H450_3=1 H450_12_ADV=1
 H450_12_USAGE=1
Jan 21 01:12:21.437://-1//APPL:/AppPrepareCommonInfoContent:SS_CI ss_evt=18
featureList=0xC0000000 featureValues=[0][0][0][0] featureControl=0x0
```

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

Table 25: debug voip application Field Descriptions

Field	Description
Called #	Called # may not appear in the initial /AFS_CALLSETUPIND message; it appears later in the /afsSetupCall message.
peer_tag	Dial peer tag.
/afsFreeHND	Verifies that the application completed properly.
H450_2	A value of 0 indicates that H.450.2 capabilities are disabled. A value of 1 indicates that H.450.2 capabilities are enabled.
H450_3	A value of 0 indicates that H.450.3 capabilities are disabled. A value of 1 indicates that H.450.3 capabilities are enabled.
H450_12_ADV= 0 and H450_12_USAGE = 0	H.450.12 capabilities are disabled.
H450_12_ADV= 1 and H450_12_USAGE = 0	H.450.12 capabilities are enabled in advertise-only mode.
H450_12_ADV= 1 and H450_12_USAGE = 1	H.450.12 capabilities are enabled.

The following is sample output from the **debug voip application accounting**command:

```
Router# debug voip application accounting
*Jan  6 19:34:22.535: //-1//Dest:/DestSetup:
*Jan  6 19:34:22.535: :DestSetup iw inc guid is 0-0-0-0
*Jan  6 19:34:22.535: //-1//Dest:/DestSetup:
*Jan  6 19:34:22.535: :DestSetup iw guid is 45AB9E05-7E2211DA-8088D216-195F6285
*Jan  6 19:34:22.535: :DestSetup iw guid is 45AB9E05-7E2211DA-8088D216-195F6285
*Jan  6 19:34:22.539: //-1//Dest:/DestSetup:
*Jan  6 19:34:22.539: :DestSetup setup inc guid is 0-0-0-0
*Jan  6 19:34:22.539: //-1//Dest:/DestSetup:
*Jan  6 19:34:22.539: :DestSetup setup guid is 45AB9E05-7E2211DA-8088D216-195F6285
*Jan  6 19:34:45.667: //-1//Dest:/DestSetup:
*Jan  6 19:34:45.667: :DestSetup iw inc guid is 0-0-0-0
*Jan  6 19:34:45.667: //-1//Dest:/DestSetup:
*Jan  6 19:34:45.671: :DestSetup iw guid is 527B07DA-7E2211DA-808DD216-195F6285
*Jan  6 19:34:45.671: :DestSetup iw guid is 527B07DA-7E2211DA-808DD216-195F6285
*Jan  6 19:34:45.671: //-1//Dest:/DestSetup:
*Jan  6 19:34:45.671: :DestSetup setup inc guid is 0-0-0-0
*Jan  6 19:34:45.671: //-1//Dest:/DestSetup:
*Jan  6 19:34:45.671: :DestSetup setup guid is 527B07DA-7E2211DA-808DD216-195F6285
*Jan  6 19:35:04.975: %VOIPAAA-5-VOIP_CALL_HISTORY: CallLegType 1, ConnectionId 45AB9E05
7E2211DA 8088D216 195F6285, SetupTime *19:34:22.535 UTC Fri Jan 6 2006, PeerAddress
1011011007, PeerSubAddress , DisconnectCause 10  , DisconnectText normal call clearing (16),
 ConnectTime *19:34:25.135 UTC Fri Jan 6 2006, DisconnectTime *19:35:04.975 UTC Fri Jan 6
2006, CallOrigin 1, ChargedUnits 0, InfoType 2, TransmitPackets 0, TransmitBytes 0,
ReceivePackets 0, ReceiveBytes 0
*Jan  6 19:35:04.991: %VOIPAAA-5-VOIP_CALL_HISTORY: CallLegType 1, ConnectionId 527B07DA
7E2211DA 808DD216 195F6285, SetupTime *19:34:43.861 UTC Fri Jan 6 2006, PeerAddress
1011011007, PeerSubAddress , DisconnectCause 10  , DisconnectText normal call clearing (16),
 ConnectTime *19:34:46.451 UTC Fri Jan 6 2006, DisconnectTime *19:35:04.991 UTC Fri Jan 6
2006, CallOrigin 2, ChargedUnits 0, InfoType 2, TransmitPackets 0, TransmitBytes 0,
ReceivePackets 0, ReceiveBytes 0
*Jan  6 19:36:05.627: %VOIPAAA-5-VOIP_CALL_HISTORY: CallLegType 1, ConnectionId 45AB9E05
7E2211DA 8088D216 195F6285, SetupTime *19:34:22.377 UTC Fri Jan 6 2006, PeerAddress
1011011006, PeerSubAddress , DisconnectCause 10  , DisconnectText normal call clearing (16),
 ConnectTime *19:34:25.137 UTC Fri Jan 6 2006, DisconnectTime *19:36:05.627 UTC Fri Jan 6
2006, CallOrigin 2, ChargedUnits 0, InfoType 2, TransmitPackets 0, TransmitBytes 0,
ReceivePackets 0, ReceiveBytes 0
*Jan  6 19:36:05.631: %VOIPAAA-5-VOIP_CALL_HISTORY: CallLegType 1, ConnectionId 527B07DA
7E2211DA 808DD216 195F6285, SetupTime *19:34:45.671 UTC Fri Jan 6 2006, PeerAddress
1011011008, PeerSubAddress , DisconnectCause 10  , DisconnectText normal call clearing (16),
 ConnectTime *19:34:46.451 UTC Fri Jan 6 2006, DisconnectTime *19:36:05.631 UTC Fri Jan 6
2006, CallOrigin 1, ChargedUnits 0, InfoType 2, TransmitPackets 0, TransmitBytes 0,
ReceivePackets 0, ReceiveBytes 0
*Jan  6 19:36:12.287: %IPPHONE-6-UNREGISTER_NORMAL: ephone-6:SEP111100011006 IP:10.3.32.56
 Socket:1 DeviceType:Phone has unregistered normally.
*Jan  6 19:36:12.287: %IPPHONE-6-UNREGISTER_NORMAL: ephone-7:SEP111100011007 IP:10.3.32.56
 Socket:2 DeviceType:Phone has unregistered normally.
*Jan  6 19:36:12.295: %IPPHONE-6-UNREGISTER_NORMAL: ephone-8:SEP111100011008 IP:10.3.32.56
 Socket:3 DeviceType:Phone has unregistered normally.
*Jan  6 19:36:13.227: %SYS-5-CONFIG_I: Configured from console by console
```

The following is sample output from the **debug voip application oodrefer** command:

```
Router# debug voip application oodrefer
Aug 22 18:16:21.625: //-1//AFW_:/C_ServiceThirdParty_Event_Handle:
Aug 22 18:16:21.625: //-1//AFW_:/AFW_ThirdPartyCC_New:
Aug 22 18:16:21.625: //-1//AFW_:EE461DC520000:/C_PackageThirdPartyCC_NewReq: ThirdPartyCC
module listened by TclModule_45F39E28_0_91076048
Aug 22 18:16:21.625: //-1//AFW_:EE461DC520000:/OCOpen_SetupRequest: Refer Dest1: 1011, Refer
 Dest2: 1001; ReferBy User: root
Aug 22 18:16:21.693: //-1//AFW_:EE461DC520000:/OCHandle_SignalEvent_1:
Aug 22 18:16:21.693: //-1//AFW_:/Third_Party_CC_Send_Notify: Third_Party_CC_Send_Notify:
sending notify respStatus=2, final=FALSE, failureCause=16
Aug 22 18:16:21.693: //-1//AFW_:/Third_Party_CC_Send_Notify: AppNotify successful!
Aug 22 18:16:26.225: //-1//AFW_:EE461DC520000:/OCHandle_SignalEvent_1:
Aug 22 18:16:26.229: //-1//AFW_:EE461DC520000:/OCHandle_SignalEvent_1:
Aug 22 18:16:26.249: //-1//AFW_:EE461DC520000:/OCHandle_SignalEvent_2:
Aug 22 18:16:29.341: //-1//AFW_:EE461DC520000:/OCHandle_SignalEvent_2:
Aug 22 18:16:29.341: //-1//AFW_:/Third_Party_CC_Send_Notify: Third_Party_CC_Send_Notify:
sending notify respStatus=4, final=TRUE, failureCause=16
```

```
Aug 22 18:16:29.341: //-1//AFW_:/Third_Party_CC_Send_Notify: AppNotify successful!
Aug 22 18:16:29.349: //-1//AFW_:EE461DC520000:/OCHandle_Handoff: BAG contains:
Aug 22 18:16:29.349: LEG[895    ][LEG_INCCONNECTED(5)][Cause(0)]
Aug 22 18:16:29.349: CON[7      ][CONNECTION_CONFED(2)] {LEG[895
][LEG_INCCONNECTED(5)][Cause(0)],LEG[896    ][LEG_OUTCONNECTED(10)][Cause(0)]}
Aug 22 18:16:29.349: LEG[896    ][LEG_OUTCONNECTED(10)][Cause(0)]
Aug 22 18:16:29.365: //-1//AFW_:EE461DC520000:/OCAnyState_IgnoreEvent: Event Ignored
Aug 22 18:16:29.365: //-1//AFW_:/C_ServiceThirdParty_Event_Handle:
Aug 22 18:16:29.365: //-1//AFW_:EE461DC520000:/C_ServiceThirdParty_Event_Handle: Received
event APP_EV_NOTIFY_DONE[174] in Main Loop
Aug 22 18:16:29.365: //-1//AFW_:EE461DC520000:/OCAnyState_IgnoreEvent: Event Ignored
Aug 22 18:16:29.365: //-1//AFW_:/C_ServiceThirdParty_Event_Handle:
Aug 22 18:16:29.365: //-1//AFW_:EE461DC520000:/C_ServiceThirdParty_Event_Handle: Received
event APP_EV_NOTIFY_DONE[174] in Main Loop
Aug 22 18:16:29.369: //-1//AFW_:EE461DC520000:/OCHandle_SubscribeCleanup:
Aug 22 18:16:29.369: //-1//AFW_:EE461DC520000:/Third_Party_CC_Cleaner:
Aug 22 18:16:29.453: //-1//AFW_:EE461DC520000:/OCClosing_AnyEvent:
Aug 22 18:16:29.453: //-1//AFW_:EE461DC520000:/Third_Party_CC_Cleaner:
Aug 22 18:16:29.453: //-1//AFW_:EE461DC520000:/OCClosing_AnyEvent:
Aug 22 18:16:29.453: //-1//AFW_:EE461DC520000:/Third_Party_CC_Cleaner:
```

Related Commands

Command	Description
debug voip ivr all	Displays all IVR and application framework messages.
supplementary-service h450.2 (dial-peer)	Enables H.450.2 capabilities for call transfers for an individual dial peer.
supplementary-service h450.2 (voice-service)	Globally enables H.450.2 capabilities for call transfers.
supplementary-service h450.3 (dial-peer)	Enables H.450.3 capabilities for call forwarding for an individual dial peer.
supplementary-service h450.3 (voice-service)	Globally enables H.450.3 capabilities for call forwarding.
supplementary-service h450.12 (dial-peer)	Enables H.450.12 capabilities for an individual dial peer.
supplementary-service h450.12 (voice-service)	Globally enables H.450.12 capabilities.

debug voip application stcapp all

To display debugging information for the components of the SCCP Telephony Control Application (STCAPP), use the **debug voip application stcapp all**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voip application stcapp all

no debug voip application stcapp all

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.3(14)T	This command was introduced.
12.4(4)T	Command output was enhanced to display codec capabilities for modem transport.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.
12.4(6)XE	Command output was enhanced to display fax relay, RFC 2833 DTMF digit relay, dial tone after remote onhook, call control feature mode and visual message waiting indicator (VMWI) information for skinny client control protocol (SCCP) analog ports.
12.4(11)T	This command was integrated into Cisco IOS Release 12.4(11)T.

Usage Guidelines

The **debug voip application stcapp all**command provides debugging output for all the STCAPP debug commands compiled into one display.

Examples

The following is sample output from the **debug voip application stcapp all** command for a Cisco VG 224 voice gateway in Cisco IOS Release 12.4(6)XE showing call control feature mode messages for the drop last active call feature. Port 2/0 calls port 2/1, performs a hook flash to a get dial tone while port 2/1 is on hold, and calls port 2/3. Ports 2/0 and 2/3 are active, while port 2/1 is on hold.

```
Router# debug voip application stcapp al
l
Port 2/0 performs a hook flash to activate the drop last call feature.
Mar  3 20:41:07.022: 2/0   : stcapp_screen_api_event
Mar  3 20:41:07.022: 2/0   :     event:STCAPP_CC_EV_CALL_FEATURE_HOOKFLASH received.
Mar  3 20:41:07.022: 2/0   : stcapp_screen_call_feature_hookflash
Mar  3 20:41:07.022: 2/0   :     lcb->num_ccbs=2, lcb->mode=CALL_TRANSFER(1),
```

```
lcb->state=ACTIVE (8)
Mar  3 20:41:07.022: 2/0   :     lcb->fm_mode=CALL_FM_NONE(0),
Mar  3 20:41:07.022: 2/0   :    ACTIVE call_state=5, lcb->mode=CALL_TRANSFER(1), lcb->state=8
Mar  3 20:41:07.022: 2/0   :     STANDBY call_state=8, lcb->mode=CALL_TRANSFER(1),
lcb->state=8
Mar  3 20:41:07.022: 2/0   :     enter FM: set event id to STCAPP_EV_FEATURE_MODE(143)
Mar  3 20:41:07.022: 2/0   : ==> Received event:STCAPP_EV_FEATURE_MODE for CallId: 63
Mar  3 20:41:07.022: 2/0   :     Call State:ACTIVE
Mar  3 20:41:07.022: 2/0   : stcapp_feature_mode_eh
```

The following line shows a message to generate feature tone.

