debug modem

To observe modem line activity on an access server, use the **debug modem** privileged EXEC command. The **no** form of this command disables debugging output.

debug modem

no debug modem

Syntax Description

This command has no arguments or keywords.

Examples

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

```
Router# debug modem

15:25:51: TTY4: DSR came up
15:25:51: tty4: Modem: IDLE->READY
15:25:51: TTY4: Autoselect started
15:27:51: TTY4: Autoselect failed
15:27:51: TTY4: Line reset
15:27:51: TTY4: Modem: READY->HANGUP
15:27:52: TTY4: dropping DTR, hanging up
15:27:52: tty4: Modem: HANGUP->IDLE
15:27:57: TTY4: restoring DTR
15:27:58: TTY4: DSR came up
```

The output shows when the modem line changes state.

debug modem csm

To debug the Call Switching Module (CSM), used to connect calls on the modem, use the **debug modem csm** privileged EXEC command. The **no** form of this command disables debugging output.

debug modem csm [*slot/port* | **group** *group-number*]

no debug modem csm [*slot/port* | **group** *group-number*]

Syntax Description

slot/port	(Optional) The slot and modem port number.
group *group-number*	(Optional) The modem group.

Usage Guidelines

Use the **debug modem csm** command to troubleshoot call switching problems. With this command, you can trace the complete sequence of switching incoming and outgoing calls.

Examples

The following is sample output from the **debug modem csm** command. In this example, a call enters the modem (incoming) on slot 1, port 0:

```
Router(config)# service timestamps debug uptime

Router(config)# end

Router# debug modem csm

00:04:09: ccpri_ratetoteup bear rate is 10
00:04:09: CSM_MODEM_ALLOCATE: slot 1 and port 0 is allocated.
00:04:09: MODEM_REPORT(0001): DEV_INCALL at slot 1 and port 0
00:04:09: CSM_PROC_IDLE: CSM_EVENT_ISDN_CALL at slot 1, port 0
00:04:11: CSM_RING_INDICATION_PROC: RI is on
00:04:13: CSM_RING_INDICATION_PROC: RI is off
00:04:15: CSM_PROC_IC1_RING: CSM_EVENT_MODEM_OFFHOOK at slot 1, port 0
00:04:15: MODEM_REPORT(0001): DEV_CONNECTED at slot 1 and port 0
00:04:15: CSM_PROC_IC2_WAIT_FOR_CARRIER: CSM_EVENT_ISDN_CONNECTED at slot 1, port 0
```

The following is sample output from the **debug modem csm** command when call is dialed from the modem into the network (outgoing) from slot 1, port 2:

```
Router# debug modem csm

atdt16665202
00:11:21: CSM_PROC_IDLE: CSM_EVENT_MODEM_OFFHOOK at slot 1, port 2
00:11:21: T1_MAIL_FROM_NEAT: DC_READY_RSP: mid = 1, slot = 0, unit = 0
00:11:21: CSM_PROC_OC1_REQUEST_DIGIT: CSM_EVENT_DIGIT_COLLECT_READY at slot 1, port 2
00:11:24: T1_MAIL_FROM_NEAT: DC_FIRST_DIGIT_RSP: mid = 1, slot = 0, unit = 0
00:11:24: CSM_PROC_OC2_COLLECT_1ST_DIGIT: CSM_EVENT_GET_1ST_DIGIT at slot 1, port 2
00:11:27: T1_MAIL_FROM_NEAT: DC_ALL_DIGIT_RSP: mid = 1, slot = 0, unit = 0
00:11:27: CSM_PROC_OC3_COLLECT_ALL_DIGIT: CSM_EVENT_GET_ALL_DIGITS (16665202) at slot 1,
port 2
00:11:27: ccpri_ratetoteup bear rate is 10
00:11:27: MODEM_REPORT(A000): DEV_CALL_PROC at slot 1 and port 2
00:11:27: CSM_PROC_OC4_DIALING: CSM_EVENT_ISDN_BCHAN_ASSIGNED at slot 1, port 2
00:11:31: MODEM_REPORT(A000): DEV_CONNECTED at slot 1 and port 2
00:11:31: CSM_PROC_OC5_WAIT_FOR_CARRIER: CSM_EVENT_ISDN_CONNECTED at slot 1, port 2
CONNECT 19200/REL - MNP
```

