

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
           55.0001.0001.1111.1111.1111.1111.1111.1111.00 - 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
```

Table 286 describes the significant fields shown in display.

Table 286 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.
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
```

[Table 287](#) describes the significant fields shown in the display.

Table 287 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.

Table 287 *debug tarp packets Field Descriptions—tarp query Command (continued)*

Field	Descriptions
TID corresponding to NET	TID (in this case cerdiwen) for the specified NSAP address.
*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.

Defaults 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.
```

[Table 288](#) describes the significant fields shown in the display.

Table 288 *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 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.

Defaults 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 network.
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 network.
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 network.
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 network.
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 network.
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 network.

```

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

<i>api</i>	(Optional) Displays a debugging message whenever the TDM subsystem application programming interface (API) is invoked from another subsystem.
<i>detail</i>	(Optional) Displays detailed messages (i.e., trace messages) whenever the TDM software executes.
<i>dynamic</i>	(Optional) Displays TDM debugging information whenever a backplane timeslot is allocated or deallocated.
<i>pri</i>	(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.
<i>test</i>	(Optional) Simulates the failure of allocating a TDM timeslot. Verifies that the software and TDM hardware recover from the failure.
<i>tsi</i>	(Optional) Displays debugging information about the TSI Chip MT8980/MT90820 driver.
<i>vdev</i>	(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 ST08, channel 1
23:16:04: TDM(reg: 0x3C500100): Connect STi4, channel 1 to ST08, channel 1
23:16:04: TDM(reg: 0x3C500100): Close connection to ST04, channel 1
23:16:04: TDM(reg: 0x3C500100): Connect STi8, channel 1 to ST04, channel 1
23:16:04: TDM(reg: 0x3C400100): Close connection to ST012, channel 31
23:16:04: TDM(reg: 0x3C400100): Close connection to ST08, 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

```

Dynamic Backplane Timeslot Pool

```

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.

Defaults 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.

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:rcv SB 24 0
*Oct 28 21:31:12.091:Telnet1/00:rcv SB 44 10 TTY1/00:Telnet COMPORT rcvd bad
suboption:0xA/0x1E
*Oct 28 21:31:12.091:Telnet1/00:rcv 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
```

Table 289 describes the significant fields shown in the display.

Table 289 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: <ul style="list-style-type: none"> • RFC 854—<i>Telnet Protocol Specification</i> • RFC 856—<i>Telnet Binary Transmission</i> • RFC 858—<i>Telnet Suppress Go Ahead Option</i> • RFC 1091—<i>Telnet Terminal-Type Option</i> • RFC 1123, sec. 3—<i>Requirements for Internet Hosts—Application and Support</i> • RFC 2217—<i>Telnet Com Port Control Option</i>

Related Commands

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

Defaults 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
```

[Table 290](#) describes the significant fields in the first line of output.

Table 290 *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.

Defaults 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 0xFFFFFFFFB4 0x0
0x0 0x4 0x58 0x6 0x7
0xFFFFFFFF98 0xFFFFFFFFFA9 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.

Defaults 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

```

[Table 291](#) describes the significant fields shown in the display.

Table 291 *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.

Command	Description
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.

Defaults 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 0xFFFFFD2 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 0xFFFFFD2
-----
```

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 0xFFFFFD2
-----
+++++
  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.

Defaults 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 0xFFFFFB4 0x0
0x0 0x4 0x58 0x6 0x7
0xFFFFFFFF98 0xFFFFFA9 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.

Command	Description
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.

Defaults 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 0xFFFFFFFFB4 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.

Defaults 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 0xFFFFF96 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 0xFFFFFD2 0x0 0x0
0x0 0x4 0x0 0x6 0x2
0x1 0x0 0x0 0x4 0xFFFFFD2
0x0 0x5 0x0 0x6 0x2
0x1 0x0 0x0 0x4 0xFFFFFD2
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 0xFFFFFD2 0x0
0x0 0x0 0x4 0x0 0x6
0x2 0x1 0x0 0x0 0x4
0xFFFFFD2 0x0 0x5 0x0 0x6
0x2 0x1 0x0 0x0 0x4
0xFFFFFD2 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 0xFFFFF96 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 0xFFFFFD2 0x0 0x0
0x0 0x4 0x0 0x6 0x2
0x1 0x0 0x0 0x4 0xFFFFFD2
0x0 0x5 0x0 0x6 0x2
0x1 0x0 0x0 0x4 0xFFFFFD2
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.