```
Mar  3 20:41:07.022: 2/0   :     Sending ccGenerateTone(2048(0x800)):FEATURE tone
Mar  3 20:41:07.022: 2/0   :     set lcb->fm_mode = CALL_FM_TRANSF_FT_ON (4)
Mar  3 20:41:07.022: 2/0   :     Sending ccCallReportDigits
Mar  3 20:41:07.022: 2/0   :     New State = FM_DIGIT_COLLECT
```

The following lines show port 2/0 dialing the feature access code (FAC) #1 to drop the last active call

```
Mar  3 20:41:11.234: htsp_digit_ready(2/0): digit = #
Mar  3 20:41:11.234: STCAPP:Receive CC event:: call_id=63, ccb=0x64A5059C
Mar  3 20:41:11.234: 2/0   : ==> Received event:STCAPP_CC_EV_CALL_DIGIT_END for CallId: 63
Mar  3 20:41:11.234: 2/0   :     Call State:FM_DIGIT_COLLECT
Mar  3 20:41:11.234: 2/0   : stcapp_fm_dc_digit_end_eh
Mar  3 20:41:11.234: 2/0   :     Digit received is (#)
Mar  3 20:41:11.234: 2/0   :     lcb->fm_mode = CALL_FM_TRANSF_FT_ON(4)
Mar  3 20:41:11.234: 2/0   :     Sending ccGenerateTone(0x0)
Mar  3 20:41:11.234: 2/0   :      set lcb->fm_mode to CALL_FM_TRANSF_FT_OFF (5)
Mar  3 20:41:11.234: 2/0   :     So far the fm feature code =#, fm_string_idx=1
Mar  3 20:41:11.238: 2/0   :     No state change
Mar  3 20:41:12.346: htsp_digit_ready(2/0): digit = 1
Mar  3 20:41:12.346: STCAPP:Receive CC event:: call_id=63, ccb=0x64A5059C
Mar  3 20:41:12.346: 2/0   : ==> Received event:STCAPP_CC_EV_CALL_DIGIT_END for CallId: 63
Mar  3 20:41:12.346: 2/0   :     Call State:FM_DIGIT_COLLECT
Mar  3 20:41:12.346: 2/0   : stcapp_fm_dc_digit_end_eh
Mar  3 20:41:12.346: 2/0   :     Digit received is (1)
Mar  3 20:41:12.346: 2/0   :     lcb->fm_mode = CALL_FM_TRANSF_FT_OFF(5)
Mar  3 20:41:12.346: 2/0   :     So far the fm feature code =#1, fm_string_idx=2
```

The following lines show the SCCP gateway processing FAC #1 messages.

```
Mar  3 20:41:12.346: 2/0   : stcapp_handle_fm_feature_id
Mar  3 20:41:12.346: 2/0   :     lcb->fm_mode = CALL_FM_TRANSF_FT_OFF(5),
Mar  3 20:41:12.346: 2/0   :     STANDBY call_state=8
Mar  3 20:41:12.346: 2/0   : stcapp_send_softkey_event
Mar  3 20:41:12.346: 2/0   :     Sending dcDeviceSoftKeyEvent(EndCall: event=9) for device
 id: 1, call_ref: 16810780
Mar  3 20:41:12.346: 2/0   : stcapp_send_softkey_event
Mar  3 20:41:12.346: 2/0   :     Sending dcDeviceSoftKeyEvent(Resume: event=10) for device
 id: 1, call_ref: 16810777
Mar  3 20:41:12.346: 2/0   :      set lcb->fm_mode to CALL_FM_NONE (0)
Mar  3 20:41:12.346: 2/0   :     No state change
Mar  3 20:41:12.366: 2/0   : ==> Received event:STCAPP_DC_EV_MEDIA_CLOSE_RCV_CHNL
Mar  3 20:41:12.366: 2/0   :     Call State:FM_DIGIT_COLLECT
Mar  3 20:41:12.366: 2/0   : stcapp_close_rcv_chnl_eh
Mar  3 20:41:12.366: 2/0   : stcapp_disconnect_call_leg
Mar  3 20:41:12.366: 2/0   :     Sending ccCallDisconnect for VoIP_LEG with call id:70
Mar  3 20:41:12.366: 2/0   :     No state change
Mar  3 20:41:12.370: 2/0   : ==> Received event:STCAPP_DC_EV_MEDIA_CLOSE_XMT_CHNL
Mar  3 20:41:12.370: 2/0   :     Call State:FM_DIGIT_COLLECT
Mar  3 20:41:12.370: 2/0   : stcapp_active_close_xmt_chnl_eh
Mar  3 20:41:12.370: 2/0   :     New State = ONHOOK_PEND
```

The following is sample output from the **debug voip application stcapp all** command for a Cisco VG224 voice gateway in Cisco IOS Release 12.4(6)XE showing call control feature mode messages for the call transfer feature:

```
Router# debug voip application stcapp all
Mar  3 21:00:56.014: 2/0   : stcapp_handle_fm_feature_id
```

```
Mar  3 21:00:56.014: 2/0   :     lcb->fm_mode = CALL_FM_TRANSF_FT_OFF(5),
Mar  3 21:00:56.014: 2/0   :      set lcb->mode to CALL_TRANSFER (1)
Mar  3 21:00:56.014: 2/0   : stcapp_send_softkey_event
Mar  3 21:00:56.014: 2/0   :     Sending dcDeviceSoftKeyEvent(Transfer: event=4) for device
 id: 1, call_ref: 16810789
Mar  3 21:00:56.014: 2/0   :      set lcb->fm_mode to CALL_FM_NONE (0)
Mar  3 21:00:56.014: 2/0   :     No state change
Mar  3 21:00:56.034: 2/0   : ==> Received event:STCAPP_DC_EV_MEDIA_CLOSE_RCV_CHNL
Mar  3 21:00:56.034: 2/0   :     Call State:FM_DIGIT_COLLECT
Mar  3 21:00:56.038: 2/0   : stcapp_close_rcv_chnl_eh
Mar  3 21:00:56.038: 2/0   : stcapp_disconnect_call_leg
Mar  3 21:00:56.038: 2/0   :     No state change
Mar  3 21:00:56.042: 2/0   : ==> Received event:STCAPP_DC_EV_MEDIA_CLOSE_XMT_CHNL
Mar  3 21:00:56.042: 2/0   :     Call State:FM_DIGIT_COLLECT
Mar  3 21:00:56.042: 2/0   : stcapp_active_close_xmt_chnl_eh
Mar  3 21:00:56.042: 2/0   :     New State = ONHOOK_PEND
```

The following is sample output in Cisco IOS 12.4(6)XE from the **debug voip application stcapp all** command showing call control feature mode messages for the call conference feature:

```
Router# debug voip application stcapp all
Mar  3 21:18:54.258: 2/0   : stcapp_handle_fm_feature_id
Mar  3 21:18:54.258: 2/0   :     lcb->fm_mode = CALL_FM_TRANSF_FT_OFF(5),
Mar  3 21:18:54.258: 2/0   : stcapp_send_softkey_event
Mar  3 21:18:54.258: 2/0   :     Sending dcDeviceSoftKeyEvent(Conference: event=13) for
device id: 1, call_ref: 16810798
Mar  3 21:18:54.258: 2/0   :      set lcb->fm_mode to CALL_FM_NONE (0)
Mar  3 21:18:54.258: 2/0   :     No state change
Mar  3 21:18:54.298: 2/0   : ==> Received event:STCAPP_DC_EV_MEDIA_CLOSE_RCV_CHNL
Mar  3 21:18:54.298: 2/0   :     Call State:FM_DIGIT_COLLECT
Mar  3 21:18:54.298: 2/0   : stcapp_close_rcv_chnl_eh
Mar  3 21:18:54.298: 2/0   : stcapp_disconnect_call_leg
Mar  3 21:18:54.298: 2/0   :     No state change
Mar  3 21:18:54.302: 2/0   : ==> Received event:STCAPP_DC_EV_MEDIA_CLOSE_XMT_CHNL
Mar  3 21:18:54.302: 2/0   :     Call State:FM_DIGIT_COLLECT
Mar  3 21:18:54.302: 2/0   : stcapp_active_close_xmt_chnl_eh
Mar  3 21:18:54.302: 2/0   :     New State = ONHOOK_PEND
```

The following is sample output n Cisco IOS 12.4(6)XE from the **debug voip application stcapp all** command showing call control feature mode messages for the drop last conferee feature:

```
Router# debug voip application stcapp all
Mar  3 21:27:05.170: 2/0   : stcapp_handle_fm_feature_id
Mar  3 21:27:05.170: 2/0   :     lcb->fm_mode = CALL_FM_CONF_FT_OFF(7),
Mar  3 21:27:05.170: 2/0   : stcapp_send_softkey_event
Mar  3 21:27:05.170: 2/0   :     Sending dcDeviceSoftKeyEvent(DropLastConferee: event=19)
for device id: 1, call_ref: 16810795
Mar  3 21:27:05.170: 2/0   :      set lcb->fm_mode to CALL_FM_NONE (0)
Mar  3 21:27:05.170: 2/0   :     No state change
Mar  3 21:27:05.194: 2/0   : ==> Received event:STCAPP_DC_EV_DEVICE_CALL_INFO
Mar  3 21:27:05.194: 2/0   :     Call State:FM_DIGIT_COLLECT
Mar  3 21:27:05.194: 2/0   : stcapp_conn_call_info_eh
Mar  3 21:27:05.194: 2/0   :     No state change
Mar  3 21:27:05.194: 2/0   : ==> Received event:STCAPP_DC_EV_MEDIA_CLOSE_RCV_CHNL
Mar  3 21:27:05.194: 2/0   :     Call State:FM_DIGIT_COLLECT
Mar  3 21:27:05.194: 2/0   : stcapp_close_rcv_chnl_eh
Mar  3 21:27:05.198: 2/0   : stcapp_disconnect_call_leg
Mar  3 21:27:05.198: 2/0   :     No state change
Mar  3 21:27:05.198: 2/0   : ==> Received event:STCAPP_DC_EV_MEDIA_CLOSE_XMT_CHNL
Mar  3 21:27:05.198: 2/0   :     Call State:FM_DIGIT_COLLECT
Mar  3 21:27:05.202: 2/0   : stcapp_active_close_xmt_chnl_eh
Mar  3 21:27:05.202: 2/0   :     New State = ONHOOK_PEND
```

The following is sample output in Cisco IOS 12.4(6)XE from the **debug voip application stcapp all** command showing call control feature mode messages for the toggle feature:

```
Router# debug voip application stcapp all
Mar  3 21:37:11.650: 2/0   : stcapp_handle_fm_feature_id
```

```
Mar  3 21:37:11.650: 2/0   :     lcb->fm_mode = CALL_FM_TRANSF_FT_OFF(5),
Mar  3 21:37:11.650: 2/0   :     STANDBY call_state=8
Mar  3 21:37:11.650: 2/0   : stcapp_send_softkey_event
Mar  3 21:37:11.650: 2/0   :     Sending dcDeviceSoftKeyEvent(Hold: event=3) for device id:
 1, call_ref: 16810811
Mar  3 21:37:11.650: 2/0   : stcapp_send_softkey_event
Mar  3 21:37:11.650: 2/0   :     Sending dcDeviceSoftKeyEvent(Resume: event=10) for device
 id: 1, call_ref: 16810808
Mar  3 21:37:11.650: 2/0   :     set new lcb->fm_mode=CALL_FM_NONE (0)
Mar  3 21:37:11.650: 2/0   :     in toggle: (1)
Mar  3 21:37:11.650: 2/0   :     No state change
Mar  3 21:37:11.654: 2/0   : ==> Received event:STCAPP_DC_EV_MEDIA_CLOSE_RCV_CHNL
Mar  3 21:37:11.658: 2/0   :     Call State:FM_DIGIT_COLLECT
Mar  3 21:37:11.658: 2/0   : stcapp_close_rcv_chnl_eh
Mar  3 21:37:11.658: 2/0   : stcapp_disconnect_call_leg
Mar  3 21:37:11.658: 2/0   :     No state change
Mar  3 21:37:11.674: 2/0   : ==> Received event:STCAPP_DC_EV_MEDIA_CLOSE_XMT_CHNL
Mar  3 21:37:11.674: 2/0   :     Call State:FM_DIGIT_COLLECT
Mar  3 21:37:11.674: 2/0   : stcapp_active_close_xmt_chnl_eh
Mar  3 21:37:11.674: 2/0   :     New State = ONHOOK_PEND
```

The following is sample output from the **debug voip application stcapp all** command for a Cisco VG224 voice gateway in Cisco IOS Release 12.4(6)XE showing T.38 fax relay messages:

```
Router# debug voip application stcapp all
01:10:40: //68/xxxxxxxxxxxx/CCAPI/cc_api_set_fax_mode:
   Destination Interface=0x464EDDFC, Destination Call Id=68, Source Call Id=67
01:10:40: [1185752428]CNFSM: new_container:fax_t38_container
01:10:40: [1185752428]CNFSM: next_state:S_DSMP_GW_FAX_T38_CONNECTED
01:10:40: STCAPP:Receive CC event:: call_id=68, ccb=0x46B264C4
01:10:40: 1/0/0: ==> Received event:STCAPP_CC_EV_CALL_FEATURE_T38_CODEC for CallId: 68
01:10:40: 1/0/0:    Call State:ACTIVE
01:10:40: 1/0/0: stcapp_t38_remote_codec_dnld_done_eh
01:10:40: //68/xxxxxxxxxxxx/CCAPI/ccCallFeature:
   Feature Type=39, Call Id=68
01:10:40: //67/xxxxxxxxxxxx/CCAPI/cc_api_remote_codec_dnld_done:
   Destination Interface=0x4631148C, Destination Call Id=67, Source Call Id=68, Xmit
Function=0x4230E4D4
01:10:40: //-1/xxxxxxxxxxxx/DSM:():-1/dsp_stream_mgr_remote_dnld_done:
```

The following is sample output from the **debug voip application stcapp all** command for a voice gateway in Cisco IOS Release 12.4(4)T showing device modem transport capability:

```
Router# debug voip application stcapp all
01:10:40: //68/xxxxxxxxxxxx/CCAPI/cc_api_set_fax_mode:
   Destination Interface=0x464EDDFC, Destination Call Id=68, Source Call Id=67
01:10:40: [1185752428]CNFSM: new_container:fax_t38_container
01:10:40: [1185752428]CNFSM: next_state:S_DSMP_GW_FAX_T38_CONNECTED
01:10:40: STCAPP:Receive CC event:: call_id=68, ccb=0x46B264C4
01:10:40: 1/0/0: ==> Received event:STCAPP_CC_EV_CALL_FEATURE_T38_CODEC for CallId: 68
01:10:40: 1/0/0:    Call State:ACTIVE
01:10:40: 1/0/0: stcapp_t38_remote_codec_dnld_done_eh
01:10:40: //68/xxxxxxxxxxxx/CCAPI/ccCallFeature:
   Feature Type=39, Call Id=68
01:10:40: //67/xxxxxxxxxxxx/CCAPI/cc_api_remote_codec_dnld_done:
   Destination Interface=0x4631148C, Destination Call Id=67, Source Call Id=68, Xmit
Function=0x4230E4D4
01:10:40: //-1/xxxxxxxxxxxx/DSM:():-1/dsp_stream_mgr_remote_dnld_done:
```

The following is sample output from the **debug voip application stcapp all**command showing modem transport device capability:

```
Router# debug voip application stcapp all
*Jan 11 12:24:18.443: stcapp_start
*Jan 11 12:24:18.443:     stcapp process started
*Jan 11 12:24:18.443: stcapp_init_symphony
*Jan 11 12:24:18.443:     CCAPI successfully initialized
*Jan 11 12:24:18.443: stcapp_init_rtp
```

```
*Jan 11 12:24:18.443: stcapp_vp_shut
*Jan 11 12:24:18.443: stcapp_port_up_down
*Jan 11 12:24:18.443:     RTP successfully brought in service
*Jan 11 12:24:18.443: stcapp_create_dcbs_from_dialpeers
*Jan 11 12:24:18.447: 1/1/0: stcapp_create_device
*Jan 11 12:24:18.447: 1/1/0:     Endpoint base name generated->AN0D65D8DD40280
*Jan 11 12:24:18.447: 1/1/0:     New dialpeer id: 999110
*Jan 11 12:24:18.447: 1/1/0:     Analog device is ready to be registered
```

The following lines show the codec subtype, which indicates the modem transport method, 0=None, 1=V.150.1 (modem relay), 2=VBD (modem pass-through):

```
*Jan 11 12:24:18.447: 1/1/0:     reg caps including codec=5 (g711ulaw) subtype=2
*Jan 11 12:24:18.447: 1/1/0:     reg caps including codec=1 (g729ar8) subtype=2
*Jan 11 12:24:18.447: 1/1/0:     reg caps including codec=5 (g711ulaw) subtype=1
*Jan 11 12:24:18.447: 1/1/0:     reg caps including codec=1 (g729ar8) subtype=1
*Jan 11 12:24:18.447: 1/1/0:     reg caps including codec=5 (g711ulaw) subtype=0
*Jan 11 12:24:18.447: 1/1/0:     reg caps including codec=6 (g711alaw) subtype=0
*Jan 11 12:24:18.447: 1/1/0:     reg caps including codec=1 (g729ar8) subtype=0
*Jan 11 12:24:18.447: 1/1/0:     reg caps including codec=2 (g726r16) subtype=0
*Jan 11 12:24:18.447: 1/1/0:     reg caps including codec=3 (g726r24) subtype=0
*Jan 11 12:24:18.447: 1/1/0:     reg caps including codec=4 (g726r32) subtype=0
*Jan 11 12:24:18.447: 1/1/0:     reg caps including codec=7 (g728) subtype=0
*Jan 11 12:24:18.447: 1/1/0:     reg caps including codec=8 (g723r63) subtype=0
*Jan 11 12:24:18.447: 1/1/0:     reg caps including codec=9 (g723r53) subtype=0
*Jan 11 12:24:18.447: 1/1/0:     reg caps including codec=12 (g729br8) subtype=0
*Jan 11 12:24:18.447: 1/1/0:     reg caps including codec=14 (g723ar63) subtype=0
*Jan 11 12:24:18.447: 1/1/0:     reg caps including codec=15 (g723ar53) subtype=0
*Jan 11 12:24:18.447: 1/1/0:    Device: AN0D65D8DD40280 Id: 7 successfully registered with
 CM
*Jan 11 12:24:18.455: ==> Received event:STCAPP_DC_EV_DEVICE_REGISTER_DONE
*Jan 11 12:24:18.455: 1/1/0:     Device State:OOS
*Jan 11 12:24:18.455: 1/1/0: stcapp_dev_default_eh
*Jan 11 12:24:18.455: 1/1/0:     New State = INIT
*Jan 11 12:24:18.455: ==> Received event:STCAPP_DC_EV_DEVICE_CAP_REQ
*Jan 11 12:24:18.455: 1/1/0:     Device State:INIT
*Jan 11 12:24:18.455: 1/1/0: stcapp_cap_req_eh
*Jan 11 12:24:18.455: 1/1/0:     Sending dcDeviceHeadsetStatus for devID:7
*Jan 11 12:24:18.455: 1/1/0:     Sending dcDeviceButtonTemplateReq for devID:7
*Jan 11 12:24:18.455: 1/1/0:     No state change
*Jan 11 12:24:18.647: ==> Received event:STCAPP_DC_EV_DEVICE_BUTTON_TEMP_RES
*Jan 11 12:24:18.647: 1/1/0:     Device State:INIT
*Jan 11 12:24:18.647: 1/1/0: stcapp_button_templ_res_eh
*Jan 11 12:24:18.647: 1/1/0:     Sending dcDeviceLineStatReq for devID:7
*Jan 11 12:24:18.647: 1/1/0:     No state change
*Jan 11 12:24:18.647: ==> Received event:STCAPP_DC_EV_DEVICE_FORWARD_STAT_RES
*Jan 11 12:24:18.647: 1/1/0:     Device State:INIT
*Jan 11 12:24:18.647: 1/1/0: stcapp_forward_stat_res_eh
*Jan 11 12:24:18.647: 1/1/0:     lineNumber: 1
*Jan 11 12:24:18.647: 1/1/0:     forwardAllActive: 0
*Jan 11 12:24:18.647: 1/1/0:     forwardBusyActive: 0
*Jan 11 12:24:18.647: 1/1/0:     forwardNoAnswerActive: 0
*Jan 11 12:24:18.651: 1/1/0:     ForwardAllDirNumber:
*Jan 11 12:24:18.651: 1/1/0:     No state change
*Jan 11 12:24:18.651: ==> Received event:STCAPP_DC_EV_DEVICE_LINE_STAT_RES
*Jan 11 12:24:18.651: 1/1/0:     Device State:INIT
*Jan 11 12:24:18.455: 1/1/0: stcapp_cap_req_eh
*Jan 11 12:24:18.455: 1/1/0:     Sending dcDeviceHeadsetStatus for devID:7
*Jan 11 12:24:18.455: 1/1/0:     Sending dcDeviceButtonTemplateReq for devID:7
*Jan 11 12:24:18.455: 1/1/0:     No state change
*Jan 11 12:24:18.647: ==> Received event:STCAPP_DC_EV_DEVICE_BUTTON_TEMP_RES
*Jan 11 12:24:18.647: 1/1/0:     Device State:INIT
*Jan 11 12:24:18.647: 1/1/0: stcapp_button_templ_res_eh
*Jan 11 12:24:18.647: 1/1/0:     Sending dcDeviceLineStatReq for devID:7
*Jan 11 12:24:18.647: 1/1/0:     No state change
*Jan 11 12:24:18.647: ==> Received event:STCAPP_DC_EV_DEVICE_FORWARD_STAT_RES
*Jan 11 12:24:18.647: 1/1/0:     Device State:INIT
*Jan 11 12:24:18.647: 1/1/0: stcapp_forward_stat_res_eh
*Jan 11 12:24:18.647: 1/1/0:     lineNumber: 1
*Jan 11 12:24:18.647: 1/1/0:     forwardAllActive: 0
*Jan 11 12:24:18.647: 1/1/0:     forwardBusyActive: 0
*Jan 11 12:24:18.647: 1/1/0:     forwardNoAnswerActive: 0
```

```
*Jan 11 12:24:18.651: 1/1/0:     ForwardAllDirNumber:
*Jan 11 12:24:18.651: 1/1/0:     No state change
*Jan 11 12:24:18.651: ==> Received event:STCAPP_DC_EV_DEVICE_LINE_STAT_RES
*Jan 11 12:24:18.651: 1/1/0:     Device State:INIT
*Jan 11 12:24:18.651: 1/1/0: stcapp_line_stat_eh
*Jan 11 12:24:18.651: 1/1/0:     lineNumber: 1
*Jan 11 12:24:18.651: 1/1/0:     lineDirNumber: 5902
*Jan 11 12:24:18.651: 1/1/0:     display name: 5902
*Jan 11 12:24:18.651: 1/1/0:     Sending dcDeviceRegAvailableLines for devID:7
*Jan 11 12:24:18.651: 1/1/0:     Sending dcDeviceDateTimeReq for devID:7
*Jan 11 12:24:18.651: 1/1/0:     No state change
*Jan 11 12:24:18.823: ==> Received event:STCAPP_DC_EV_DEVICE_DEFINE_DATE_TIME_RES
*Jan 11 12:24:18.827: 1/1/0:     Device State:INIT
*Jan 11 12:24:18.827: 1/1/0: stcapp_define_date_time_eh
*Jan 11 12:24:18.827: 1/1/0:     New State = IS
*Jan 11 12:24:18.827: ==> Received event:STCAPP_DC_EV_DEVICE_DISPLAY_PROMPT_STATUS
*Jan 11 12:24:18.827: 1/1/0:     Device State:IS
*Jan 11 12:24:18.827: 1/1/0: stcapp_display_prompt_status_eh
*Jan 11 12:24:18.827: 1/1/0:     lineNumber: 0
*Jan 11 12:24:18.827: 1/1/0:     call reference: 0
*Jan 11 12:24:18.827: 1/1/0:     promptStatus: Your current options
*Jan 11 12:24:18.827: 1/1/0:     device control type: 3
*Jan 11 12:24:18.827: 1/1/0:     No state change
```

The following is sample output from the **debug voip application stcapp all** command during call setup:

```
Router# debug voip application stcapp all
```

The following lines show the voice gateway beginning call setup:

```
*Jan  9 06:48:06.947: ==> Received event:STCAPP_CC_EV_CALL_SETUP_IND
(evId:CC_EV_CALL_SETUP_IND) for CallId: 5
*Jan  9 06:48:06.947: 1/0/0:     Call State:IDLE
*Jan  9 06:48:06.947: 1/0/0: stcapp_setup_ind_eh
*Jan  9 06:48:06.947: 1/0/0:     Acquired CCB 0x66F12558 for device id:4
*Jan  9 06:48:06.947: 1/0/0:     Voice Setup: callID:5, vdb_ptr:66CA57B4
```

The voice gateway notifies the Cisco Unified Communications Manager of the endpoint device (phone) in the offhook condition.