The following is sample output from the **debug modem csm** command for an incoming call:

```
Router# debug modem csm

Router#1.19.36.7 2001
Trying 1.19.36.7, 2001 ... Open
atdt111222333444555666
*Apr  7 12:39:42.475: Mica Modem(1/0): Rcvd Dial String(111222333444555666)
*Apr  7 12:39:42.475: CSM_PROC_IDLE: CSM_EVENT_MODEM_OFFHOOK at slot 1, port 0
*Apr  7 12:39:42.479: CSM_RX_CAS_EVENT_FROM_NEAT:(A001):  EVENT_CHANNEL_LOCK at slot 1 and
port 0
*Apr  7 12:39:42.479: CSM_PROC_OC4_DIALING: CSM_EVENT_DSX0_BCHAN_ASSIGNED at slot 1, port
0
*Apr  7 12:39:42.479: Mica Modem(1/0): Configure(0x1)
*Apr  7 12:39:42.479: Mica Modem(1/0): Configure(0x5)
*Apr  7 12:39:42.479: Mica Modem(1/0): Call Setup
*Apr  7 12:39:42.479: neat msg at slot 0: (1/0): Tx LOOP_CLOSURE (ABCD=1101)
*Apr  7 12:39:42.491: neat msg at slot 0: (0/0): Rx LOOP_CLOSURE (ABCD=1101)
*Apr  7 12:39:42.531: VDEV_ALLOCATE: slot 1 and port 3 is allocated.
*Apr  7 12:39:42.531: CSM_RX_CAS_EVENT_FROM_NEAT:(0004):  EVENT_CALL_DIAL_IN at slot 1 and
port 3
*Apr  7 12:39:42.531: CSM_PROC_IDLE: CSM_EVENT_DSX0_CALL at slot 1, port 3
*Apr  7 12:39:42.531: Mica Modem(1/3): Configure(0x0)
*Apr  7 12:39:42.531: Mica Modem(1/3): Configure(0x5)
*Apr  7 12:39:42.531: Mica Modem(1/3): Call Setup
*Apr  7 12:39:42.595: Mica Modem(1/0): State Transition to Call Setup
*Apr  7 12:39:42.655: Mica Modem(1/3): State Transition to Call Setup
*Apr  7 12:39:42.655: Mica Modem(1/3): Went offhook
*Apr  7 12:39:42.655: CSM_PROC_IC1_RING: CSM_EVENT_MODEM_OFFHOOK at slot 1, port 3
*Apr  7 12:39:42.671: neat msg at slot 0: (0/0): Tx LOOP_CLOSURE (ABCD=1101)
*Apr  7 12:39:42.691: neat msg at slot 0: (1/0): Rx LOOP_CLOSURE (ABCD=1101)
*Apr  7 12:39:42.731: CSM_RX_CAS_EVENT_FROM_NEAT:(A001):  EVENT_START_TX_TONE at slot 1
and port 0
*Apr  7 12:39:42.731: CSM_PROC_OC4_DIALING: CSM_EVENT_DSX0_START_TX_TONE at slot 1, port 0
*Apr  7 12:39:42.731: Mica Modem(1/0): Generate digits:called_party_num= len=1
*Apr  7 12:39:42.835: Mica Modem(1/3): Rcvd Digit detected(#)
*Apr  7 12:39:42.835: CSM_PROC_IC2_COLLECT_ADDR_INFO: CSM_EVENT_KP_DIGIT_COLLECTED (DNIS=,
ANI=) at slot 1, port 3
*Apr  7 12:39:42.855: neat msg at slot 0: (0/0): Tx LOOP_OPEN (ABCD=0101)
*Apr  7 12:39:42.871: neat msg at slot 0: (1/0): Rx LOOP_OPEN (ABCD=0101)
*Apr  7 12:39:42.899: Mica Modem(1/0): Rcvd Digits Generated
*Apr  7 12:39:42.911: CSM_RX_CAS_EVENT_FROM_NEAT:(A001):  EVENT_END_TX_TONE at slot 1 and
port 0
*Apr  7 12:39:42.911: CSM_PROC_OC4_DIALING: CSM_EVENT_DSX0_END_TX_TONE at slot 1, port 0
*Apr  7 12:39:42.911: Mica Modem(1/0): Generate digits:called_party_num=A len=1
*Apr  7 12:39:43.019: Mica Modem(1/0): Rcvd Digits Generated
*Apr  7 12:39:43.019: CSM_PROC_OC4_DIALING: CSM_EVENT_TONE_GENERATED at slot 1, port 0
*Apr  7 12:39:43.019: Mica Modem(1/3): Rcvd Digit detected(A)
*Apr  7 12:39:43.335: CSM_RX_CAS_EVENT_FROM_NEAT:(A001):  EVENT_START_TX_TONE at slot 1
and port 0
*Apr  7 12:39:43.335: CSM_PROC_OC4_DIALING: CSM_EVENT_DSX0_START_TX_TONE at slot 1, port 0
*Apr  7 12:39:43.335: Mica Modem(1/0): Generate digits:called_party_num=111222333444555666
len=19
*Apr  7 12:39:43.439: Mica Modem(1/3): Rcvd Digit detected(1)
*Apr  7 12:39:43.559: Mica Modem(1/3): Rcvd Digit detected(1)
*Apr  7 12:39:43.619: Mica Modem(1/3): Rcvd Digit detected(1)
*Apr  7 12:39:43.743: Mica Modem(1/3): Rcvd Digit detected(2)
*Apr  7 12:39:43.859: Mica Modem(1/3): Rcvd Digit detected(2)
*Apr  7 12:39:43.919: Mica Modem(1/3): Rcvd Digit detected(2)
*Apr  7 12:39:44.043: Mica Modem(1/3): Rcvd Digit detected(3)
*Apr  7 12:39:44.163: Mica Modem(1/3): Rcvd Digit detected(3)
*Apr  7 12:39:44.223: Mica Modem(1/3): Rcvd Digit detected(3)
*Apr  7 12:39:44.339: Mica Modem(1/3): Rcvd Digit detected(4)
*Apr  7 12:39:44.459: Mica Modem(1/3): Rcvd Digit detected(4)
```

```
*Apr  7 12:39:44.523: Mica Modem(1/3): Rcvd Digit detected(4)
*Apr  7 12:39:44.639: Mica Modem(1/3): Rcvd Digit detected(5)
*Apr  7 12:39:44.763: Mica Modem(1/3): Rcvd Digit detected(5)
*Apr  7 12:39:44.883: Mica Modem(1/3): Rcvd Digit detected(5)
*Apr  7 12:39:44.943: Mica Modem(1/3): Rcvd Digit detected(6)
*Apr  7 12:39:45.063: Mica Modem(1/3): Rcvd Digit detected(6)
*Apr  7 12:39:45.183: Mica Modem(1/3): Rcvd Digit detected(6)
*Apr  7 12:39:45.243: Mica Modem(1/3): Rcvd Digit detected(B)
*Apr  7 12:39:45.243: CSM_PROC_IC2_COLLECT_ADDR_INFO: CSM_EVENT_DNIS_COLLECTED
(DNIS=111222333444555666, ANI=) at slot 1, port 3
*Apr  7 12:39:45.363: Mica Modem(1/0): Rcvd Digits Generated
*Apr  7 12:39:45.891: neat msg at slot 0: (0/0): Tx LOOP_CLOSURE (ABCD=1101)
*Apr  7 12:39:45.907: neat msg at slot 0: (1/0): Rx LOOP_CLOSURE (ABCD=1101)
*Apr  7 12:39:46.115: neat msg at slot 0: (0/0): Tx LOOP_OPEN (ABCD=0101)
*Apr  7 12:39:46.131: neat msg at slot 0: (1/0): Rx LOOP_OPEN (ABCD=0101)
*Apr  7 12:39:46.175: CSM_RX_CAS_EVENT_FROM_NEAT:(A001):  EVENT_START_TX_TONE at slot 1
and port 0
*Apr  7 12:39:46.175: CSM_PROC_OC4_DIALING: CSM_EVENT_DSX0_START_TX_TONE at slot 1, port 0
*Apr  7 12:39:46.175: Mica Modem(1/0): Generate digits:called_party_num= len=3
*Apr  7 12:39:46.267: Mica Modem(1/3): Rcvd Digit detected(#)
*Apr  7 12:39:46.387: Mica Modem(1/3): Rcvd Digit detected(A)
*Apr  7 12:39:46.447: Mica Modem(1/3): Rcvd Digit detected(B)
*Apr  7 12:39:46.447: CSM_PROC_IC2_COLLECT_ADDR_INFO: CSM_EVENT_ADDR_INFO_COLLECTED
(DNIS=111222333444555666, ANI=) at slot 1, port 3
*Apr  7 12:39:46.507: Mica Modem(1/0): Rcvd Digits Generated
*Apr  7 12:39:46.507: CSM_PROC_OC4_DIALING: CSM_EVENT_ADDR_INFO_COLLECTED at slot 1, port
0
*Apr  7 12:39:47.127: CSM_RX_CAS_EVENT_FROM_NEAT:(0004):  EVENT_CHANNEL_CONNECTED at slot
1 and port 3
*Apr  7 12:39:47.127: CSM_PROC_IC4_WAIT_FOR_CARRIER: CSM_EVENT_DSX0_CONNECTED at slot 1,
port 3
*Apr  7 12:39:47.127: Mica Modem(1/3): Link Initiate
*Apr  7 12:39:47.131: neat msg at slot 0: (0/0): Tx LOOP_CLOSURE (ABCD=1101)
*Apr  7 12:39:47.147: neat msg at slot 0: (1/0): Rx LOOP_CLOSURE (ABCD=1101)
*Apr  7 12:39:47.191: CSM_RX_CAS_EVENT_FROM_NEAT:(A001):  EVENT_CHANNEL_CONNECTED at slot
1 and port 0
*Apr  7 12:39:47.191: CSM_PROC_OC5_WAIT_FOR_CARRIER: CSM_EVENT_DSX0_CONNECTED at slot 1,
port 0
*Apr  7 12:39:47.191: Mica Modem(1/0): Link Initiate
*Apr  7 12:39:47.227: Mica Modem(1/3): State Transition to Connect
*Apr  7 12:39:47.287: Mica Modem(1/0): State Transition to Connect
*Apr  7 12:39:49.103: Mica Modem(1/0): State Transition to Link
*Apr  7 12:39:52.103: Mica Modem(1/3): State Transition to Link
*Apr  7 12:40:00.927: Mica Modem(1/3): State Transition to Trainup
*Apr  7 12:40:00.991: Mica Modem(1/0): State Transition to Trainup
*Apr  7 12:40:02.615: Mica Modem(1/0): State Transition to EC Negotiating
*Apr  7 12:40:02.615: Mica Modem(1/3): State Transition to EC Negotiating
CONNECT 31200 /V.42/V.42bis
Router>
*Apr  7 12:40:05.983: Mica Modem(1/0): State Transition to Steady State
*Apr  7 12:40:05.983: Mica Modem(1/3): State Transition to Steady State+++
OK
ath
*Apr  7 12:40:09.167: Mica Modem(1/0): State Transition to Steady State Escape
*Apr  7 12:40:10.795: Mica Modem(1/0): State Transition to Terminating
*Apr  7 12:40:10.795: Mica Modem(1/3): State Transition to Terminating
*Apr  7 12:40:11.755: Mica Modem(1/3): State Transition to Idle
*Apr  7 12:40:11.755: Mica Modem(1/3): Went onhook
*Apr  7 12:40:11.755: CSM_PROC_IC5_OC6_CONNECTED: CSM_EVENT_MODEM_ONHOOK at slot 1, port 3
*Apr  7 12:40:11.755: VDEV_DEALLOCATE: slot 1 and port 3 is deallocated
*Apr  7 12:40:11.759: neat msg at slot 0: (0/0): Tx LOOP_OPEN (ABCD=0101)
*Apr 7 12:40:11.767: neat msg at slot 0: (1/0): Rx LOOP_OPEN (ABCD=0101)
*Apr  7 12:40:12.087: neat msg at slot 0: (1/0): Tx LOOP_OPEN (ABCD=0101)
*Apr  7 12:40:12.091: neat msg at slot 0: (0/0): Rx LOOP_OPEN (ABCD=0101)
```

```
*Apr  7 12:40:12.111: CSM_RX_CAS_EVENT_FROM_NEAT:(A001):  EVENT_CALL_IDLE at slot 1 and
port 0
*Apr  7 12:40:12.111: CSM_PROC_IC5_OC6_CONNECTED: CSM_EVENT_DSX0_DISCONNECTED at slot 1,
port 0
*Apr  7 12:40:12.111: Mica Modem(1/0): Link Terminate(0x6)
*Apr  7 12:40:12.779: Mica Modem(1/3): State Transition to Terminating
*Apr  7 12:40:12.839: Mica Modem(1/3): State Transition to Idle
*Apr  7 12:40:13.495: Mica Modem(1/0): State Transition to Idle
*Apr  7 12:40:13.495: Mica Modem(1/0): Went onhook
*Apr  7 12:40:13.495: CSM_PROC_IC6_OC8_DISCONNECTING: CSM_EVENT_MODEM_ONHOOK at slot 1,
port 0
*Apr  7 12:40:13.495: VDEV_DEALLOCATE: slot 1 and port 0 is deallocated
Router#disc
Closing connection to 1.19.36.7 [confirm]
Router#
*Apr  7 12:40:18.783: Mica Modem(1/0): State Transition to Terminating
*Apr  7 12:40:18.843: Mica Modem(1/0): State Transition to Idle
Router#
```

The MICA technologies modem goes through the following internal link states when the call comes in:

- Call Setup
- Off Hook
- Connect
- Link
- Trainup
- EC Negotiation
- Steady State

The following section describes the CSM activity for an incoming call.

When a voice call comes in, CSM is informed of the incoming call. This allocates the modem and sends the Call Setup message to the MICA modem. The Call_Proc message is sent through D channel. The modem sends an offhook message to CSM by sending the state change to Call Setup. The D channel then sends a CONNECT message. When the CONNECT_ACK message is received, the Link initiate message is sent to the MICA modem and it negotiates the connection with the remote modem. In the following debug examples, a modem on slot 1, port 13 is allocated. It goes through its internal states before it is in Steady State and answers the call.

```
Router# debug modem csm

Modem Management Call Switching Module debugging is on
*May 13 15:01:00.609: MODEM_REPORT:dchan_idb=0x60D437F8, call_id=0xE, ces=0x1
   bchan=0x12, event=0x1, cause=0x0
*May 13 15:01:00.609: VDEV_ALLOCATE: slot 1 and port 13 is allocated.
*May 13 15:01:00.609: MODEM_REPORT(000E): DEV_INCALL at slot 1 and port 13
*May 13 15:01:00.609: CSM_PROC_IDLE: CSM_EVENT_ISDN_CALL at slot 1, port 13
*May 13 15:01:00.609: Mica Modem(1/13): Configure(0x0)
*May 13 15:01:00.609: Mica Modem(1/13): Configure(0x0)
*May 13 15:01:00.609: Mica Modem(1/13): Configure(0x6)
*May 13 15:01:00.609: Mica Modem(1/13): Call Setup
*May 13 15:01:00.661: Mica Modem(1/13): State Transition to Call Setup
*May 13 15:01:00.661: Mica Modem(1/13): Went offhook
*May 13 15:01:00.661: CSM_PROC_IC1_RING: CSM_EVENT_MODEM_OFFHOOK at slot 1, port 13
*May 13 15:01:00.661: MODEM_REPORT:dchan_idb=0x60D437F8, call_id=0xE, ces=0x1
   bchan=0x12, event=0x4, cause=0x0
*May 13 15:01:00.661: MODEM_REPORT(000E): DEV_CONNECTED at slot 1 and port 13
*May 13 15:01:00.665: CSM_PROC_IC3_WAIT_FOR_CARRIER:
CSM_EVENT_ISDN_CONNECTED at slot 1, port 13
```

```
*May 13 15:01:00.665: Mica Modem(1/13): Link Initiate
*May 13 15:01:00.693: Mica Modem(1/13): State Transition to Connect
*May 13 15:01:01.109: Mica Modem(1/13): State Transition to Link
*May 13 15:01:09.433: Mica Modem(1/13): State Transition to Trainup
*May 13 15:01:11.541: Mica Modem(1/13): State Transition to EC Negotiating
*May 13 15:01:12.501: Mica Modem(1/13): State Transition to Steady State
```

The following section describes the status of CSM when a call is connected.