Defaults 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
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 0xFFFFFFFFB4 0x0
0x0 0x4 0x58 0x6 0x7
0xFFFFFFFF98 0xFFFFFFFFA9 0x0 0xC 0x0
0x1 0x0 0x8 0x0 0x2
0x0 0x4 0x0 0x0 0x0
0x3
```



```
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.
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.

Defaults 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
```

[Table 292](#) describes the significant fields shown in the display.

Table 292 *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.

Defaults 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: =====

```

Table 293 describes the significant fields in the display.

Table 293 debug tgrep tripr Field Descriptions

Field	Description
ev id	This field can contain the following entries: <ul style="list-style-type: none"> • 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: <ul style="list-style-type: none"> • 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

Table 293 *debug tgrep tripr* Field Descriptions (continued)

Field	Description
reason: (for a trunk group family event)	This field can contain the following entries: <ul style="list-style-type: none"> • 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: <ul style="list-style-type: none"> • 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.
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 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.

Defaults

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
```

Table 294 describes the significant fields shown in the display.

Table 294 debug tgrm all Field Descriptions

Field	Description
//-1/xxxxxxxxxxxxx/TGRM/ tgrm_accept_call:	The format of this message is //callid/GUID/module name/function name: <ul style="list-style-type: none"> • 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. • TGRM is the module name. • The tgrm_accept_call field 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.
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.

Defaults 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_EBUFFERER pppp(pl 616E9994)
i>> tiff_reader_engine() case FAX_EBUFFERER 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_EBUFFERER pppp(pl 616E9994)
i>> tiff_reader_engine() case FAX_EBUFFERER 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_EBUFFERER pppp(pl 616E9994)
i>> tiff_reader_engine() case FAX_EBUFFERER 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_EBUFFERER pppp(pl 616E9994)
i>> tiff_reader_engine() case FAX_EBUFFERER 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_EBUFFERER pppp(pl 616E9994)
i>> tiff_reader_engine() case FAX_EBUFFERER 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_EBUFFERER pppp(pl 616E9994)
i>> tiff_reader_engine() case FAX_EBUFFERER gggg

*Jan 1 18:59:13.851: tiff_reader_data_handler: buffer size: 1524i>> tiff_reader_engine()
case FAX_EBUFFERER pppp(pl 616E9994)
i>> tiff_reader_engine() case FAX_EBUFFERER 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_EBUFFERER pppp(pl 616E9994)
i>> tiff_reader_engine() case FAX_EBUFFERER 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_EBUFFERER pppp(pl 616E9994)
i>> tiff_reader_engine() case FAX_EBUFFERER 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_EBUFFERER pppp(pl 616E9994)
i>> tiff_reader_engine() case FAX_EBUFFERER 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.

Defaults 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 in privileged 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.

Defaults No default behavior or values

Command Modes Privileged EXEC

Command History	Release	Modification
	12.2(2)T	This command was introduced.

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:  riflcn 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:  riflcn 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:  riflcn 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
```

Table 295 describes the significant fields shown in the second line of output.

Table 295 *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).
MAC:	Type of packet, as follows: <ul style="list-style-type: none"> • 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.

Table 296 describes the significant fields shown in the third line of output.

Table 296 *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.

Table 297 describes the significant fields shown in the fifth line of output.

Table 297 *debug token ring Field Descriptions*

Message	Description
TR0:	Name of the interface associated with the Token Ring event.
out:	Indication of whether the packet was input to the interface (in) or output from the interface (out).
LLC:	Type of frame, as follows: <ul style="list-style-type: none"> • MAC—Media Access Control • LLC—Link Level Control
AAAA0300	This and the octets that follow it indicate the contents (hex) of the frame.
In: 28	The length of the information field (in bytes).

debug track

To display tracking activity for tracked objects, use the **debug track** command in privileged EXEC mode. To disable debugging 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 command output was enhanced to include the track-list objects.