```
*Jan  9 06:48:06.947: 1/0/0:     Sending StationOffHook to CallManager
*Jan  9 06:48:06.947: 1/0/0:     Sending ccCallSetupAck to Symphony for voice call id:5
*Jan  9 06:48:06.947: 1/0/0:     New State = OFFHOOK
*Jan  9 06:48:06.955: 1/0/0:     No line (line=0) found... most likely old Call Ref: event
 STCAPP_DC_EV_DEVICE_SET_RINGER
```

The following lines show that the Cisco Unified Communications Manager acknowledged the offhook condition:

```
*Jan  9 06:48:06.955: ==> Received event:STCAPP_DC_EV_DEVICE_CALL_STATE_OFFHOOK
(evID:DC_EV_DEVICE_CALL_STATE_OFFHOOK)
*Jan  9 06:48:06.955: 1/0/0:     Call State:OFFHOOK
*Jan  9 06:48:06.955: 1/0/0: stcapp_cs_offhook_eh
*Jan  9 06:48:06.955: 1/0/0:     No state change
```

The voice gateway receives the Cisco Unified Communications Manager notification to send dial tone to the phone.

```
*Jan  9 06:48:06.955: ==> Received event:STCAPP_DC_EV_DEVICE_START_TONE
(evID:DC_EV_DEVICE_START_TONE)
*Jan  9 06:48:06.955: 1/0/0:     Call State:OFFHOOK
*Jan  9 06:48:06.955: 1/0/0: stcapp_start_tone_eh
```

The voice gateway generates dial tone and prepares to collect dialed digits.

```
*Jan  9 06:48:06.955: 1/0/0:     Sending ccGenerateTone(8(0x8))
*Jan  9 06:48:06.955: 1/0/0:     Sending ccCallReportDigits
*Jan  9 06:48:06.955: 1/0/0:     No state change
```

Related Commands

Command	Description
debug voip application stcapp error	Displays STCAPP error log contents.
debug voip application stcapp events	Traces STCAPP call flow events.
debug voip application stcapp functions	Displays STCAPP entry and exit function calls for all voice ports.
debug voip application stcapp port	Displays debugging information for the components of the STCAPP for a specified port.

debug voip application stcapp buffer-history

To enable event logging for SCCP Telephony Control Application (STCAPP) analog voice ports, use the **debug voip application stcapp buffer-history** command in privileged EXEC mode. To disable event logging, use the **no** form of this command.

debug voip application stcapp buffer-history {**all**| **port** *port*}

no debug voip application stcapp buffer-history {**all**| **port** *port*}

Syntax Description

all	Enables logging for all analog voice ports.
port *port*	Enables logging for only the specified analog voice port. **Note** *Port* syntax is platform-dependent; type ? to determine.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.4(2)T	This command was introduced.

Usage Guidelines

This command enables logging of call flow events and device events, including registering and unregistering. You can use the event log to help troubleshoot performance problems and isolate faults related to analog endpoints. To display the records in the event log, use the **show sctapp buffer-history** command.

A maximum of 2000 records are saved for each analog port. The event log uses a circular buffer that stores the 2000 most recent records. To clear the buffer, you can disable logging with the **no debug voip application stcapp buffer-history** command. The event log uses approximately 64 KB of memory for each port, or approximately 1.5 MB of memory if logging is enabled for all 24 ports.

Note The **debug voip application stcapp all** command has no impact on event logging. Enabling or disabling STCAPP debug output is separate from the event logging feature.

Examples

The following example enables event logging for analog port 2/3. To display the events, you must use the **show sctapp buffer-history** command.

```
Router# debug voip application stcapp buffer-history port 2/3
stcapp buffer-history logging for port 2/3 is on
```

Related Commands

Command	Description
debug voip application stcapp all	Displays debug output for all the debug commands for the STCAPP compiled into one display.
debug voip application stcapp error	Displays STCAPP error log contents.
debug voip application stcapp events	Traces STCAPP call flow events.
debug voip application stcapp functions	Displays STCAPP entry and exit function calls for all voice ports.
debug voip application stcapp port	Displays STCAPP debug output for a specific port.
show stcapp buffer-history	Displays event logs for STCAPP analog voice ports.

debug voip application stcapp error

To troubleshoot the SCCP Telephony Control Application (STCAPP) error log contents, use the **debug voip application stcapp error**command in privileged EXEC mode. To disable STCAPP error debugging, use the **no** form of this command.

debug voip application stcapp error

no debug voip application stcapp error

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

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

Usage Guidelines

The **debug voip application stcapp error** command traces application error logs. STCAPP error logs are generated during normal call processing, when there are insufficient resources, or when there are problems in the underlying application code. This command shows error events or unexpected behavior in system software. Usually no events are generated.

Examples

The following example shows the error log contents when STCAPP debugging is enabled:

```
Router# debug voip application stcapp error
STCAPP error debugging is on
Router#
*Jan  9 06:54:07.583:    stcapp_process_queue_events:ERROR:STCAPP_DCB_ACCESS_ERR from state
 machine
```

Related Commands

Command	Description
debug voip application stcapp all	Displays debug output for all the debug commands for the STCAPP compiled into one display.
debug voip application stcapp events	Traces STCAPP call flow events.
debug voip application stcapp functions	Displays STCAPP entry and exit function calls for all voice ports.
debug voip application stcapp port	Displays debug information for the components of the STCAPP for a specified port.

debug voip application stcapp events

To trace SCCP Telephony Control Application (STCAPP) call flow events, use the **debug voip application stcapp events**command in privileged EXEC mode. To disable STCAPP event call traces, use the **no** form of this command.

debug voip application stcapp events

no debug voip application stcapp events

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

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

Usage Guidelines

Use this command to debug call flow events for all ports controlled by the STCAPP.

Examples

The following example displays call teardown and disconnect events:

```
Router# debug voip application stcapp events
```

The following lines show the application running on the voice gateway receiving notice to stop dial tone generation, following the onhook condition of the endpoint device (phone):

```
*Jan  9 06:48:55.011: ==> Received event:STCAPP_DC_EV_DEVICE_STOP_TONE
(evID:DC_EV_DEVICE_STOP_TONE)
*Jan  9 06:48:55.011: 1/0/1:     Call State:REM_ONHOOK_PEND
*Jan  9 06:48:55.011: 1/0/1: stcapp_stop_tone_eh
*Jan  9 06:48:55.011: 1/0/1:     Sending ccGenerateTone(NULL)
*Jan  9 06:48:55.015: 1/0/1:     No state change
```

The application takes no additional action to process the uninteresting event.

```
*Jan  9 06:48:55.015: ==> Received event:STCAPP_DC_EV_DEVICE_START_TONE
(evID:DC_EV_DEVICE_START_TONE)
*Jan  9 06:48:55.015: 1/0/1:     Call State:REM_ONHOOK_PEND
*Jan  9 06:48:55.015: 1/0/1:     Uninteresting event
```

The application receives the call disconnect notice and proceeds to tear down the telephony call leg.

```
*Jan  9 06:48:58.903: ==> Received event:STCAPP_CC_EV_CALL_DISCONNECTED
(evId:CC_EV_CALL_DISCONNECTED) for CallId: 6
*Jan  9 06:48:58.903: 1/0/1:     Call State:REM_ONHOOK_PEND
*Jan  9 06:48:58.903: 1/0/1: stcapp_loc_onhook_eh
*Jan  9 06:48:58.903: 1/0/1:     Sending StationOnHook to CallManager
*Jan  9 06:48:58.903: 1/0/1: stcapp_call_cleanup
*Jan  9 06:48:58.903: 1/0/1: stcapp_set_ring_mode
*Jan  9 06:48:58.903: 1/0/1:     SCCP ring mode:1
*Jan  9 06:48:58.903: 1/0/1:     Invoking Feature:33. Mode:0 for callid:6
```

```
*Jan  9 06:48:58.903: 1/0/1: stcapp_disconnect_call_leg
*Jan  9 06:48:58.903: 1/0/1:     Sending ccCallDisconnect for call id:6
*Jan  9 06:48:58.903: 1/0/1:     CCB 0x65CF3EC4 unlinked
*Jan  9 06:48:58.903: 1/0/1:     New State = IDLE
```

Related Commands

Command	Description
debug voip application stcapp all	Displays debug output for all the debug commands for the STCAPP compiled into one display.
debug voip application stcapp error	Displays STCAPP error log contents.
debug voip application stcapp functions	Displays STCAPP entry and exit function calls for all voice ports.
debug voip application stcapp port	Displays debug information for the components of the STCAPP for a specified port.

debug voip application stcapp functions

To debug SCCP Telephony Control Application (STCAPP) functions, use the **debug voip application stcapp functions**command in privileged EXEC mode. To disable STCAPP function debugging, use the **no** form of this command.

debug voip application stcapp functions

no debug voip application stcapp functions

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

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

Usage Guidelines

Use this command to display STCAPP entry and exit function calls for all voice ports.

Examples

The following example displays function calls for STCAPP ports 1/0/0 and 1/0/1:

```
Router# debug voip application stcapp functions
STCAPP function debugging is on
*Jan  9 06:55:27.583: 1/0/0: stcapp_setup_ind_eh
*Jan  9 06:55:27.591: 1/0/0: stcapp_cs_offhook_eh
*Jan  9 06:55:27.591: 1/0/0: stcapp_start_tone_eh
*Jan  9 06:55:27.591: 1/0/0: stcapp_report_digits_done_eh
*Jan  9 06:55:28.923: 1/0/0: stcapp_digit_end_eh
*Jan  9 06:55:28.927: 1/0/0: stcapp_stop_tone_eh
*Jan  9 06:55:29.063: 1/0/0: stcapp_digit_end_eh
*Jan  9 06:55:29.203: 1/0/0: stcapp_digit_end_eh
*Jan  9 06:55:29.343: 1/0/0: stcapp_digit_end_eh
*Jan  9 06:55:29.355: 1/0/0: stcapp_cs_proceed_eh
*Jan  9 06:55:29.359: 1/0/0: stcapp_proceed_call_info_eh
*Jan  9 06:55:29.359: 1/0/0: stcapp_start_tone_eh
*Jan  9 06:55:29.359: 1/0/0: stcapp_proceed_call_info_eh
*Jan  9 06:55:29.359: 1/0/1: stcapp_cs_ringin_eh
*Jan  9 06:55:29.359: 1/0/1: stcapp_call_info_eh
```

Related Commands

Command	Description
debug voip application stcapp all	Displays debug output for all the debug commands for the STCAPP compiled into one display.
debug voip application stcapp error	Displays STCAPP error log contents.

Command	Description
debug voip application stcapp events	Traces STCAPP call flow events.
debug voip application stcapp port	Displays debug information for the components of the STCAPP for a specified port.

debug voip application stcapp port

To enable SCCP Telephony Control Application (STCAPP) debugging for a specific port, use the **debug voip application stcapp port**command in privileged EXEC mode. To disable specific STCAPP port debugging, use the **no** form of this command.

debug voip application stcapp port *port-number*

no debug voip application stcapp port *port-number*

Syntax Description

port-number	Number of the port on the interface. See the appropriate platform manual or online help for port numbers on your networking device.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.3(14)T	This command was introduced.
12.4(4)T	Command output was enhanced to display modem transport method.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.
12.4(6)XE	Command output was enhanced to display fax relay, RFC 2833 DTMF digit relay, dial tone after remote onhook, call control feature mode and visual message waiting indicator (VMWI) information for SCCP analog ports.
12.4(11)T	This command was integrated into Cisco IOS Release 12.4(11)T.

Usage Guidelines

Use this command to display debugging information for the components of the STCAPP for a specified port.

Examples

The following example displays RFC 2833 DTMF digits messages sent and received on a voice gateway in Cisco IOS Release 12.4(6)XE:

```
Router# debug voip application stcapp port 2/3
```

The following lines show the SCCP gateway receiving the RFC payload.

```
Mar  4 00:23:31.166: 2/3   : ==> Received event:STCAPP_DC_EV_MEDIA_OPEN_RCV_CHNL
*Mar  4 00:23:31.166: 2/3   :     Call State:PROCEEDING
```

```
*Mar  4 00:23:31.166: 2/3   : stcapp_open_rcv_chnl_eh
*Mar  4 00:23:31.166: 2/3   :     call_ref=20797703
*Mar  4 00:23:31.166: 2/3   : stcapp_get_ccb_ptr
*Mar  4 00:23:31.166: 2/3   :     received ORC: rcv payload=101
*Mar  4 00:23:31.166: 2/3   : stcapp_set_up_voip_leg
*Mar  4 00:23:31.166: 2/3   : stcapp_get_ccb_ptr
*Mar  4 00:23:31.166: 2/3   : stcapp_set_up_modem_parms
```

The following lines show the SCCP gateway sending the RFC payload.

```
*Mar  4 00:23:31.174: 2/3   : ==> Received event:STCAPP_DC_EV_MEDIA_OPEN_XMT_CHNL
*Mar  4 00:23:31.174: 2/3   :     Call State:CONNECTING
*Mar  4 00:23:31.174: 2/3   : stcapp_start_media_eh
*Mar  4 00:23:31.174: 2/3   :     call_ref=20797703
*Mar  4 00:23:31.174: 2/3   : stcapp_get_ccb_ptr
*Mar  4 00:23:31.174: 2/3   :     lcb->mode 0, lcb->conf_status 0
*Mar  4 00:23:31.174: 2/3   :     received XMT: send payload=101
*Mar  4 00:23:31.174: 2/3   :     Start media: CCB Count:1 Call Leg Count:2
*Mar  4 00:23:31.174: 2/3   :     New State = ACTIVE_PENDING
```

The following example displays VMWI lamp state messages between Cisco Unified Communications Manager and a voice gateway in Cisco IOS Release 12.4(6)XE:

```
Router# debug voip application stcapp port 2/4
*Mar  1 01:41:58.395: 2/0   :     No state change... call remaining
*Mar 16 21:47:14.045: 2/4   : stcapp_screen_api_event
```

The following lines show the gateway receiving messages from Cisco Unified Communications Manager to activate the VMWI lamp.