The **show modem csm x/y** command is similar to AS5200 access server. For an active incoming analog call, the modem_status and csm_status should be VDEV_STATUS_ACTIVE_CALL and CSM_IC4_CONNECTED, respectively.

```
Router# show modem csm 1/13

MODEM_INFO: slot 1, port 13, unit 0, modem_mask=0x0000, modem_port_offset=0
tty_hwidb=0x60D0BCE0, modem_tty=0x60B6FE7C, oobp_info=0x00000000,
modem_pool=0x60ADC998
modem_status(0x0002):VDEV_STATUS_ACTIVE_CALL.
csm_state(0x0204)=CSM_IC4_CONNECTED, csm_event_proc=0x600C6968, current
call thru PRI line
invalid_event_count=0, wdt_timeout_count=0
wdt_timestamp_started is not activated
wait_for_dialing:False, wait_for_bchan:False
pri_chnl=TDM_PRI_STREAM(s0, u0, c18), modem_chnl=TDM_MODEM_STREAM(s1, c13)
dchan_idb_start_index=0, dchan_idb_index=0, call_id=0x000E, bchan_num=18
csm_event=CSM_EVENT_ISDN_CONNECTED, cause=0x0000
ring_indicator=0, oh_state=0, oh_int_enable=0, modem_reset_reg=0
ring_no_answer=0, ic_failure=0, ic_complete=1
dial_failure=0, oc_failure=0, oc_complete=0
oc_busy=0, oc_no_dial_tone=0, oc_dial_timeout=0
remote_link_disc=0, stat_busyout=0, stat_modem_reset=0
oobp_failure=0
call_duration_started=1d02h, call_duration_ended=00:00:00,
total_call_duration=00:00:00
The calling party phone number = 4085552400
The called party phone number  = 4085551400
total_free_rbs_timeslot = 0, total_busy_rbs_timeslot = 0,
total_dynamic_busy_rbs_timeslot = 0, total_static_busy_rbs_timeslot = 0,
min_free_modem_threshold = 6
```

The following section describes the CSM activity for an outgoing call.

For MICA modems, the dial tone is not required to initiate an outbound call. Unlike in the AS5200, the digit collection step is not required. The dialed digit string is sent to the CSM in the outgoing request to the CSM. CSM signals the D channel to generate an outbound voice call, and the B channel assigned is connected to the modem and the CSM.

The modem is ordered to connect to the remote side with a CONNECT message, and by sending a link initiate message, the modem starts to train.

```
Router# debug modem csm

Modem Management Call Switching Module debugging is on
Router# debug isdn q931
ISDN Q931 packets debugging is on
*May 15 12:48:42.377: Mica Modem(1/0): Rcvd Dial String(5552400)
*May 15 12:48:42.377: CSM_PROC_IDLE: CSM_EVENT_MODEM_OFFHOOK at slot 1, port 0
*May 15 12:48:42.377: CSM_PROC_OC3_COLLECT_ALL_DIGIT:
CSM_EVENT_GET_ALL_DIGITS at slot 1, port 0
*May 15 12:48:42.377: CSM_PROC_OC3_COLLECT_ALL_DIGIT: called party num:
(5552400) at slot 1, port 0
*May 15 12:48:42.381:  process_pri_call making a voice_call.
*May 15 12:48:42.381: ISDN Se0:23: TX ->  SETUP pd = 8  callref = 0x0011
```

```
*May 15 12:48:42.381:         Bearer Capability i = 0x8090A2
*May 15 12:48:42.381:         Channel ID i = 0xE1808397
*May 15 12:48:42.381:         Called Party Number i = 0xA1, '5552400'
*May 15 12:48:42.429: ISDN Se0:23: RX <-  CALL_PROC pd = 8  callref = 0x8011
*May 15 12:48:42.429:         Channel ID i = 0xA98397
*May 15 12:48:42.429: MODEM_REPORT:dchan_idb=0x60D437F8, call_id=0xA011, ces=0x1
   bchan=0x16, event=0x3, cause=0x0
*May 15 12:48:42.429: MODEM_REPORT(A011): DEV_CALL_PROC at slot 1 and port 0
*May 15 12:48:42.429: CSM_PROC_OC4_DIALING: CSM_EVENT_ISDN_BCHAN_ASSIGNED
at slot 1, port 0
*May 15 12:48:42.429: Mica Modem(1/0): Configure(0x1)
*May 15 12:48:42.429: Mica Modem(1/0): Configure(0x0)
*May 15 12:48:42.429: Mica Modem(1/0): Configure(0x6)
*May 15 12:48:42.429: Mica Modem(1/0): Call Setup
*May 15 12:48:42.489: Mica Modem(1/0): State Transition to Call Setup
*May 15 12:48:42.589: ISDN Se0:23: RX <-  ALERTING pd = 8  callref = 0x8011
*May 15 12:48:43.337: ISDN Se0:23: RX <-  CONNECT pd = 8  callref = 0x8011
*May 15 12:48:43.341: MODEM_REPORT:dchan_idb=0x60D437F8, call_id=0xA011, ces=0x1
   bchan=0x16, event=0x4, cause=0x0
*May 15 12:48:43.341: MODEM_REPORT(A011): DEV_CONNECTED at slot 1 and port 0
*May 15 12:48:43.341: CSM_PROC_OC5_WAIT_FOR_CARRIER:
CSM_EVENT_ISDN_CONNECTED at slot 1, port 0
*May 15 12:48:43.341: Mica Modem(1/0): Link Initiate
*May 15 12:48:43.341: ISDN Se0:23: TX ->  CONNECT_ACK pd = 8  callref = 0x0011
*May 15 12:48:43.385: Mica Modem(1/0): State Transition to Connect
*May 15 12:48:43.849: Mica Modem(1/0): State Transition to Link
*May 15 12:48:52.665: Mica Modem(1/0): State Transition to Trainup
*May 15 12:48:54.661: Mica Modem(1/0): State Transition to EC Negotiating
*May 15 12:48:54.917: Mica Modem(1/0): State Transition to Steady State
```

Related Commands

Command	Description
debug modem oob	Creates modem startup messages between the network management software and the modem on the specificed OOB port.
debug modem trace	Performs a call trace on the specified modem, which allows you to determine why calls are terminated.

debug modem dsip

To display output for modem control messages that are received or sent to the router, use the **debug modem dsip** privileged EXEC command. To disable the output, use the **no** form of this command.

debug modem dsip {*tty-range* | **group** | *shelf/slot/port*}

no debug modem dsip {*tty-range* | **group** | *shelf/slot/port*}

Syntax Description

tty-range	Modem tty number or range. You can specify a single TTY line number or a range from 0 through the number of modems you have in your Cisco AS5800 access server. Be sure to include a dash (-) between the range values you specify.
group	Modem group information.
shelf/slot/port	Location of the modem by shelf/slot/port numbers for internal modems.

Command History

Release	Modification
11.3(2)AA	This command was introduced.

Usage Guidelines

The **debug modem dsip** command displays each DSIP message that relates to a modem and is sent from or received at the router shelf. This command can be applied to a single modem or a group of modems.