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 is sample output from the **debug track** command. This 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 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.

Defaults

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.



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_fwdsn: 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_fwdsn: 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_fwdsn: tsn=0h, seq=Ph, for_hb=1
```

**Note**

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

Table 298 describes the significant fields shown in the display.

Table 298 *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_fwdsn	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
```

Table 299 describes the significant fields shown in the display.

Table 299 *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.
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 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.

Defaults

Debugging is not enabled for the txconn subsystem.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(5)XN	This command was introduced.

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.
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.

Defaults Debugging is not enabled for the txconn subsystem.

Command Modes Privileged EXEC

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

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: 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.

Defaults Debugging is not enabled for the txconn subsystem.

Command Modes Privileged EXEC

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

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.
	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.

Defaults Debugging is not enabled for the txconn subsystem.

Command Modes Privileged EXEC

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

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.

Defaults Debugging is not enabled for the txconn subsystem.

Command Modes Privileged EXEC

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

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.

Defaults Debugging is not enabled for the txconn subsystem.

Command Modes Privileged EXEC

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

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.
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.

Defaults

Debugging is not enabled for the txconn subsystem.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(5)XN	This command was introduced.

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.
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.

Command	Description
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.

Defaults Disabled

Command Modes Privileged EXEC

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

Examples The following is sample output from the **debug 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.
<i>transfer-method</i>	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.

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 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:11i 256 C/R 0 P/F=1
0:03:27: BR0:1: O UA:11i 256 C/R 1 P/F=1
0:03:27: BR0:1: O IFRAME:11i 256 C/R 0 N(R)=0 N(S)=0 P/F=0 len 2
0x83 0xD 0xA 0xD 0xA 0x55 0x73 0x65
0x72 0x20 0x41 0x63 0x63 0x65 0x73 0x73
0:03:27: BR0:1: I RR:11i 256 C/R 1 N(R)=1 P/F=0
0:03:28: BR0:1: I IFRAME:11i 256 C/R 0 N(R)=1 N(S)=0 P/F=0 len 2
0x83 0x63
0:03:28: BR0:1: O RR:11i 256 C/R 1 N(R)=1 P/F=0
0:03:29: BR0:1: I IFRAME:11i 256 C/R 0 N(R)=1 N(S)=1 P/F=0 len 2
0x83 0x31
0:03:29: BR0:1: O RR:11i 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:11i 256 C/R 0 N(R)=1 N(S)=2 P/F=0 len 2
0x83 0x55
0:03:32: BR0:1: I IFRAME:11i 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:11i 256 C/R 1 N(R)=3 P/F=0
0:03:32: BR0:1: I IFRAME:11i 256 C/R 0 N(R)=1 N(S)=4 P/F=0 len 2
0x83 0x73
0:03:32: BR0:1: O RR:11i 256 C/R 1 N(R)=5 P/F=0
0:03:32: BR0:1: I IFRAME:11i 256 C/R 0 N(R)=1 N(S)=5 P/F=0 len 2
0x83 0xA
0:03:32: BR0:1: O IFRAME:11i 256 C/R 0 N(R)=6 N(S)=1 P/F=0 len 9
0x83 0xD 0xA 0x68 0x65 0x66 0x65 0x72 0x3E
```

Table 300 describes the significant fields in the display.

Table 300 *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: <ul style="list-style-type: none"> • 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 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 This command 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
```

[Table 301](#) lists the messages that could be generated by this command.

Table 301 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.

Table 301 debug vg-anylan Message Descriptions (continued)

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: <ul style="list-style-type: none"> • %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-NOTHP10 0VG: 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 vcm

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

debug video vcm

no debug video vcm

Syntax Description This command has no arguments or keywords.

Defaults 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 vcm** command. Comments are enclosed in asterisks (*).

```
Router# debug video vcm

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
```