```
*Mar 16 21:47:14.045: 2/4   :     event:STCAPP_DC_EV_DEVICE_SET_LAMP received.
*Mar 16 21:47:14.049: 2/4   :     msg_mwi 1, mwi 0, vmwi 0
*Mar 16 21:47:14.049: 2/4   :     event STCAPP_DC_EV_DEVICE_SET_LAMP_PROCESS_VMWI
created.
*Mar 16 21:47:14.049: 2/4   :     New State = VMWI_DSP_SETUP
*Mar 16 21:47:14.053: 2/4   : ==> Received event:STCAPP_CC_EV_CALL_PROCEEDING for
CallId: 229
*Mar 16 21:47:14.053: 2/4   :     Call State:VMWI_DSP_SETUP
*Mar 16 21:47:14.053: 2/4   : stcapp_vmwi_call_proceed_eh
*Mar 16 21:47:14.057: 2/4   :     No state change
*Mar 16 21:47:14.057: 2/4   : ==> Received event:STCAPP_CC_EV_CALL_MODIFY_DONE for
CallId: 229
*Mar 16 21:47:14.057: 2/4   :     Call State:VMWI_DSP_SETUP
*Mar 16 21:47:14.057: 2/4   : stcapp_vmwi_call_modify_done_eh
*Mar 16 21:47:14.057: 2/4   :     Sending ccCallFeature (vmwi = on)
*Mar 16 21:47:14.057: 2/4   :     New State = VMWI_PENDING
```

The following lines show that the VMWI activation is completed and call-tear down is beginning.

```
*Mar 16 21:47:15.237: 2/4   : ==> Received event:STCAPP_CC_EV_VBD_XMIT_DONE for CallId:229
*Mar 16 21:47:15.237: 2/4   :     Call State:VMWI_PENDING
*Mar 16 21:47:15.237: 2/4   : stcapp_vmwi_fsk_gen_done_eh
*Mar 16 21:47:15.237: 2/4   : stcapp_get_ccb_ptr
*Mar 16 21:47:15.237: 2/4   :     disconnect voice call leg
*Mar 16 21:47:15.237: 2/4   : stcapp_disconnect_call_leg
*Mar 16 21:47:15.237: 2/4   :     Sending ccCallDisconnect for VOICE_LEG with call id:229
```

The following example displays information on a call between ports 1/1/0 and 1/1/1 after port 1/1/1 hangs up and the automatic dial tone generation after remote onhook feature is enabled:

```
Router# debug voip application stcapp port 1/1/0
Jan  7 00:41:37.484: 1/1/0: ==> Received event:STCAPP_DC_EV_MEDIA_CLOSE_RCV_CHNL
*Jan  7 00:41:37.484: 1/1/0:    Call State:ACTIVE
```

The following lines show the call being disconnected.

```
*Jan  7 00:41:37.484: 1/1/0: stcapp_close_rcv_chnl_eh
```

```
*Jan  7 00:41:37.484: 1/1/0:    call_ref=209
*Jan  7 00:41:37.484: 1/1/0: stcapp_get_ccb_ptr
*Jan  7 00:41:37.484: 1/1/0:    Sending ccConferenceDestroy
*Jan  7 00:41:37.484: 1/1/0:    Sending ccCallDisconnect for voip call id:44
*Jan  7 00:41:37.484: 1/1/0: stcapp_disconnect_call_leg
*Jan  7 00:41:37.484: 1/1/0:    Sending ccCallDisconnect for VoIP_LEG with call id:44
*Jan  7 00:41:37.484: 1/1/0:    No state change
*Jan  7 00:41:37.488: 1/1/0: ==> Received event:STCAPP_DC_EV_MEDIA_CLOSE_XMT_CHNL
*Jan  7 00:41:37.488: 1/1/0:    Call State:ACTIVE
*Jan  7 00:41:37.488: 1/1/0: stcapp_active_close_xmt_chnl_eh
*Jan  7 00:41:37.488: 1/1/0:    lcb->mode 0 lcb->conf_status 0
*Jan  7 00:41:37.488: 1/1/0:    New State = ONHOOK_PEND
*Jan  7 00:41:37.488: 1/1/0: stcapp_cs_onhook_eh
*Jan  7 00:41:37.488: 1/1/0: stcapp_get_ccb_ptr
*Jan  7 00:41:37.488: 1/1/0:    call_ref=209, ccb=0x4662B31C, lcb->num_ccbs=1
*Jan  7 00:41:37.488: 1/1/0: stcapp_process_cs_onhook
*Jan  7 00:41:37.488: 1/1/0:    lcb->mode=CALL_BASIC (0)
```

The following lines show power denial-based supervisory disconnect signal being sent.

```
*Jan  7 00:41:37.488: 1/1/0:    Sending power denial signal to device 9
*Jan  7 00:41:37.488: 1/1/0: stcapp_update_dialtone_gen_trigger
*Jan  7 00:41:37.488: 1/1/0: stcapp_send_softkey_event
*Jan  7 00:41:37.488: 1/1/0:    Sending dcDeviceSoftKeyEvent(NewCall: event=2) for device
id: 9, call_ref: 0
*Jan  7 00:41:37.488: 1/1/0:    New State = REM_ONHOOK_PEND
*Jan  7 00:41:37.488: 1/1/0: ==> Received event:STCAPP_DC_EV_DEVICE_STOP_TONE
*Jan  7 00:41:37.488: 1/1/0:    Call State:REM_ONHOOK_PEND
*Jan  7 00:41:37.488: 1/1/0: stcapp_stop_tone_eh
*Jan  7 00:41:37.488: 1/1/0:    call_ref=209
*Jan  7 00:41:37.488: 1/1/0: stcapp_get_ccb_ptr
*Jan  7 00:41:37.488: 1/1/0:    Sending ccGenerateTone(NULL)
*Jan  7 00:41:37.488: 1/1/0:    No state change
*Jan  7 00:41:37.492: 1/1/0: ==> Received event:STCAPP_CC_EV_CONF_DESTROY_DONE for CallId:

41
*Jan  7 00:41:37.492: 1/1/0:    Call State:REM_ONHOOK_PEND
*Jan  7 00:41:37.492: 1/1/0:    Uninteresting event
*Jan  7 00:41:37.492: 1/1/0: stcapp_screen_api_event
*Jan  7 00:41:37.492: 1/1/0:    event:STCAPP_CC_EV_CALL_DISCONNECT_DONE received.
*Jan  7 00:41:37.492: STCAPP:Receive CC event:: call_id=44, ccb=0x4662B31C
*Jan  7 00:41:37.492: 1/1/0:     Received event:CC_EV_CALL_DISCONNECT_DONE for CallId: 44
*Jan  7 00:41:37.492: 1/1/0: stcapp_process_disconnect_done
*Jan  7 00:41:37.492: 1/1/0: stcapp_reset_call_leg
*Jan  7 00:41:37.492: 1/1/0:    ccb(0x4662B31C): voice/voip call_id=41/44, reset
call_id=44
*Jan  7 00:41:37.492: 1/1/0: stcapp_conn_db_delete_ccb
*Jan  7 00:41:37.492: 1/1/0:    ccb=0x4662B31C
*Jan  7 00:41:37.492: 1/1/0:    Disconnect Done: CCB Count:1 Call Leg Count:1
*Jan  7 00:41:37.496: 1/1/0: stcapp_get_dcb_and_lcb
```

The following lines show the call transitioning to off hook.

```
*Jan  7 00:41:37.496: 1/1/0: stcapp_screen_api_event
*Jan  7 00:41:37.496: 1/1/0:    event:STCAPP_DC_EV_DEVICE_CALL_STATE_OFFHOOK received.
*Jan  7 00:41:37.496: 1/1/0:    Create new event
STCAPP_DC_EV_DEV_CS_OFFHOOK_DIALTONE_GEN*Jan  7 00:41:37.496: 1/1/0: ==> Received
event:STCAPP_DC_EV_DEVICE_CALL_STATE_OFFHOOK_DIALTONE_GEN
*Jan  7 00:41:37.496: 1/1/0:    Call State:REM_ONHOOK_PEND
*Jan  7 00:41:37.496: 1/1/0: stcapp_cs_offhook_dialtone_gen_eh
*Jan  7 00:41:37.496: 1/1/0:    call_ref=211
*Jan  7 00:41:37.496: 1/1/0:    New State = OFFHOOK
*Jan  7 00:41:37.496: 1/1/0: ==> Received event:STCAPP_DC_EV_DEVICE_DISPLAY_PROMPT_STATUS
*Jan  7 00:41:37.496: 1/1/0:    Device State:IS
*Jan  7 00:41:37.496: 1/1/0: stcapp_display_prompt_status_eh
*Jan  7 00:41:37.496: 1/1/0:    lineNumber: 1
*Jan  7 00:41:37.496: 1/1/0:    call reference: 211
*Jan  7 00:41:37.496: 1/1/0:    promptStatus: ' '
*Jan  7 00:41:37.496: 1/1/0:    No state change
```

The following lines show the message to generate a dial tone for the new call.

```
*Jan  7 00:41:37.496: 1/1/0: ==> Received event:STCAPP_DC_EV_DEVICE_START_TONE
*Jan  7 00:41:37.496: 1/1/0:    Call State:OFFHOOK
*Jan  7 00:41:37.496: 1/1/0: stcapp_start_tone_eh
*Jan  7 00:41:37.496: 1/1/0: stcapp_get_ccb_ptr
*Jan  7 00:41:37.496: 1/1/0:    call_ref=211, ccb=0x4662B31C, tone=8(0x8), dir=1
*Jan  7 00:41:37.496: 1/1/0:    plar enable (0), hookflash (0)
*Jan  7 00:41:37.496: 1/1/0:    plar enable (0), tone (0x8), hookflash (0)
*Jan  7 00:41:37.496: 1/1/0:    Sending ccGenerateTone(8(0x8))  ? produce dial tone
*Jan  7 00:41:37.496: 1/1/0:    Sending ccCallReportDigits
*Jan  7 00:41:37.496: 1/1/0:    No state change
```

The following example displays information about a modem-relay call on a voice gateway in Cisco IOS Release 12.4(4)T:

Router# **debug voip application stcapp port 1/1/0**

```
*Jan 11 12:37:48.631: ==> Received event:STCAPP_CC_EV_CALL_SETUP_IND
(evId:CC_EV_CALL_SETUP_IND) for CallId: 326
*Jan 11 12:37:48.631: 1/1/0:     Call State:IDLE
*Jan 11 12:37:48.631: 1/1/0: stcapp_setup_ind_eh
*Jan 11 12:37:48.631: 1/1/0: stcapp_get_ccb
*Jan 11 12:37:48.631: 1/1/0:     dcb->lcb[line_inst - 1].num_ccbs=0
*Jan 11 12:37:48.631: 1/1/0:     Acquired CCB 0x65D932B8 for device id:7
*Jan 11 12:37:48.631: 1/1/0:     num_ccbs++, num_ccbs=1
*Jan 11 12:37:48.631: 1/1/0:     Voice Setup: callID:326, vdb_ptr:666581AC
*Jan 11 12:37:48.631: 1/1/0:     Sending StationOffHook to CallManager
*Jan 11 12:37:48.631: 1/1/0:     Sending ccCallSetupAck to Symphony for voice call id:326
*Jan 11 12:37:48.631: 1/1/0:     New State = OFFHOOK
*Jan 11 12:37:48.643: 1/1/0:     No line (line=0) found... most likely old Call Ref: event
 STCAPP_DC_EV_DEVICE_SET_RINGER
*Jan 11 12:37:48.643: ==> Received event:STCAPP_DC_EV_DEVICE_CALL_STATE_OFFHOOK
(evID:DC_EV_DEVICE_CALL_STATE_OFFHOOK)
*Jan 11 12:37:48.643: 1/1/0:     Call State:OFFHOOK
*Jan 11 12:37:48.643: 1/1/0: stcapp_cs_offhook_eh
*Jan 11 12:37:48.643: 1/1/0:     call_ref=16777250
*Jan 11 12:37:48.643: 1/1/0: stcapp_get_ccb_ptr
*Jan 11 12:37:48.643: 1/1/0: stcapp_get_ccb_ptr
*Jan 11 12:37:48.643: 1/1/0:     Using call_ref 0 to get ccb=0x65D932B8
*Jan 11 12:37:48.643: 1/1/0:     No state change
*Jan 11 12:37:48.643: ==> Received event:STCAPP_DC_EV_DEVICE_DISPLAY_PROMPT_STATUS
*Jan 11 12:37:48.643: 1/1/0:     Device State:IS
*Jan 11 12:37:48.643: 1/1/0: stcapp_display_prompt_status_eh
*Jan 11 12:37:48.643: 1/1/0:     lineNumber: 1
*Jan 11 12:37:48.643: 1/1/0:     call reference: 16777250
*Jan 11 12:37:48.643: 1/1/0:     promptStatus: Enter Number
*Jan 11 12:37:48.643: 1/1/0:     No state change
*Jan 11 12:37:48.643: ==> Received event:STCAPP_DC_EV_DEVICE_START_TONE
(evID:DC_EV_DEVICE_START_TONE)
*Jan 11 12:37:48.643: 1/1/0:     Call State:OFFHOOK
*Jan 11 12:37:48.643: 1/1/0: stcapp_start_tone_eh
*Jan 11 12:37:48.643: 1/1/0: stcapp_get_ccb_ptr
*Jan 11 12:37:48.643: 1/1/0:     call_ref=16777250, ccb=0x65D932B8, tone=8(0x8)
*Jan 11 12:37:48.643: 1/1/0:     Sending ccGenerateTone(8(0x8))
*Jan 11 12:37:48.643: 1/1/0:     Sending ccCallReportDigits
*Jan 11 12:37:48.643: 1/1/0:     No state change
*Jan 11 12:37:48.643: ==> Received event:STCAPP_CC_EV_CALL_REPORT_DIGITS_DONE
(evId:CC_EV_CALL_REPORT_DIGITS_DONE) for CallId: 326
*Jan 11 12:37:48.647: 1/1/0:     Call State:OFFHOOK
*Jan 11 12:37:48.647: 1/1/0: stcapp_report_digits_done_eh
*Jan 11 12:37:48.647: 1/1/0:     No state change
*Jan 11 12:37:52.643: ==> Received event:STCAPP_CC_EV_CALL_DIGIT_BEGIN
(evId:CC_EV_CALL_DIGIT_BEGIN) for CallId: 326
*Jan 11 12:37:52.643: 1/1/0:     Call State:OFFHOOK
*Jan 11 12:37:52.643: 1/1/0:     Uninteresting event
*Jan 11 12:37:52.683: ==> Received event:STCAPP_CC_EV_CALL_DIGIT_END
(evId:CC_EV_CALL_DIGIT_END) for CallId: 326
*Jan 11 12:37:52.683: 1/1/0:     Call State:OFFHOOK
*Jan 11 12:37:52.683: 1/1/0: stcapp_digit_end_eh
```