Examples

The following examples show a display of the available **debug modem** command options and **debug modem dsip** command options:

```
Router# debug modem ?

  dsip         Modem DSIP activity
  maintenance  Modem maintenance activity
  oob          Modem out of band activity
  trace        Call Trace Upload
  traffic      Modem data traffic
  <cr>

Router# debug modem dsip ?

  <0-935>  First Modem TTY Number
  group    Modem group information
  x/y/z    Shelf/Slot/Port for Internal Modems
  <cr>
```

The following example indicates that an RTS status message was received from the router shelf, and an ACK message was sent back:

```
Router# debug modem dsip

00:11:02: RSMODEM_SEND-1/2/06: MODEM_RING_INDICATION_MSG cci1 si0 ms0 mm65535,0 dc0
00:11:02: RSMODEM_sRCV-1/2/06:112,MODEM_CALL_ACK_MSG:
00:11:02: RSMODEM_SEND-1/2/06: MODEM_CALL_ACCEPT_MSG
00:11:11: RSMODEM_sRCV-2:10,MODEM_POLL_MSG: 0 16 0 7 0 146 0 36 21
00:11:18: RSMODEM_sRCV-1/2/06:112,MODEM_SET_DCD_STATE_MSG: 1
```

```
00:11:19: RSMODEM_SEND-1/2/06: MODEM_RTS_STATUS_MSG 1
00:11:19: RSMODEM_dRCV-2:11258607996,MODEM_RTS_STATUS_MSG: 0 6 0 23 0 0 0 0 0
00:11:23: RSMODEM_sRCV-2:10,MODEM_POLL_MSG: 0 16 0 7 0 146 0 150 21
00:12:31: RSMODEM_sRCV-1/2/06:112,MODEM_SET_DCD_STATE_MSG: 0
00:12:31: RSMODEM_SEND-1/2/06: MODEM_CALL_HANGUP_MSG
00:12:31: RSMODEM_sRCV-1/2/06:112,MODEM_ONHOOK_MSG:
00:12:32: RSMODEM_SEND-1/2/06: MODEM_RTS_STATUS_MSG 1
00:12:32: RSMODEM_SEND-1/2/06: MODEM_SET_DTR_STATE_MSG 0
00:12:32: RSMODEM_dRCV-2:11258659676,MODEM_RTS_STATUS_MSG: 0 6 0 16 0 0 0 0 0
00:12:32: RSMODEM_SEND-1/2/06: MODEM_RTS_STATUS_MSG 1
00:12:32: RSMODEM_dRCV-2:11258600700,MODEM_RTS_STATUS_MSG: 0 6 0 13 0 0 0 0 0
00:12:33: RSMODEM_SEND-1/2/06: MODEM_SET_DTR_STATE_MSG 0
00:12:33: RSMODEM_SEND-1/2/06: MODEM_RTS_STATUS_MSG 1
00:12:33: RSMODEM_dRCV-2:11258662108,MODEM_RTS_STATUS_MSG: 0 6 0 16 0 0 0 0 0
00:12:35: RSMODEM_sRCV-2:10,MODEM_POLL_MSG: 0 16 0 7 0 146 1 34 22
00:12:38: RSMODEM_SEND-1/2/06: MODEM_SET_DTR_STATE_MSG 1
00:12:47: RSMODEM_sRCV-2:10,MODEM_POLL_MSG: 0 16 0 7 0 146 0 12 22
```

Table 123 describes the significant fields shown in the display.

Table 123 debug modem dsip Field Descriptions

Field	Description
RSMODEM_SEND-1/2/06	Router shelf modem shelf sends a MODEM_RING_INDICATION_MSG message.
RSMODEM_sRCV-1/2/06	Router shelf modem received a MODEM_CALL_ACK_MSG message.
MODEM_CALL_ACCEPT_MSG	Router shelf accepts the call.
MODEM_CALL_HANGUP_MSG	Router shelf sends a hangup message.
MODEM_RTS_STATUS_MSG	Request to send message status.

Related Commands

Command	Description
debug modem traffic	Displays output for framed, unframed, and asynchronous data transmission received from the modem cards.
debug dsip	Displays output for DSIP used between the router shelf and the dial shelf.

debug modem oob

To debug the out-of-band port used to poll modem events on the modem, use the **debug modem oob** privileged EXEC command. The **no** form of this command disables debugging output.

debug modem oob [*slot/modem-port* | **group** *group-number*]

no debug modem oob [*slot/modem-port* | **group** *group-number*]

Syntax Description

slot/modem-port	(Optional) The slot and modem port number.
group *group-number*	(Optional) The modem group.

Usage Guidelines

The message types and sequence numbers that appear in the debug output are initiated by the Modem Out-of-Band Protocol and used by service personnel for debugging purposes.

Caution Entering the **debug modem oob** command without specifying a slot and modem number debugs *all* out-of-band ports, which generates a substantial amount of information.

Examples

The following is sample output from the **debug modem oob** command. This example debugs the out-of-band port on modem 2/0, which creates modem startup messages between the network management software and the modem.

```
Router# debug modem oob 2/0

MODEM(2/0): One message sent --Message type:3, Sequence number:0
MODEM(2/0): Modem DC session data reply
MODEM(2/0): One message sent --Message type:83, Sequence number:1
MODEM(2/0): DC session event =
MODEM(2/0): One message sent --Message type:82, Sequence number:2
MODEM(2/0): No status changes since last polled
MODEM(2/0): One message sent --Message type:3, Sequence number:3
MODEM(2/0): Modem DC session data reply
MODEM(2/0): One message sent --Message type:83, Sequence number:4
```

Related Commands

Command	Description
debug modem csm	Debugs the CSM used to connect calls on the modem.
debug modem trace	Performs a call trace on the specified modem, which allows you to determine why calls are terminated.

debug modem trace

To debug a call trace on the modem to determine why calls are terminated, use the **debug modem trace** privileged EXEC command. The **no** form of this command disables debugging output.

debug modem trace [**normal** | **abnormal** | **all**] [*slot/modem-port* | **group** *group-number*]

no debug modem trace [**normal** | **abnormal** | **all**] [*slot/modem-port* | **group** *group-number*]

Syntax Description

normal	(Optional) Uploads the call trace to the syslog server on normal call termination (for example, a local user hangup or a remote user hangup).
abnormal	(Optional) Uploads the call trace to the syslog server on abnormal call termination (for example, any call termination other than normal termination, such as a lost carrier or a watchdog timeout).
all	(Optional) Uploads the call trace on all call terminations including normal and abnormal call termination.
slot/modem-port	(Optional) The slot and modem port number.
group *group-number*	(Optional) The modem group.

Usage Guidelines

The **debug modem trace** command applies only to manageable modems. For additional information, use the **show modem** command.

Examples

The following is sample output from the **debug modem trace abnormal** command:

```
Router# debug modem trace abnormal 1/14

Modem 1/14 Abnormal End of Connection Trace. Caller 123-4567
    Start-up Response: AS5200 Modem, Firmware 1.0
    Control Reply: 0x7C01
    DC session response: brasil firmware 1.0
    RS232 event:
    DSR=On, DCD=On, RI=Off, TST=Off
    changes: RTS=No change, DTR=No change, CTS=No change
    changes: DSR=No change, DCD=No change, RI=No change, TST=No change
    Modem State event: Connected
    Connection event: Speed = 19200, Modulation = VFC
    Direction = Originate, Protocol = reliable/LAPM, Compression = V42bis
    DTR event: DTR On
    Modem Activity event: Data Active
    Modem Analog signal event: TX = -10, RX = -24, Signal to noise = -32
    End connection event: Duration = 10:34-11:43,
    Number of xmit char =    67, Number of rcvd char = 88, Reason: Watchdog Time-out.
```

Related Commands

Command	Description
debug modem csm	Debugs the CSM used to connect calls on the modem.
debug modem oob	Creates modem startup messages between the network management software and the modem on the specificed OOB port.

debug modem traffic

To display output for framed, unframed, and asynchronous data sent received from the modem cards, use the **debug modem traffic** privileged EXEC command. To disable output, use the **no** form of this command.

debug modem traffic

no debug modem traffic

Syntax Description

This command has no arguments or keywords.

Command History

Release	Modification
11.3(2)AA	This command was introduced.

Usage Guidelines

The **debug modem traffic** command displays output for framed, unframed, and asynchronous data sent or received by the modem cards.

Examples

The following example displays information about unframed or framed data sent to or received from the modem cards:

```
Router# debug modem traffic

MODEM-RAW-TX:modem = 6/5/00, length = 1, data = 0x61, 0xFF, 0x7D, 0x23
MODEM-RAW-RX:modem = 6/5/00, length = 1, data = 0x61, 0x0, 0x0, 0x0
```

The information indicates unframed asynchronous data transmission and reception involving the modem on shelf 6, slot 5, port 00.

The following example displays framed asynchronous data transmission and reception involving the modem on shelf 6, slot 5, port 00:

```
Router# debug modem traffic

MODEM-FRAMED-TX:modem = 6/5/00, length = 8, data = 0xFF, 0x3, 0x82
MODEM-FRAMED-RX:modem = 6/5/00, length = 14, data = 0xFF, 0x3, 0x80
```

Related Commands

Command	Description
debug modem dsip	Displays output for modem control messages that are received or sent to the router.

debug mpls adjacency

To display changes to label switching entries in the adjacency database, use the **debug mpls adjacency** EXEC command. The **no** form of this command disables debugging output.

debug mpls adjacency

no debug mpls adjacency

Usage Guidelines

This command has no keywords or arguments.

Defaults

This command has no default behavior or values.

Command Modes

Privileged EXEC

Command History

Release	Modification
11.1CT	This command was introduced.
12.1(3)T	This command was modified to reflect new MPLS IETF terminology and CLI command syntax.

Usage Guidelines

Use the **debug mpls adjacency** command to monitor when entries are updated in or added to the adjacency database.

Examples

The following is sample output generated by the **debug mpls adjacency** command:

```
Router# debug mpls adjacency

TAG ADJ: add 10.10.0.1, Ethernet0/0/0
TAG ADJ: update 10.10.0.1, Ethernet0/0/0
```

Table 124 describes the significant fields shown in the sample display above.

Table 124 debug mpls adjacency Command Field Description

Field	Description
add	Adding an entry to the database.
update	Updating the MAC address for an existing entry.
10.10.0.1	Address of neighbor TSR.
Ethernet0/0/0	Connecting interface.

debug mpls ldp backoff

To display information about the label distribution protocol (LDP) backoff mechanism parameters, use the **debug mpls ldp backoff** command in privileged EXEC mode. To disable this feature, use the **no** form of this command.

debug mpls ldp backoff

no debug mpls ldp backoff

Syntax Description

This command has no arguments or keywords.

Defaults

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(10)ST	This command was introduced.
12.1(2)T	This command was integrated into Cisco IOS Release 12.1(2)T.
12.1(8a)E	This command was integrated into Cisco IOS Release 12.1(8a)E.
12.0(22)S	This command was integrated into Cisco IOS Release 12.0(22)S.
12.2(14)S	This command was integrated into Cisco IOS Release 12.2(14)S.

Usage Guidelines

Use this command to monitor backoff parameters configured for LDP sessions.

Examples

The following shows sample output from the **debug mpls ldp backoff** command:

```
Router# debug mpls ldp backoff

LDP session establishment backoff debugging is on

Router#

Jan 6 22:31:13.012: ldp: Backoff peer ok: 12.12.12.12:0; backing off; threshold/count 8/6
Jan 6 22:31:13.824: ldp: Backoff peer ok: 12.12.12.12:1; backing off; threshold/count 8/6
Jan 6 22:31:17.848: ldp: Backoff peer ok: 12.12.12.12:0; backing off; threshold/count 8/6
Jan 6 22:31:18.220: ldp: Backoff peer ok: 12.12.12.12:1; backing off; threshold/count 8/6
Jan 6 22:31:21.908: ldp: Backoff peer ok: 12.12.12.12:0; backing off; threshold/count 8/6
Jan 6 22:31:22.980: ldp: Backoff peer ok: 12.12.12.12:1; backing off; threshold/count 8/6
Jan 6 22:31:25.724: ldp: Backoff peer ok: 12.12.12.12:0; backing off; threshold/count 8/7
Jan 6 22:31:26.944: ldp: Backoff peer ok: 12.12.12.12:1; backing off; threshold/count 8/7
Jan 6 22:31:30.140: ldp: Backoff peer ok: 12.12.12.12:0; backing off; threshold/count 8/7
Jan 6 22:31:31.932: ldp: Backoff peer ok: 12.12.12.12:1; backing off; threshold/count 8/7
Jan 6 22:31:35.028: ldp: Backoff peer ok: 12.12.12.12:0; backing off; threshold/count 8/7
Jan 6 22:31:35.788: ldp: Backoff peer ok: 12.12.12.12:1; backing off; threshold/count 8/7
```

```
Jan 6 22:31:39.332: ldp: Update backoff rec: 12.12.12.12:0, threshold = 8, tbl ents 2
Jan 6 22:31:39.640: ldp: Update backoff rec: 12.12.12.12:1, threshold = 8, tbl ents 2
```

Table 125 describes the significant fields shown in the display.

Table 125 debug mpls ldp backoff Field Descriptions

Field	Description
ldp	Identifies the Label Distribution Protocol.
Backoff peer ok: a.b.c.d:n	Identifies the LDP peer for which a session is being delayed because of a failure to establish a session due to incompatible configuration.
backing off;	Indicates that a session setup attempt failed and the LSR is delaying its next attempt (that is, is backing off).
threshold/count x/y	Identifies a set threshold (x) and a count (y) that represents the time that has passed since the last attempt to set up a session with the peer. The count is incremented every 15 seconds until it reaches the threshold. When the count equals the threshold, a fresh attempt is made to set up an LDP session with the peer.
Update backoff rec	Indicates that the backoff period is over and that it is time for another attempt to set up an LDP session.
threshold = x	Indicates the backoff time of x*15 seconds, for the next LDP session attempt with the peer.
tbl ents 2	Indicates unsuccessful attempts to set up an LDP session with two different LDP peers. In this example, attempts to set up sessions with LDP peers 12.12.12.12:0 and 12.12.12.12:1 are failing.

Related Commands

Command	Description
mpls ldp backoff	Configures session setup delay parameters for the LDP backoff mechanism.
show mpls ldp backoff	Displays information about the configured session setup backoff parameters and any potential LDP peers with which session setup attempts are being throttled.

debug mpls events

To display information about significant MPLS events, use the **debug mpls events** privileged EXEC command. Use the **no** form of this command to disable this feature.

debug mpls events

no debug mpls events

Syntax Description

This command has no keywords or arguments.

Defaults

This command has no default behavior or values.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.1(3)T	This command was introduced.

Usage Guidelines

Use this command to monitor significant MPLS events. For this Cisco IOS release, the only events reported by this command are changes to the MPLS router ID.

Examples

The following is sample output from the **debug mpls events** command:

```
Router# debug mpls events

MPLS events debugging is on

TAGSW: Unbound IP address, 155.0.0.55, from Router ID
TAGSW: Bound IP address, 199.44.44.55, to Router ID
```

debug mpls lfib cef

To print detailed information about label rewrites being created, resolved, and deactivated as CEF routes are added, changed, or removed, use the **debug mpls lfib cef** EXEC command. The **no** form of this command disables debugging.

debug mpls lfib cef

no debug mpls lfib cef

Syntax Description

This command has no keywords or arguments.

Defaults

This command has no default behavior or values.

Command Modes

Privileged EXEC

Command History

Release	Modification
11.1CT	This command was introduced.
12.1(3)T	This command was modified to reflect new MPLS IETF terminology and CLI command syntax.

Usage Guidelines

Several lines of output are produced for each route placed into the LFIB. If your router has thousands of labeled routes, be careful about issuing this command. When label switching is first enabled, each of these routes is placed into the LFIB, and several lines of output are displayed for each route.

Examples

The following is sample output displayed when you enter the **debug mpls lfib cef** command:

```
Router# debug mpls lfib cef

Cisco Express Forwarding related TFIB services debugging is on

tagcon: tc_ip_rtlookup fail on 10.0.0.0/8:subnet_lookup failed
TFIB: route tag chg 10.7.0.7/32,idx=1,inc=Withdrn,outg=Withdrn,enabled=0x2
TFIB: fib complete delete: prefix=10.7.0.7/32,inc tag=26,delete_info=1
TFIB: deactivate tag rew for 10.7.0.7/32,index=0
TFIB: set fib rew: pfx 10.7.0.7/32,index=0,add=0,tag_rew->adj=Ethernet2/3
TFIB: resolve tag rew,prefix=10.7.0.7/32,no tag_info,no parent
TFIB: fib scanner start:needed:1,unres:0,mac:0,loadinfo:0
TFIB: resolve tag rew,prefix=10.7.0.7/32,no tag_info,no parent
TFIB: fib upd loadinf 10.100.100.100/32,tag=Tun_hd,fib no loadin,tfib no loadin
TFIB: fib check cleanup for 10.100.100.100/32,index=0,return_value=0
TFIB: fib_scanner_end
TFIB: create dynamic entry for 10.11.0.11/32
TFIB: call find_route_tags,dist_method=1,next_hop=10.93.0.11,Et2/3
TFIB: route tag chg 10.11.0.11/32,idx=0,inc=26,outg=Unkn,enabled=0x3
TFIB: create tag info 10.11.0.11/32,inc tag=26,has no info
TFIB: resolve tag rew,prefix=10.11.0.11/32,has tag_info,no parent
```

```
TFIB: finish fib res 10.11.0.11/32:index 0,parent outg tag no parent
TFIB: fib upd loadinf 10.11.0.11/32,tag=26,fib no loadin,tfib no loadin
TFIB: set fib rew: pfx 10.11.0.11/32,index=0,add=1,tag_rew->adj=Ethernet2/3
tagcon: route_tag_change for: 10.250.0.97/32
        intag 33, outtag 28, nexthop tsr 10.11.0.11:0
TFIB: route tag chg 10.250.0.97/32,idx=0,inc=33,outg=28,enabled=0x3
TFIB: deactivate tag rew for 10.250.0.97/32,index=0
TFIB: set fib rew: pfx 10.250.0.97/32,index=0,add=0,tag_rew->adj=Ethernet2/3
TFIB: create tag info 10.250.0.97/32,inc tag=33,has old info
On VIP:
TFIB: route tag chg 10.13.72.13/32,idx=0,inc=34,outg=Withdrn,enabled=0x3
TFIB: deactivate tag rew for 10.13.72.13/32,index=0
TFIB: set fib rew: pfx 10.13.72.13/32,index=0,add=0,tag_rew->adj=
TFIB: create tag info 10.13.72.13/32,inc tag=34,has old info
TFIB: resolve tag rew,prefix=10.13.72.13/32,has tag_info,no parent
TFIB: finish fib res 10.13.72.13/32:index 0,parent outg tag no parent
TFIB: set fib rew: pfx 10.100.100.100/32,index=0,add=0,tag_rew->adj=
TFIB: create tag info 10.100.100.100/32,inc tag=37,has old info
TFIB: resolve tag rew,prefix=10.100.100.100/32,has tag_info,no parent
TFIB: finish fib res 10.100.100.100/32:index 0,parent outg tag no parent
TFIB: fib upd loadinf 10.100.100.100/32,tag=37,fib no loadin,tfib no loadin
```

Table 126 lists the significant fields shown in the display.

See Table 128 for a description of special labels that appear in the output of this debug command.

Table 126 debug mpls lfib cef Field Descriptions

Field	Description
tagcon	The name of the subsystem issuing the debug output (Label Control).
LFIB	The name of the subsystem issuing the debug output.
tc_ip_rtlookup fail on x.y.w.z/m: subnet_lookup failed	The destination with IP address and mask shown is not in the routing table.
route tag chg x.y.w.z/m	Request to create the LFIB entry for the specified prefix/mask.
idx=-1	The index within the FIB entry of the path whose LFIB entry is being created. The parameter –1 means all paths for this FIB entry.
inc=s	Incoming label of the entry being processed.
outg=s	Outgoing label of the entry being processed.
enabled=0xn	Bit mask indicating the types of label switching currently enabled: • 0x1 = dynamic • 0x2 = TSP tunnels • 0x3 = both
fib complete delete	Indicates that the FIB entry is being deleted.
prefix=x.y.w.z/m	A destination prefix.
delete_info=1	Indicates that label_info is also being deleted.
deactivate tag rew for x.y.w.z/m	Indicates that label rewrite for specified prefix is being deleted.
index=n	Index of path in the FIB entry being processed.
set fib rew: pfx x.y.w.z/m	Indicates that label rewrite is being installed or deleted from the FIB entry for the specified destination for label imposition purposes.