```
*Jan 11 12:37:52.683: 1/1/0:     Digit received is (5)
*Jan 11 12:37:52.683: 1/1/0:     Sending StationKeypadButton(5) to CallManager
*Jan 11 12:37:52.683: 1/1/0:     No state change
*Jan 11 12:37:52.687: ==> Received event:STCAPP_DC_EV_DEVICE_STOP_TONE
(evID:DC_EV_DEVICE_STOP_TONE)
*Jan 11 12:37:52.687: 1/1/0:     Call State:OFFHOOK
*Jan 11 12:37:52.687: 1/1/0: stcapp_stop_tone_eh
*Jan 11 12:37:52.687: 1/1/0:     call_ref=16777250
*Jan 11 12:37:52.687: 1/1/0: stcapp_get_ccb_ptr
*Jan 11 12:37:52.687: 1/1/0:     Sending ccGenerateTone(NULL)
*Jan 11 12:37:52.687: 1/1/0:     No state change
*Jan 11 12:37:52.775: ==> Received event:STCAPP_CC_EV_CALL_DIGIT_BEGIN
(evId:CC_EV_CALL_DIGIT_BEGIN) for CallId: 326
*Jan 11 12:37:52.775: 1/1/0:     Call State:OFFHOOK
*Jan 11 12:37:52.775: 1/1/0:     Uninteresting event
*Jan 11 12:37:52.823: ==> Received event:STCAPP_CC_EV_CALL_DIGIT_END
(evId:CC_EV_CALL_DIGIT_END) for CallId: 326
*Jan 11 12:37:52.823: 1/1/0:     Call State:OFFHOOK
*Jan 11 12:37:52.823: 1/1/0: stcapp_digit_end_eh
*Jan 11 12:37:52.823: 1/1/0:     Digit received is (8)
*Jan 11 12:37:52.823: 1/1/0:     Sending StationKeypadButton(8) to CallManager
*Jan 11 12:37:52.823: 1/1/0:     No state change
*Jan 11 12:37:52.923: ==> Received event:STCAPP_CC_EV_CALL_DIGIT_BEGIN
(evId:CC_EV_CALL_DIGIT_BEGIN) for CallId: 326
*Jan 11 12:37:52.923: 1/1/0:     Call State:OFFHOOK
*Jan 11 12:37:52.923: 1/1/0:     Uninteresting event
*Jan 11 12:37:52.963: ==> Received event:STCAPP_CC_EV_CALL_DIGIT_END
(evId:CC_EV_CALL_DIGIT_END) for CallId: 326
*Jan 11 12:37:52.963: 1/1/0:     Call State:OFFHOOK
*Jan 11 12:37:52.963: 1/1/0: stcapp_digit_end_eh
*Jan 11 12:37:52.963: 1/1/0:     Digit received is (0)
*Jan 11 12:37:52.963: 1/1/0:     Sending StationKeypadButton(0) to CallManager
*Jan 11 12:37:52.963: 1/1/0:     No state change
*Jan 11 12:37:53.063: ==> Received event:STCAPP_CC_EV_CALL_DIGIT_BEGIN
(evId:CC_EV_CALL_DIGIT_BEGIN) for CallId: 326
*Jan 11 12:37:53.063: 1/1/0:     Call State:OFFHOOK
*Jan 11 12:37:53.063: 1/1/0:     Uninteresting event
*Jan 11 12:37:53.103: ==> Received event:STCAPP_CC_EV_CALL_DIGIT_END
(evId:CC_EV_CALL_DIGIT_END) for CallId: 326
*Jan 11 12:37:53.103: 1/1/0:     Call State:OFFHOOK
*Jan 11 12:37:53.103: 1/1/0: stcapp_digit_end_eh
*Jan 11 12:37:53.103: 1/1/0:     Digit received is (2)
*Jan 11 12:37:53.103: 1/1/0:     Sending StationKeypadButton(2) to CallManager
*Jan 11 12:37:53.103: 1/1/0:     No state change
*Jan 11 12:37:53.235: ==> Received event:STCAPP_DC_EV_DEVICE_CALL_STATE_PROCEED
(evID:DC_EV_DEVICE_CALL_STATE_PROCEED)
*Jan 11 12:37:53.235: 1/1/0:     Call State:OFFHOOK
*Jan 11 12:37:53.235: 1/1/0: stcapp_cs_proceed_eh
*Jan 11 12:37:53.235: 1/1/0: stcapp_get_ccb_ptr
*Jan 11 12:37:53.235: 1/1/0:     Sending ccCallProceeding for voice call id:326
*Jan 11 12:37:53.235: 1/1/0:     Stopping the initial and inter digit timer!
*Jan 11 12:37:53.235: 1/1/0:     New State = PROCEEDING
*Jan 11 12:37:53.235: ==> Received event:STCAPP_DC_EV_DEVICE_CALL_INFO
(evID:DC_EV_DEVICE_CALL_INFO)
*Jan 11 12:37:53.235: 1/1/0:     Call State:PROCEEDING
*Jan 11 12:37:53.235: 1/1/0: stcapp_proceed_call_info_eh
*Jan 11 12:37:53.235: 1/1/0: stcapp_get_ccb_ptr
*Jan 11 12:37:53.239: 1/1/0:     No state change
*Jan 11 12:37:53.239: ==> Received event:STCAPP_DC_EV_DEVICE_START_TONE
(evID:DC_EV_DEVICE_START_TONE)
*Jan 11 12:37:53.239: 1/1/0:     Call State:PROCEEDING
*Jan 11 12:37:53.239: 1/1/0: stcapp_start_tone_eh
*Jan 11 12:37:53.239: 1/1/0: stcapp_get_ccb_ptr
*Jan 11 12:37:53.239: 1/1/0:     call_ref=16777250, ccb=0x65D932B8, tone=1(0x1)
*Jan 11 12:37:53.239: 1/1/0:     Sending ccCallAlert(signal:1) for voice call id:326
*Jan 11 12:37:53.239: 1/1/0:     No state change
*Jan 11 12:37:53.239: ==> Received event:STCAPP_DC_EV_DEVICE_CALL_STATE_RINGOUT
(evID:DC_EV_DEVICE_CALL_STATE_RINGOUT)
*Jan 11 12:37:53.239: 1/1/0:     Call State:PROCEEDING
*Jan 11 12:37:53.239: 1/1/0: stcapp_set_call_state_eh
*Jan 11 12:37:53.239: 1/1/0:     call_ref=16777250, call_state=2
*Jan 11 12:37:53.239: 1/1/0: stcapp_get_ccb_ptr
*Jan 11 12:37:53.239: 1/1/0:     No state change
```

```
*Jan 11 12:37:53.239: ==> Received event:STCAPP_DC_EV_DEVICE_DISPLAY_PROMPT_STATUS
*Jan 11 12:37:53.239: 1/1/0:     Device State:IS
*Jan 11 12:37:53.239: 1/1/0: stcapp_display_prompt_status_eh
*Jan 11 12:37:53.239: 1/1/0:     lineNumber: 1
*Jan 11 12:37:53.239: 1/1/0:     call reference: 16777250
*Jan 11 12:37:53.239: 1/1/0:     promptStatus: Ring Out
*Jan 11 12:37:53.239: 1/1/0:     No state change
*Jan 11 12:37:53.239: ==> Received event:STCAPP_DC_EV_DEVICE_CALL_INFO
(evID:DC_EV_DEVICE_CALL_INFO)
*Jan 11 12:37:53.239: 1/1/0:     Call State:PROCEEDING
*Jan 11 12:37:53.239: 1/1/0: stcapp_proceed_call_info_eh
*Jan 11 12:37:53.239: 1/1/0: stcapp_get_ccb_ptr
*Jan 11 12:37:53.239: 1/1/0:     No state change
*Jan 11 12:37:56.635: ==> Received event:STCAPP_DC_EV_DEVICE_STOP_TONE
(evID:DC_EV_DEVICE_STOP_TONE)
*Jan 11 12:37:56.635: 1/1/0:     Call State:PROCEEDING
*Jan 11 12:37:56.635: 1/1/0: stcapp_stop_tone_eh
*Jan 11 12:37:56.635: 1/1/0:     call_ref=16777250
*Jan 11 12:37:56.635: 1/1/0: stcapp_get_ccb_ptr
*Jan 11 12:37:56.639: 1/1/0:     Sending ccGenerateTone(NULL)
*Jan 11 12:37:56.639: 1/1/0:     No state change
*Jan 11 12:37:56.639: ==> Received event:STCAPP_DC_EV_MEDIA_OPEN_RCV_CHNL
(evID:DC_EV_MEDIA_OPEN_RCV_CHNL)
*Jan 11 12:37:56.639: 1/1/0:     Call State:PROCEEDING
*Jan 11 12:37:56.639: 1/1/0: stcapp_open_rcv_chnl_eh
*Jan 11 12:37:56.639: 1/1/0:     call_ref=16777250
*Jan 11 12:37:56.639: 1/1/0: stcapp_get_ccb_ptr
*Jan 11 12:37:56.639: 1/1/0: stcapp_set_up_voip_leg
*Jan 11 12:37:56.639: 1/1/0: stcapp_get_ccb_ptr
*Jan 11 12:37:56.639: 1/1/0:     Codec: 5 ptime :20, codecbytes: 160
```

The following line indicates the modem transport method that will be used:

```
*Jan 11 12:37:56.639: 1/1/0:     CCM directive -> enabling modem relay
```

The following lines show modem relay parameters:

```
*Jan 11 12:37:56.639: 1/1/0:     MR parms: sprt_retries=10, sprt_latency=250,
sprt_rx_v14_pb_hold_time=32, sprt_tx_v14_hold_time=12, sprt_tx_v14_hold_count=22, gw_xid=1,
 dictsize=1024, stringlen=16, compressdir=3, sse_red_interval=16, sse_red_pkt_count=2,
sse_t1=2100, sse_retries=5
*Jan 11 12:37:56.639: 1/1/0:     Info provided to RTPSPI - sess_mode 2, desired_qos 0, codec
 5, pkt_period 20, lr_port 17180
*Jan 11 12:37:56.639: 1/1/0:     Sending ccIFCallSetupRequest for voip leg
*Jan 11 12:37:56.639: 1/1/0:     ccIFCallSetRequest returned voip call id:327
*Jan 11 12:37:56.639: 1/1/0:     Sending dcDeviceOpenReceiveChannelAck
*Jan 11 12:37:56.639: 1/1/0:     ORChnlAck Info: codec:5, loc_port:17180, chnl_id:16777521
*Jan 11 12:37:56.639: 1/1/0:     New State = CONNECTING
*Jan 11 12:37:56.643: ==> Received event:STCAPP_DC_EV_DEVICE_CALL_STATE_CONNECTED
(evID:DC_EV_DEVICE_CALL_STATE_CONNECTED)
*Jan 11 12:37:56.643: 1/1/0:     Call State:CONNECTING
*Jan 11 12:37:56.643: 1/1/0: stcapp_set_call_state_eh
*Jan 11 12:37:56.643: 1/1/0:     call_ref=16777250, call_state=6
*Jan 11 12:37:56.643: 1/1/0: stcapp_get_ccb_ptr
*Jan 11 12:37:56.643: 1/1/0:     No state change
*Jan 11 12:37:56.643: ==> Received event:STCAPP_DC_EV_DEVICE_DISPLAY_PROMPT_STATUS
*Jan 11 12:37:56.643: 1/1/0:     Device State:IS
*Jan 11 12:37:56.643: 1/1/0: stcapp_display_prompt_status_eh
*Jan 11 12:37:56.643: 1/1/0:     lineNumber: 1
*Jan 11 12:37:56.643: 1/1/0:     call reference: 16777250
*Jan 11 12:37:56.643: 1/1/0:     promptStatus: Connected
*Jan 11 12:37:56.643: 1/1/0:     No state change
*Jan 11 12:37:56.643: ==> Received event:STCAPP_DC_EV_DEVICE_CALL_INFO
(evID:DC_EV_DEVICE_CALL_INFO)
*Jan 11 12:37:56.643: 1/1/0:     Call State:CONNECTING
*Jan 11 12:37:56.643: 1/1/0: stcapp_conn_call_info_eh
*Jan 11 12:37:56.647: 1/1/0: stcapp_get_ccb_ptr
*Jan 11 12:37:56.647: 1/1/0:     stcapp_call_info_eh::caller_name=
*Jan 11 12:37:56.647: 1/1/0:     Irrelevant CALL_INFO message is ignore!
*Jan 11 12:37:56.647: 1/1/0:     No state change
*Jan 11 12:37:56.647: ==> Received event:STCAPP_DC_EV_DEVICE_STOP_TONE
(evID:DC_EV_DEVICE_STOP_TONE)
*Jan 11 12:37:56.647: 1/1/0:     Call State:CONNECTING
```

```
*Jan 11 12:37:56.647: 1/1/0: stcapp_stop_tone_eh
*Jan 11 12:37:56.647: 1/1/0:     call_ref=16777250
*Jan 11 12:37:56.647: 1/1/0: stcapp_get_ccb_ptr
*Jan 11 12:37:56.647: 1/1/0:     Sending ccGenerateTone(NULL)
*Jan 11 12:37:56.647: 1/1/0:     No state change
*Jan 11 12:37:56.647: ==> Received event:STCAPP_DC_EV_MEDIA_OPEN_XMT_CHNL
(evID:DC_EV_MEDIA_OPEN_XMT_CHNL)
*Jan 11 12:37:56.647: 1/1/0:     Call State:CONNECTING
*Jan 11 12:37:56.647: 1/1/0: stcapp_start_media_eh
*Jan 11 12:37:56.647: 1/1/0:     call_ref=16777250
*Jan 11 12:37:56.647: 1/1/0: stcapp_get_ccb_ptr
*Jan 11 12:37:56.647: 1/1/0:     New State = ACTIVE_PENDING
*Jan 11 12:37:56.647: ==> Received event:STCAPP_CC_EV_CALL_CONNECTED
(evId:CC_EV_CALL_CONNECTED) for CallId: 327
*Jan 11 12:37:56.647: 1/1/0:     Call State:ACTIVE_PENDING
*Jan 11 12:37:56.647: 1/1/0: stcapp_call_connected_eh
*Jan 11 12:37:56.647: 1/1/0: stcapp_create_conference
*Jan 11 12:37:56.647: 1/1/0:     Sending ccConferenceCreate to Symphony
*Jan 11 12:37:56.651: 1/1/0:     Conference created. voice call id:326, voip call id:327
*Jan 11 12:37:56.651: 1/1/0:     No state change
*Jan 11 12:37:56.651: ==> Received event:STCAPP_CC_EV_CONF_CREATE_DONE
(evId:CC_EV_CONF_CREATE_DONE) for CallId: 326
*Jan 11 12:37:56.651: 1/1/0:     Call State:ACTIVE_PENDING
*Jan 11 12:37:56.651: 1/1/0: stcapp_active_pending_eh
*Jan 11 12:37:56.651: 1/1/0:     Sending ccCallModify for voice call id:326
*Jan 11 12:37:56.651: 1/1/0:     codec=5, vad=0
*Jan 11 12:37:56.651: 1/1/0:     Stopping the initial and inter digit timer!
*Jan 11 12:37:56.651: 1/1/0:     Sending ccCallModify for voip call id:327
*Jan 11 12:37:56.651: 1/1/0:     Updated SMT info to RTPSPI - sess_mode:3,desired_qos:0,
codec:5, pkt_period:20,rem_port:18968 vad:0 ip_tos:4
*Jan 11 12:37:56.655: 1/1/0:     No state change
*Jan 11 12:37:56.655: ==> Received event:STCAPP_CC_EV_VOICE_MODE_DONE
(evId:CC_EV_VOICE_MODE_DONE) for CallId: 326
*Jan 11 12:37:56.655: 1/1/0:     Call State:ACTIVE_PENDING
*Jan 11 12:37:56.655: 1/1/0:     Uninteresting event
*Jan 11 12:37:56.655: ==> Received event:STCAPP_CC_EV_CALL_REPORT_DIGITS_DONE
(evId:CC_EV_CALL_REPORT_DIGITS_DONE) for CallId: 326
*Jan 11 12:37:56.655: 1/1/0:     Call State:ACTIVE_PENDING
*Jan 11 12:37:56.655: 1/1/0:     Uninteresting event
*Jan 11 12:37:56.655: ==> Received event:STCAPP_CC_EV_CALL_MODIFY_DONE
(evId:CC_EV_CALL_MODIFY_DONE) for CallId: 326
*Jan 11 12:37:56.655: 1/1/0:     Call State:ACTIVE_PENDING
*Jan 11 12:37:56.655: 1/1/0: stcapp_default_eh
*Jan 11 12:37:56.655: 1/1/0:     call_ref=0, call_state=0
*Jan 11 12:37:56.655: 1/1/0:     New State = ACTIVE
*Jan 11 12:37:56.655: ==> Received event:STCAPP_CC_EV_CALL_MODIFY_DONE
(evId:CC_EV_CALL_MODIFY_DONE) for CallId: 327
*Jan 11 12:37:56.655: 1/1/0:     Call State:ACTIVE
*Jan 11 12:37:56.655: 1/1/0:     Uninteresting event
*Jan 11 12:37:59.963: ==> Received event:STCAPP_CC_EV_CALL_FEATURE_OFFHOOK
(evId:CC_EV_CALL_FEATURE) for CallId: 326
*Jan 11 12:37:59.963: 1/1/0:     Call State:ACTIVE
*Jan 11 12:37:59.963: 1/1/0: stcapp_call_feature_eh
*Jan 11 12:37:59.963: 1/1/0:     lcb->num_ccbs = 1
*Jan 11 12:37:59.963: 1/1/0:     No CC_FEATURE match!
*Jan 11 12:37:59.967: 1/1/0:     No state change... call remaining
```

The following example displays information on STCAPP controlled FXS port 1/0/1 during call setup:

```
Router# debug voip application stcapp port 1/0/1
stcapp port debugging is on
```

The following lines show the voice gateway receiving notification from the Cisco Unified Communications Manager of an incoming call:

```
*Jan  9 06:57:24.403: ==> Received event:STCAPP_DC_EV_DEVICE_CALL_STATE_RINGIN
(evID:DC_EV_DEVICE_CALL_STATE_RINGIN)
*Jan  9 06:57:24.403: 1/0/1:     Call State:IDLE
*Jan  9 06:57:24.403: 1/0/1: stcapp_cs_ringin_eh
*Jan  9 06:57:24.407: 1/0/1:     Acquired CCB 0x66C0A428 for device id:3
```

The next lines show the new call processing state for the port.

```
*Jan  9 06:57:24.407: 1/0/1:     New State = RINGIN
*Jan  9 06:57:24.407: ==> Received event:STCAPP_DC_EV_DEVICE_CALL_INFO
(evID:DC_EV_DEVICE_CALL_INFO)
*Jan  9 06:57:24.407: 1/0/1:     Call State:RINGIN
*Jan  9 06:57:24.407: 1/0/1: stcapp_call_info_eh
```

The next lines show the application sending a call setup request for the telephony leg.

```
*Jan  9 06:57:24.407: 1/0/1: stcapp_set_up_voice_leg
*Jan  9 06:57:24.407: 1/0/1:     Sending ccIFCallSetupRequest for voice leg
*Jan  9 06:57:24.407: 1/0/1:     ccIFCallSetRequest returned voice call id:22.
CdPN:7702CgPN:7701
*Jan  9 06:57:24.407: 1/0/1:     No state change
```

The next lines show the application invoking the ringing state.

```
*Jan  9 06:57:24.407: ==> Received event:STCAPP_DC_EV_DEVICE_SET_RINGER
(evID:DC_EV_DEVICE_SET_RINGER)
*Jan  9 06:57:24.407: 1/0/1:     Call State:RINGIN
*Jan  9 06:57:24.407: 1/0/1: stcapp_set_ringer_eh
*Jan  9 06:57:24.407: 1/0/1: stcapp_set_ring_mode
*Jan  9 06:57:24.407: 1/0/1:     SCCP ring mode:2
*Jan  9 06:57:24.407: 1/0/1:     Invoking Feature:12. Mode:0 for callid:22
*Jan  9 06:57:24.407: 1/0/1:     No state change
```

Related Commands

Command	Description
debug voip application stcapp all	Displays debug output for all the debug commands for the STCAPP compiled into one display.
debug voip application stcapp error	Displays STCAPP error log contents.
debug voip application stcapp events	Traces STCAPP call flow events.
debug voip application stcapp functions	Displays STCAPP entry and exit function calls for all voice ports.

debug voip application vxml

To troubleshoot a VoiceXML application, use the **debug voip application vxml** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voip application vxml [**all**| **application**| **background**| **default**| **error** [**call [informational]**| **software [informational]**]| **event**| **function**| **grammar**| **gtd**| **inout**| **log**| **puts**| **ssml**| **trace**| **warning**]

no debug voip application vxml

Syntax Description

all	(Optional) Displays all VoiceXML debugging messages.
application	(Optional) Displays VoiceXML application states information.
background	(Optional) Displays VoiceXML background messages.
default	(Optional) Displays output for all of the following keywords: • **application** • **background** • **error** • **event** • **gtd** • **inout** • **puts** • **trace** • **warning** This option is also available if no keywords are added.
error	(Optional) Displays VoiceXML errors.
call	(Optional) Displays call processing errors.
informational	(Optional) Displays minor errors and major errors. Without the **informational** keyword, only major errors are displayed.
software	(Optional) Displays software errors.

event	(Optional) Displays VoiceXML asynchronous events.
function	(Optional) Displays VoiceXML functions.
grammar	(Optional) Enables syntax checking of XML grammar by the VoiceXML interpreter and displays syntax debugging messages.
gtd	(Optional) Displays VoiceXML generic transparency descriptors.
inout	(Optional) Displays VoiceXML in/out functions.
log	(Optional) Displays the results of the VoiceXML <log> tag.
puts	(Optional) Displays the results of VoiceXML <cisco-puts> and <cisco-putvar> tags. **Note** In Cisco IOS Release 12.4(6th)T and later releases, the **puts** keyword is obsolete. Use the **log** keyword instead.
ssml	(Optional) Enables syntax checking of Speech Synthesis Markup Language (SSML) by the VoiceXML interpreter and displays syntax debugging messages.
trace	(Optional) Displays a trace of all activities for the current VoiceXML document.
warning	(Optional) Displays VoiceXML warning messages.

Command Default

Debugging is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(8)T	This command replaces the **debug vxml** command.
12.4(15)T	The **puts** keyword was obsoleted. The **log** keyword was added to replace it.

Usage Guidelines

If this debug encounters a fetch failure when using an HTTP interface, an **error.badfetch.http**.*response code* message is displayed. The values for the response code are shown in the table below.

Table 26: error.badfetch.http Response Codes

Response Code	Description
0	No response from HTTP server
400	Bad request
401	Unauthorized
402	Payment required
403	Forbidden
404	Not found
405	Method not allowed
406	Not acceptable
407	Proxy authentication required
408	Request timeout
409	Conflict
410	Gone
411	Length required
412	Precondition failed
413	Request entity too large
414	Request-URI too large
415	Unsupported media type
416	Requested range not satisfiable
417	Expectation failed
500	Internal server error
501	Not implemented

Response Code	Description
502	Bad gateway
503	Service unavailable
504	Gateway timeout
505	Version not supported

Examples

The following is sample output from the **debug voip application vxml all** command if there is an HTTP badfetch error call:

```
Router# debug voip application vxml all
Aug  7 04:53:03.003: //-1/000000000000/VAPP:/vapp_evt_handler:
   State VAPP_ACTIVE got event CC_EV_CALL_SETUP_IND
Aug  7 04:53:03.003: //-1/000000000000/VAPP:/vapp_driver:
   pInterp[6383BA48]:
Aug  7 04:53:03.003: //-1/000000000000/VAPP:/vapp_driver:
   evtID: 29 vapp record state: 0
Aug  7 04:53:03.003: //-1/000000000000/VAPP:/vapp_evt_setup:
Aug  7 04:53:03.003: //-1//VAPP:/vapp_incoming_callblock:
Aug  7 04:53:03.003: vapp_incoming_callblock:
```

Before the incoming call block data comes in, the CallEntry ID is -1, which indicates that the call leg had not been identified. In the next excerpt, the call leg is shown as 1 and the GUID is also assigned.