Table 126 debug mpls lfib cef Field Descriptions (continued)

Field	Description
add=0	Indicates that label rewrite is being deleted from the FIB (no longer imposing labels).
tag_rew->adj=s	Adjacency of label rewrite for label imposition.
resolve tag rew,prefix=x.y.w.z/m	Indicates that the FIB route to the specified prefix is being resolved.
no tag_info	Indicates that there is no label_info for the destination (destination not labeled).
no parent	Indicates that the route is not recursive.
fib scanner start	Indicates that the periodic scan of the FIB has started.
needed:1	Indicates that the LFIB needs the FIB to be scanned.
unres:n	Indicates the number of unresolved TFIB entries.
mac:n	Indicates the number of TFIB entries missing MAC strings.
loadinfo:n	Indicates whether the nonrecursive accounting state has changed and whether the loadinfo information in the LFIB needs to be adjusted.
fib upd loadinf x.y.w.z/m	Indicates that a check for nonrecursive accounting is being made and that the LFIB loadinfo information for the specified prefix is being updated.
tag=s	Incoming label of entry.
fib no loadin	Indicates that the corresponding FIB entry has no loadinfo.
tfib no loadin	Indicates that the LFIB entry has no loadinfo.
fib check cleanup for x.y.w.z/m	Indicates that a check is being made on the LFIB entry for the specified destination to determine if rewrite needs to be removed from the LFIB.
return_value=x	If *x* is 0, indicates that no change has occurred in the LFIB entry. If *x* is 1, there was a change.
fib_scanner_end	Indicates that the FIB scan has come to an end.
create dynamic entry for x.y.w.z/m	Indicates that the LFIB has been enabled and that an LFIB entry is being created for the specified destination.
call find_route_tags	Indicates that the labels for that destination are being requested.
dist_method=n	Identifies the label distribution method—TDP, TC-ATM, and so on.
next_hop=x.y.z.w	Identifies the next hop for the destination.
interface name	Identifies the outgoing interface for the destination.
create tag info	Indicates that a label_info data structure is being created for the destination.
has no info	Indicates that the destination does not already have label_info.
finish fib re x.y.z.w/m	Indicates that the LFIB entry for the specified route is being completed.
parent outg tag s	If recursive, specifies the outgoing label of the route through which it is recursive (the parent). If not recursive, *s* = "no parent."
tagcon: route_tag_change for: x.y.z.w/m	Indicates that label control is notifying LFIB that labels are available for the specified destination.
intag s	Identifies the incoming label for the destination.

Table 126 debug mpls lfib cef Field Descriptions (continued)

Field	Description
outtag s	Identifies the outgoing label for the destination.
nexthop tsr x.y.z.w.i	Identifies the TDP ID of the next hop that sent the tag.

Related Commands

Command	Description
debug mpls lfib cef	Prints detailed information about label rewrites being created, resolved, and deactivated as CEF routes are added, changed, or removed.
debug mpls lfib lsp	Prints detailed information about label rewrites being created and deleted as LSP tunnels are added or removed.
debug mpls lfib state	Traces what happens when label switching is enabled or disabled.
debug mpls lfib struct	Traces the allocation and freeing of LFIB-related data structures, including the LFIB itself, label rewrites, and label_info data.

debug mpls lfib enc

To print detailed information about label encapsulations while label rewrites are created or updated and placed in the label forwarding information base (LFIB), use the **debug mpls lfib enc** privileged EXEC command. The **no** form of this command disables debugging output.

debug mpls lfib enc

no debug mpls lfib enc

Syntax Description

This command has no keywords or arguments.

Defaults

This command has no default behavior or values.

Command Modes

Privileged EXEC

Command History

Release	Modification
11.1CT	This command was introduced.
12.1(3)T	This command was modified to reflect new MPLS IETF terminology and CLI command syntax.

Usage Guidelines

Several lines of output are produced for each route placed into the LFIB. If your router has thousands of labeled routes, issue this command with care. When label switching is first enabled, each of these routes is placed into the LFIB and a label encapsulation is created. The command output shows you on which adjacency the label rewrite is being created and the labels assigned.

Examples

The following is an example of output generated when you issue the **debug mpls lfib enc** command. This example shows the encapsulations for three routes that have been created and placed into the LFIB.

```
Router# debug mpls lfib enc

TFIB: finish res:inc tag=28,outg=Imp_null,next_hop=10.93.72.13,Ethernet4/0/3
TFIB: update_mac, mac_length = 14,addr=10.93.72.13,idb=Ethernet4/0/3
TFIB: get ip adj: addr=10.93.72.13,is_p2p=0,fibidb=Ethernet4/0/3,linktype=7
TFIB: get tag adj: addr=10.93.72.13,is_p2p=0,fibidb=Ethernet4/0/3,linktype=79
TFIB: encaps:inc=28,outg=Imp_null,idb:Ethernet4/0/3,sizes 14,14,1504,type 0
TFIB: finish res:inc tag=30,outg=27,next_hop=10.93.72.13,Ethernet4/0/3
TFIB: get ip adj: addr=10.93.72.13,is_p2p=0,fibidb=Ethernet4/0/3,linktype=7
TFIB: get tag adj: addr=10.93.72.13,is_p2p=0,fibidb=Ethernet4/0/3,linktype=79
TFIB: encaps:inc=30,outg=27,idb:Ethernet4/0/3,sizes 14,18,1500,type 0
TFIB: finish res:inc tag=30,outg=10,next_hop=0.0.0.0,ATM0/0.1
TFIB: get ip adj: addr=0.0.0.0,is_p2p=1,fibidb=ATM0/0.1,linktype=7
TFIB: get tag adj: addr=0.0.0.0,is_p2p=1,fibidb=ATM0/0.1,linktype=79
TFIB: encaps:inc=30,outg=10,idb:ATM0/0,sizes 4,8,4470,type 1
```

Table 127 describes the significant fields shown in the display.

Table 127 debug mpls lfib enc Field Descriptions

Field	Description
TFIB	Identifies the source of the message as the LFIB subsystem.
finish res	Identifies that the LFIB resolution is being finished.
inc tag=x or inc=x	An incoming (local) label for the LFIB entry is being created. Labels can be numbers or special values.
outg=y	An outgoing (remote) label for the LFIB entry is being created.
next_hop=a.b.c.d	IP address of the next hop for the destination.
interface	The outgoing interface through which a packet will be sent.
get ip adj	Identifies that the IP adjacency to use in the LFIB entry is being determined.
get tag adj	Identifies that the label switching adjacency to use for the LFIB entry is being determined.
addr = a.b.c.d	The IP address of the adjacency.
is_p2p=x	If *x* is 1, this is a point-to-point adjacency. If *x* is 0, it is not.
fibidb = s	Indicates the interface of the adjacency.
linktype = x	The link type of the adjacency, as follows: • 7 = LINK_IP • 79 = LINK_TAG
sizes x,y,z	Indicates the following values: • x = length of macstring • y = length of tag encapsulation • z = tag MTU
type = x	Tag encapsulation type, as follows: • 0 = normal • 1 = TCATM • 2 = TSP tunnel
idb:s	Indicates the outgoing interface.
update_mac	Indicates that the macstring of the adjacency is being updated.

Table 128 describes the special labels, which sometimes appear in the debug output, and their meanings.

Table 128 Special Labels Appearing in debug Command Output

Special Label	Meaning
Unassn—Inital value	No label assigned yet.
Unused	This destination does not have a label (for example, a BGP route).
Withdrn	The label for this destination has been withdrawn.
Unkn	This destination should have a label, but it is not yet known.
Get_res	A recursive route that will get a label when resolved.

Table 128 Special Labels Appearing in debug Command Output (continued)

Special Label	Meaning
Exp_null	Explicit null label—used over TC-ATM.
Imp_null	Implicit null label—for directly connected routes.
Tun_hd	Identifies head of TSP tunnel.

Related Commands

Command	Description
debug mpls lfib cef	Prints detailed information about label rewrites being created, resolved, and deactivated as CEF routes are added, changed, or removed.
debug mpls lfib lsp	Prints detailed information about label rewrites being created and deleted as LSP tunnels are added or removed.
debug mpls lfib state	Traces what happens when label switching is enabled or disabled.
debug mpls lfib struct	Traces the allocation and freeing of LFIB-related data structures, including the LFIB itself, label rewrites, and label_info data.

debug mpls lfib lsp

To print detailed information about label rewrites being created and deleted as LSP tunnels are added or removed, use the **debug mpls lfib lsp** EXEC command. The **no** form of this command disables debugging output.

debug mpls lfib lsp

no debug mpls lfib lsp

Syntax Description

This command has no keywords or arguments.

Defaults

This command has no default behavior or values.

Command Modes

Privileged EXEC

Command History

Release	Modification
11.1CT	This command was introduced.
12.1(3)T	This command was modified to reflect new MPLS IETF terminology and CLI command syntax.

Examples

The following is sample output generated from the **debug mpls lfib lsp** command:

```
Router# debug mpls lfib lsp

TSP-tunnel related TFIB services debugging is on

TFIB: tagtun,next hop=10.93.72.13,inc=35,outg=1,idb=Et4/0/3
TFIB: tsptunnel:next hop=10.93.72.13,inc=35,outg=Imp_null,if_number=7
TFIB: tsptun update loadinfo:tag=35,loadinfo_reqd=0,no new loadinfo,no old loadinfo
TFIB: tagtun tag chg linec,fiblc=0,in tg=35,o tg=1,if=7,nh=10.93.72.13
TFIB: tagtun,next hop=10.92.0.7,inc=36,outg=1,idb=Et4/0/2
TFIB: tsptunnel:next hop=10.92.0.7,inc=36,outg=Imp_null,if_number=6
TFIB: tsptun update loadinfo:tag=36,loadinfo_reqd=0,no new loadinfo,no old loadinfo
TFIB: tagtun tag chg linec,fiblc=0,in tg=36,o tg=1,if=6,nh=10.92.0.7
TFIB: tagtun_delete, inc = 36
tagtun tag del linec,itag=12
TFIB: tagtun_delete, inc = 35
tagtun tag del linec,itag=12
TFIB: tagtun,next hop=10.92.0.7,inc=35,outg=1,idb=Et4/0/2
TFIB: tsptunnel:next hop=10.92.0.7,inc=35,outg=Imp_null,if_number=6
TFIB: tsptun update loadinfo:tag=35,loadinfo_reqd=0,no new loadinfo,no old loadinfo
TFIB: tagtun tag chg linec,fiblc=0,in tg=35,o tg=1,if=6,nh=10.92.0.7

On VIP:
TFIB: tagtun chg msg,in tg=35,o tg=1,nh=10.93.72.13,if=7
TFIB: tsptunnel:next hop=10.93.72.13,inc=35,outg=Imp_null,if_number=7
TFIB: tsptun update loadinfo:tag=35,loadinfo_reqd=0,no new loadinfo,no old loadinfo
TFIB: tagtun chg msg,in tg=36,o tg=1,nh=10.92.0.7,if=6
TFIB: tsptunnel:next hop=10.92.0.7,inc=36,outg=Imp_null,if_number=6
```

```
TFIB: tsptun update loadinfo:tag=36,loadinfo_reqd=0,no new loadinfo,no old loadinfo
TFIB: tagtun chg msg,in tg=35,o tg=1,nh=10.93.72.13,if=7
TFIB: tsptunnel:next hop=10.93.72.13,inc=35,outg=Imp_null,if_number=7
TFIB: tsptun update loadinfo:tag=35,loadinfo_reqd=0,no new loadinfo,no old loadinfo
TFIB: tagtun chg msg,in tg=36,o tg=1,nh=10.92.0.7,if=6
TFIB: tsptunnel:next hop=10.92.0.7,inc=36,outg=Imp_null,if_number=6
TFIB: tsptun update loadinfo:tag=36,loadinfo_reqd=0,no new loadinfo,no old loadinfo
TFIB: tagtun chg msg,in tg=35,o tg=1,nh=10.92.0.7,if=6
TFIB: tsptunnel:next hop=10.92.0.7,inc=35,outg=Imp_null,if_number=6
TFIB: tsptun update loadinfo:tag=35,loadinfo_reqd=0,no new loadinfo,no old loadinfo
```

Table 129 describes the significant fields in the sample display shown above.

Table 129 debug mpls lfib lsp Field Descriptions

Field	Description
tagtun	Name of routine entered.
next hop=x.y.z.w	Next hop for the tunnel being created.
inc=x	Incoming label for this hop of the tunnel being created.
outg=x	Outgoing label (1 means Implicit Null label).
idb=s	Outgoing interface for the tunnel being created.
if_number=7	Interface number of the outgoing interface.
tsptunnel	Name of the routine entered.
tsptun update loadinfo	The procedure being performed.
tag=x	Incoming label of the LFIB slot whose loadinfo is being updated.
loadinfo_reqd=x	Indicates whether a loadinfo is expected for this entry (non-recursive accounting is on).
no new loadinfo	No change required in loadinfo.
no old loadinfo	No previous loadinfo available.
tagtun tag chg linec	Line card is being informed of the TSP tunnel.
fiblc=x	Indicates which line card is being informed (0 means all).
in tg=x	Indicates the incoming label of new TSP tunnel.
o tg=x	Indicates the outgoing label of new TSP tunnel.
if=x	Indicates the outgoing interface number.
nh=x.y.w.z	Indicates the next hop IP address.
tagtun_delete	Indicates that a procedure is being performed: delete a TSP tunnel.
tagtun tag del linec	Informs the line card of the TSP tunnel deletion.
tagtun chg msg	Indicates that the line card has received a message to create a TSP tunnel.

Related Commands

Command	Description
debug mpls lfib cef	Prints detailed information about label rewrites being created, resolved, and deactivated as CEF routes are added, changed, or removed.
debug mpls lfib state	Traces what happens when label switching is enabled or disabled.
debug mpls lfib struct	Traces the allocation and freeing of LFIB-related data structures, including the LFIB itself, label rewrites, and label_info data.

debug mpls lfib state

To trace what happens when label switching is enabled or disabled, use the **debug mpls lfib state** EXEC command. The **no** form of this command disables debugging output.

debug mpls lfib state

no debug mpls lfib state

Syntax Description

This command has no keywords or arguments.

Defaults

This command has no default behavior or values.

Command Modes

Privileged EXEC

Command History

Release	Modification
11.1CT	This command was introduced.
12.1(3)T	This command was modified to reflect new MPLS IETF terminology and CLI command syntax.

Usage Guidelines

Use this command when you wish to trace what happens to the LFIB when you issue the **mpls ip** or the **mpls tsp-tunnel** command.

Examples

The following is sample output generated from the **debug mpls lfib state** command:

```
Router# debug mpls lfib state

TFIB enable/disable state debugging is on
TFIB: Upd tag sb 6(status:0xC1,tmtu:1500,VPI:1-1 VC=0/32,et:0/0/0),lc 0x0
TFIB: intf status chg: idb=Et4/0/2,status=0xC1,oldstatus=0xC3
TFIB: interface dyntag change,change in state to Ethernet4/0/2
TFIB: enable entered, table exists,enabler type=0x2
TFIB: enable, TFIB already enabled, types now 0x3,returning
TFIB: enable entered, table exists,enabler type=0x1
TFIB: disable entered, table exists,type=0x1

TFIB: cleanup: tfib[32] still non-0

On linecard only:

TFIB: disable lc msg recvd, type=0x1
TFIB: Ethernet4/0/1 fibidb subblock message received
TFIB: enable lc msg recvd, type=0x1
TFIB: Tunnel301 set encapfix to 0x6016A97C
```

Table 130 describes the significant fields shown in the display.

Table 130 debug mpls lfib state Field Descriptions

Field	Description
LFIB	Identifies the source of the message as the LFIB subsystem.
Upd tag sb x	Indicates that the status of the "*x*th" label switching sub-block is being updated, where *x* is the interface number. There is a label switching sub-block for each interface on which label switching has been enabled.
(status:0xC1,tmtu:1500, VPI:1-1VC=0/32, et:0/0/0),lc 0x0)	Identifies the values of the fields in the label switching sub-block, as follows: • status byte • maximum transmission unit (*tmtu*) • range of ATM VPs • control VP • control VC (if this is a TC-ATM interface) • encapsulation type (*et*) • encapsulation information • tunnel interface number (*lc*) • line card number to which the update message is being sent (0 means all line cards)
intf status chg	Indicates that there was an interface status change.
idb=Et4/0/2	Identifies the interface whose status changed.
status=0xC1	Indicates the new status bits in the label switching sub-block of the idb.
oldstatus=0xC3	Indicates the old status bits before the change.
interface dyntag change, change in state to Ethernet4/0/2	Indicates that there was a change in the dynamic label status for the particular interface.
enable entered	Indicates that the code that enables the LFIB was invoked.
TFIB already enabled	Indicates that the LFIB was already enabled when this call was made.
table exists	Indicates that an LFIB table had already been allocated in a previous call.
cleanup: tfib[x] still non-0	Indicates that the LFIB is being deleted, but that slot *x* is still active.
disable lc mesg recvd, type=0x1	Indicates that a message to disable label switching type 1 (dynamic) was received by the line card.
disable entered, table exists,type=0x1	Indicates that a call to disable dynamic label switching was issued.
Ethernet4/0/1 fibidb subblock message received	Indicates that a message giving fibidb status change was received on the line card.
enable lc msg recvd,type=0x1	Indicates that the line card received a message to enable label switching type 1 (dynamic).

Table 130 debug mpls lfib state Field Descriptions (continued)

Field	Description
Tunnel301 set encapfix to 0x6016A97C	Shows that fibidb Tunnel301 on the line card received an encapsulation fixup.
types now 0x3, returning	Shows the value of the bitmask indicating the type of label switching enabled on the interface, as follows: • 0x1—means dynamic label switching • 0x2—means tsp-tunnels • 0x3—means both

Related Commands

Command	Description
debug mpls lfib cef	Prints detailed information about label rewrites being created, resolved, and deactivated as CEF routes are added, changed, or removed.
debug mpls lfib lsp	Prints detailed information about label rewrites being created and deleted as LSP tunnels are added or removed.
debug mpls lfib state	Traces what happens when label switching is enabled or disabled.
debug mpls lfib struct	Traces the allocation and freeing of LFIB-related data structures, including the LFIB itself, label rewrites, and label_info data.

debug mpls lfib struct

To trace the allocation and freeing of LFIB-related data structures, such as the LFIB itself, label rewrites, and label_info data, use the **debug mpls lfib struct** EXEC command. The **no** form of this command disables debugging output.

debug mpls lfib struct

no debug mpls lfib struct

Syntax Description

This command has no keywords or arguments.

Defaults

This command has no default behavior or values.

Command Modes

Privileged EXEC

Command History

Release	Modification
11.1CT	This command was introduced.
12.1(3)T	This command was modified to reflect new MPLS IETF terminology and CLI command syntax.