```
Aug  7 04:53:03.003: //1/71E56a9AF8002/VAPP:/vapp_evt_setup:
   VXML call. GTD should be saved
Aug  7 04:53:03.003: //1/71E569AF8002/VAPP:/vapp_load_or_run_script:
Aug  7 04:53:03.003: //1/71E569AF8002/VAPP:/vapp_load_or_run_script:
```

The next excerpt show script-specific information.

```
Aug  7 04:53:03.007: The VXML Script with len=1519 starts:
-------------------------------------
<?xml version="3.0" encoding="iso-8859-1"?>
<vxml version="3.0">
 <form id = "transfer_me">
 <catch event="telephone.disconnect.transfer">
 </catch>
<var name="phone_num" expr="5550100"/>
        <v
```

The CallEntry ID becomes 0 in the following excerpt, which indicates that the output is from the application server, not a call leg.

```
Aug  7 04:53:03.007: //0//VXML:/vxml_start_element_handler: Enter
Aug  7 04:53:03.007: //0//VXML:/vxml_start_element_handler: Exit
Aug  7 04:53:03.007: //0//VXML:/vxml_character_data: Enter
Aug  7 04:53:03.007: //0//VXML:/vxml_character_data:
   at line 888: length <=0, exit
.
.
.
Aug  7 04:53:03.051: //0//VXML:/vxml_end_element_handler: Enter
Aug  7 04:53:03.051: //0//VXML:/vxml_end_element_handler: Exit
Aug  7 04:53:03.051: //0//VXML:/vxml_parse:

Aug  7 04:53:03.051: vxml_parse: XML_Parse success err=0
Aug  7 04:53:03.051: //0//VXML:/vxml_session_delete:

Aug  7 04:53:03.051: vxml_session_delete:mem_mgr_mempool_free: mempool=NULL
```

```
Aug  7 04:53:03.051: //-1//VXML:/vxml_create:
   enter url=tftp://dirt/jkuo/vxml/xfer.nosound.vxml tree_handle=63282BDC
   return_handle_add=63C84F80
```

In the following excerpt, the call with the GUID 71E569AF8002 is again being tracked as the application session is initiated.

```
Aug  7 04:53:03.083: //1/71E569AF8002/VXML:/vxml_offramp_mailhdrs_get:

Aug  7 04:53:03.087: //1/71E569AF8002/VXML:/vxml_create_gtd_sess_vars:
   Created object chain for com.cisco.signal.gtdlist
Aug  7 04:53:03.087: //1/71E569AF8002/VXML:/vxml_jse_add_gtd_obj_to_list:
   Sig-event name = setup_indication, gtd-len = 140, gtd-buf =
IAM,
PRN,isdn*,,,
USI,rate,c,s,c,1
USI,lay1,ulaw
TMR,00
CPN,34,,4,52950
CPC,09
FCI,,,,,,,y,
GCI,71e569af6b5511d4800200014232e6a8
Aug  7 04:53:03.087: //1/71E569AF8002/VXML:/vxml_jse_add_gtd_obj_to_list:
   gtd_obj for sig-event [setup_indication] added to session/shadow
   var array [0x63826914]
Aug  7 04:53:03.087: //1/71E569AF8002/VXML:/vxml_create: Exit
Aug  7 04:53:03.087: //1/71E569AF8002/VXML:/vxml_start:
   vxmlhandle=6372E9BC vapphandle=6383BA48 status=0 async_status=0
Aug  7 04:53:03.087: //1/71E569AF8002/VXML:/vxml_vxml_proc:
<vxml> URI(abs):tftp://dirt/jkuo/vxml/xfer.nosound.vxml
   scheme=tftp
   host=dirt
   path=/jkuo/vxml/xfer.nosound.vxml
   base= URI(abs):tftp://dirt/jkuo/vxml/xfer.nosound.vxml
   scheme=tftp
   host=dirt
   path=/jkuo/vxml/xfer.nosound.vxml lang=none version=3.0
Aug  7 04:53:03.087: //1/71E569AF8002/VXML:/vxml_form_proc:
Aug  7 04:53:03.087:  <form>: id=transfer_me   scope=dialog
```

In the following excerpt, the phone number of the caller is shown:

```
Aug  7 04:53:03.087: vxml_form_init current scope: dialog
 <var>: namep=phone_num expr=5550100
Aug  7 04:53:03.091: //1/71E569AF8002/VXML:/vxml_expr_eval:
   expr=var phone_num=5550100
 <var>: namep=mydur
Aug  7 04:53:03.095: //1/71E569AF8002/VXML:/vxml_expr_eval:
   expr=var mydur
vxml_counter_reset:
Aug  7 04:53:03.095: //1/71E569AF8002/VXML:/vxml_formitem_select:
   Status=VXML_STATUS_OK,
Aug  7 04:53:03.095: //1/71E569AF8002/VXML:/vxml_formitem_select:
    AsyncStatus=VXML_STATUS_OK
Aug  7 04:53:03.099: //1/71E569AF8002/VXML:/vxml_transfer_proc:
  <transfer>:
```

In the following excerpts, the attributes of the incoming phone call are shown:

```
Aug  7 04:53:03.099: //1/71E569AF8002/VXML:/vxml_item_attrs_proc:
   name=mycall dest_expr='phone://'+ phone_num bridge=1 connecttimeout=50 maxtime=50
desttype=-1 destplan=-1 anitype=-1 aniplan=-1 anipi=-1, anisi=-1 rdn_exprp='phone://' +
4085550111 rdntype=2 rdnplan=1 rdnpi=0, rdnsi=3, redirectreason=0
```

The next several excerpts show the initialization and playing of audio prompts. When troubleshooting voice applications, ensure that all your assigned prompts play when required.

```
Aug  7 04:53:03.099: //1/71E569AF8002/VXML:/vxml_play_prompts: Enter
Aug  7 04:53:03.099: //1/71E569AF8002/VXML:/vxml_prompt_proc: Enter
Aug  7 04:53:03.099: //1/71E569AF8002/VXML:/vxml_prompt_proc:
   <prompt>:(default_prompt) bargein=1 count=1 typeaheadflush=0
Aug  7 04:53:03.099: //1/71E569AF8002/VXML:/vxml_audio_proc: Enter
```

```
Aug  7 04:53:03.099: //1/71E569AF8002/VXML:/vxml_audio_proc:
    <audio>: URI(abs):http://px1-sun/nosound.au
   scheme=http
   host=px1-sun
   path=/nosound.au caching=fast fetchhint=invalid fetchtimeout=0
Aug  7 04:53:03.099: //1/71E569AF8002/VXML:/vxml_audio_proc: Exit
Aug  7 04:53:03.099: //1/71E569AF8002/VXML:/vxml_vapp_media_play:
   bargein=1 timeout=0 typeaheadflush=0 vcr=0 rate=0
Aug  7 04:53:03.099: //1/71E569AF8002/VXML:/vxml_vapp_media_play:
   str=http://px1-sun/nosound.au cachable=1 timeout0
Aug  7 04:53:03.099: //1/71E569AF8002/VAPP:/vapp_media_play:
Aug  7 04:53:03.099: //1/71E569AF8002/VAPP:/vapp_media_play:
   prompt=http://px1-sun/nosound.au:
Aug  7 04:53:03.687: //1/71E569AF8002/VXML:/vxml_vapp_media_play: Exit
Aug  7 04:53:03.687: //1/71E569AF8002/VXML:/vxml_prompt_proc: Exit
Aug  7 04:53:03.687: //1/71E569AF8002/VXML:/vxml_play_prompts: Exit
```

The next several excerpts indicate that something is wrong with the XML form to which data is being written:

```
Aug  7 04:53:03.699: //1/71E569AF8002/VXML:/vxml_transfer_proc:
   GTD not present in <transfer>
Aug  7 04:53:03.699: //1/71E569AF8002/VXML:/vxml_transfer_proc: Exit
Aug  7 04:53:03.699: //1/71E569AF8002/VXML:/vxml_elem_proc:
   at line 8521: Status not OK, exit
Aug  7 04:53:03.703: //1/71E569AF8002/VXML:/vxml_formitem_select:
   at line 4651: Status not OK, exit
Aug  7 04:53:03.703: //1/71E569AF8002/VXML:/vxml_form_proc:
   at line 4791: Status not OK, exit
Aug  7 04:53:03.703: //1/71E569AF8002/VXML:/vxml_elem_proc:
   at line 8521: Status not OK, exit
Aug  7 04:53:03.703: //1/71E569AF8002/VXML:/vxml_vxml_proc:
   at line 8703: Status not OK, exit
Aug  7 04:53:03.703: //1/71E569AF8002/VXML:/vxml_load_immediate_done:
   sidp->status=180000000
Aug  7 04:53:03.703: //1/71E569AF8002/VAPP:/vapp_checksessionstate:
Aug  7 04:53:03.703: //1/71E569AF8002/VAPP:/vapp_checkifdone:
Aug  7 04:53:03.703: //1/71E569AF8002/VAPP:/vapp_driver: Exit
Aug  7 04:53:03.703: //1/71E569AF8002/VAPP:/vapp_evt_handler:
   State VAPP_ACTIVE got event CC_EV_CALL_MODIFY_DONE
Aug  7 04:53:03.703: //1/71E569AF8002/VAPP:/vapp_driver:
   pInterp[6383BA48]:
Aug  7 04:53:03.703: //1/71E569AF8002/VAPP:/vapp_driver:
   evtID: 37 vapp record state: 0
Aug  7 04:53:03.703: //1/71E569AF8002/VAPP:/vapp_checksessionstate:
Aug  7 04:53:03.703: //1/71E569AF8002/VAPP:/vapp_checkifdone:
Aug  7 04:53:03.703: //1/71E569AF8002/VAPP:/vapp_driver: Exit
Aug  7 04:53:03.735: //1/71E569AF8002/VAPP:/vapp_evt_handler:
   State VAPP_ACTIVE got event MSW_EV_SYNTHESIZER
Aug  7 04:53:03.735: //1/71E569AF8002/VAPP:/vapp_driver:
   pInterp[6383BA48]:
Aug  7 04:53:03.735: //1/71E569AF8002/VAPP:/vapp_driver:
   evtID: 84 vapp record state: 0
Aug  7 04:53:03.735: //1/71E569AF8002/VAPP:/vapp_play_done:
   evID=84 reason=8, protocol=2, status_code=404, dur=-1, rate=0
Aug  7 04:53:03.735: //1/71E569AF8002/VXML:/vxml_media_done:
   status 2 async_status 180000000 duration=-1 rate=0
Aug  7 04:53:03.735: //1/71E569AF8002/VXML:/vxml_save_lastprompt_info:
Aug  7 04:53:03.735: //1/71E569AF8002/VXML:/vxml_bind_lastprompt:
Aug  7 04:53:03.735: //1/71E569AF8002/VXML:/vxml_vapp_vcr_control_disable:
Aug  7 04:53:03.735: //1/71E569AF8002/VAPP:/vapp_notify_play_done:
Aug  7 04:53:03.735: //1/71E569AF8002/VAPP:/vapp_notify_play_done: Exit
```

After checking the status of the application, the router finds a badfetch error, which indicates that a VoiceXML form was not found. See the table above for a description of error.badfetch.http response codes.

```
Aug  7 04:53:03.735: //1/71E569AF8002/VXML:/vxml_media_done:
C
ALL_ERROR: : fail with vapp error 2, protocol_status_code=404
Aug  7 04:53:03.735: //1/71E569AF8002/VXML:/vxml_media_done:
CALL_ERROR: : *** error.badfetch.http.404 event is thrown
Aug  7 04:53:03.739: //1/71E569AF8002/VXML:/vxml_event_driver:
Aug  7 04:53:03.739: //1/71E569AF8002/VXML:/vxml_event_proc:
```

```
   <event>: event=error.badfetch.http.404 status=0
Aug  7 04:53:03.739: //1/71E569AF8002/VXML:/vxml_default_event_handler:
   use default event handler
Aug  7 04:53:03.739: //1/71E569AF8002/VAPP:/vapp_session_exit_event_name:
   Exit Event error.badfetch.http.404
Aug  7 04:53:03.739: //1/71E569AF8002/VXML:/vxml_load_immediate_done:
   sidp->status=10
Aug  7 04:53:03.739: //1/71E569AF8002/VXML:/vxml_start:
   vxmlhandle=6372E9BC vapphandle=6383BA48 status=0 async_status=10
Aug  7 04:53:03.739: //1/71E569AF8002/VXML:/vxml_vapp_terminate:
   vapp_status=0 ref_count 0
Aug  7 04:53:03.739: //1/71E569AF8002/VAPP:/vapp_terminate:
Aug  7 04:53:03.739: //1/71E569AF8002/VXML:/vxml_destroy: Enter
Aug  7 04:53:03.739: //-1//VXML:/vxml_gtd_delete_callback:
   New ref-count = 0
```

Related Commands

Command	Description
debug condition application voice	Displays debugging messages for only the specified VoiceXML application.
debug http client	Displays debugging messages for the HTTP client.
debug voip ivr	Displays debugging messages for VoIP IVR interactions.