Examples

The following is sample output generated from the **debug mpls lfib struct** command:

```
Router# debug mpls lfib struct

TFIB data structure changes debugging is on

TFIB: delete tag rew, incoming tag 32
TFIB: remove from tfib,inc tag=32
TFIB: set loadinfo,tag=32,no old loadinfo,no new loadinfo
TFIB: TFIB not in use.  Checking for entries.
TFIB: cleanup: tfib[0] still non-0
TFIB: remove from tfib,inc tag=Tun_hd
TFIB: set loadinfo,tag=Exp_null,no old loadinfo,no new loadinfo
TFIB: TFIB freed.
TFIB: enable, TFIB allocated, size 4024 bytes, maxtag = 500
TFIB: create tag rewrite: inc Tun_hd,outg Unkn
TFIB: add to tfib at Tun_hd, first in circular list, mac=0,enc=0
TFIB: delete tag rew, incoming tag Tun_hd
TFIB: remove from tfib,inc tag=Tun_hd
TFIB: set loadinfo,tag=Exp_null,no old loadinfo,no new loadinfo
TFIB: create tag rewrite: inc Tun_hd,outg Unkn
TFIB: add to tfib at Tun_hd, first in circular list, mac=0,enc=0
TFIB: create tag rewrite: inc 26,outg Unkn
TFIB: add to tfib at 26, first in circular list, mac=0,enc=0
TFIB: add to tfib at 27, added to circular list, mac=0,enc=0
TFIB: delete tag rew, incoming tag Tun_hd
TFIB: remove from tfib,inc tag=Tun_hd
TFIB: set loadinfo,tag=Exp_null,no old loadinfo,no new loadinfo
TFIB: add to tfib at 29, added to circular list, mac=4,enc=8
```

```
TFIB: delete tag rew, incoming tag 29
TFIB: remove from tfib,inc tag=29
```

Table 131 describes the significant fields shown in the display.

Table 131 debug mpls lfib struct Field Descriptions

Field	Description
TFIB	The subsystem issuing the message.
delete tag rew	A label rewrite is being freed.
remove from tfib	A label rewrite is being removed from the LFIB.
inc tag=s	The incoming label of the entry being processed.
set loadinfo	The loadinfo field in the LFIB entry is being set (used for nonrecursive accounting).
tag=s	The incoming label of the entry being processed.
no old loadinfo	The LFIB entry did not have a loadinfo before.
no new loadinfo	The LFIB entry should not have a loadinfo now.
TFIB not in use. Checking for entries.	Label switching has been disabled and the LFIB is being freed up.
cleanup: tfib[x] still non-0	The LFIB is being checked for any entries in use, and entry *x* is the lowest numbered slot still in use.
TFIB freed	The LFIB table has been freed.
enable, TFIB allocated, size x bytes, maxtag = y	Label switching has been enabled and an LFIB of *x* bytes has been allocated. The largest legal label is *y*.
create tag rewrite	A label rewrite is being created.
inc s	The incoming label.
outg s	The outgoing label.
add to tfib at s	A label rewrite has been placed in the LFIB at slots.
first in circular list	This LFIB slot had been empty and this is the first rewrite in the list.
mac=0,enc=0	Length of the mac string and total encapsulation length, including labels.
added to circular list	A label rewrite is being added to an LFIB slot that already had an entry. This rewrite is being inserted in the circular list.

Related Commands

Command	Description
debug mpls lfib cef	Prints detailed information about label rewrites being created, resolved, and deactivated as CEF routes are added, changed, or removed.
debug mpls lfib lsp	Prints detailed information about label rewrites being created and deleted as LSP tunnels are added or removed.
debug mpls lfib state	Traces what happens when label switching is enabled or disabled.

debug mpls packets

To display labeled packets switched by the host router, use the **debug mpls packets** EXEC command. The **no** form of this command disables debugging output.

debug mpls packets [*interface*]

no debug mpls packets [*interface*]

Syntax Description

interface	(Optional.) The interface or subinterface name.

Defaults

Displays all labeled packets regardless of interface.

Command Modes

Privileged EXEC

Command History

Release	Modification
11.1CT	This command was introduced.
12.1(3)T	This command was modified to reflect new MPLS IETF terminology and CLI command syntax.

Usage Guidelines

The optional *interface* parameter restricts the display to only those packets received or sent on the indicated interface.

Note Use this command with care because it generates output for every packet processed. Furthermore, enabling this command causes fast and distributed label switching to be disabled for the selected interfaces. To avoid adversely affecting other system activity, use this command only when traffic on the network is at a minimum.

Examples

The following is sample output from the **debug mpls packets** command:

```
Router# debug mpls packets

TAG: Hs3/0: recvd: CoS=0, TTL=254, Tag(s)=27
TAG: Hs0/0: xmit: (no tag)

TAG: Hs0/0: recvd: CoS=0, TTL=254, Tag(s)=30
TAG: Hs3/0: xmit: CoS=0, TTL=253, Tag(s)=27
```

Table 132 describes the significant fields shown in the display.

Table 132 debug mpls packets Field Descriptions

Field	Description
Hs0/0	The identifier for the interface on which the packet was received or sent.
recvd	Packet received.
xmit	Packet transmitted.
CoS	Class of Service field from the packet label header.
TTL	Time to live field from the packet label header.
(no tag)	Last label popped off the packet and were sent unlabeled.
Tag(s)	A list of labels on the packet, ordered from the top of the stack to the bottom.

Related Commands

Command	Description
show mpls forwarding-table	Displays the contents of the MPLS forwarding table.

debug mpls traffic-eng areas

To print information about traffic engineering area configuration change events, use the **debug mpls traffic-eng areas** privileged EXEC command. To disable debugging output, use the **no** form of this command.

debug mpls traffic-eng areas

no debug mpls traffic-eng areas

Syntax Description

This command has no arguments or keywords.

Defaults

No default behavior or values.

Command Modes

Privileged EXEC

Command History

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

Examples

In the following example, information is printed about traffic engineering area configuration change events:

```
debug mpls traffic-eng areas

TE-AREAS:isis level-1:up event
TE-PCALC_LSA:isis level-1
```

debug mpls traffic-eng autoroute

To print information about automatic routing over traffic engineering tunnels, use the **debug mpls traffic-eng autoroute** privileged EXEC command. To disable debugging output, use the **no** form of this command.

debug mpls traffic-eng autoroute

no debug mpls traffic-eng autoroute

Syntax Description

This command has no arguments or keywords.

Defaults

No default behavior or values.

Command Modes

Privileged EXEC

Command History

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

Examples

In the following example, information is printed about automatic routing over traffic engineering tunnels:

```
debug mpls traffic-eng autoroute

TE-Auto:announcement that destination 0001.0000.0003.00 has 1 tunnels
    Tunnel1 (traffic share 333, nexthop 10.112.0.12)
```

debug mpls traffic-eng link-management admission-control

To print information about traffic engineering LSP admission control on traffic engineering interfaces, use the **debug mpls traffic-eng link-management admission-control** privileged EXEC command. To disable debugging output, use the **no** form of this command.

debug mpls traffic-eng link-management admission-control [**detail**] [*aclnum*]

no debug mpls traffic-eng link-management admission-control [**detail**]

Syntax Description

detail	(Optional) Prints detailed debugging information.
aclnum	(Optional) Uses the specified access list to filter the debugging information. Prints information only for those LSPs that match the access list.

Defaults

No default behavior or values.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.05(S)	This command was introduced.
12.1(3)T	The **detail** keyword and the *aclnum* argument were added.

Examples

In the following example, information is printed about traffic engineering LSP admission control on traffic engineering interfaces:

```
debug mpls traffic-eng link-management admission-control

TE-LM-ADMIT:tunnel 10.106.0.6 1_10002:created [total 4]
TE-LM-ADMIT:tunnel 10.106.0.6 1_10002: "None" -> "New"
TE-LM-ADMIT:tunnel 10.106.0.6 1_10002: "New" -> "Admitting 2nd Path Leg"
TE-LM-ADMIT:tunnel 10.106.0.6 1_10002: "Admitting 2nd Path Leg" -> "Path Admitted"
TE-LM-ADMIT:Admission control has granted Path query for 10.106.0.6 1_10002 (10.112.0.12)
on link Ethernet4/0/1 [reason 0]
TE-LM-ADMIT:tunnel 10.106.0.6 1_10002: "Path Admitted" -> "Admitting 1st Resv Leg"
TE-LM-ADMIT:tunnel 10.106.0.6 1_10002: "Admitting 1st Resv Leg" -> "Resv Admitted"
TE-LM-ADMIT:Admission control has granted Resv query for 10.106.0.6 1_10002 (10.112.0.12)
on link Ethernet4/0/1 [reason 0]
```

debug mpls traffic-eng link-management advertisements

To print information about resource advertisements for traffic engineering interfaces, use the **debug mpls traffic-eng link-management advertisements** privileged EXEC command. To disable debugging output, use the **no** form of this command.

debug mpls traffic-eng link-management advertisements [**detail**] [*aclnum*]

no debug mpls traffic-eng link-management advertisements [**detail**] [*aclnum*]

Syntax Description

detail	(Optional) Prints detailed debugging information.
aclnum	(Optional) Uses the specified access list to filter the debugging information.

Defaults

No default behavior or values.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.05(S)	This command was introduced.
12.1(3)T	The **detail** keyword was added.

Examples

In the following example, detailed debugging information is printed about resource advertisements for traffic engineering interfaces:

```
debug mpls traffic-eng link-management advertisements detail

TE-LM-ADV:area isis level-1:IGP announcement:link Et4/0/1:info changed
TE-LM-ADV:area isis level-1:IGP msg:link Et4/0/1:includes subnet type (2), described nbrs
(1)
TE-LM-ADV:area isis level-1:IGP announcement:link Et4/0/1:info changed
TE-LM-ADV:area isis level-1:IGP msg:link Et4/0/1:includes subnet type (2), described nbrs
(1)
TE-LM-ADV:LSA:Flooding manager received message:link information change (Et4/0/1)
TE-LM-ADV:area isis level-1:*** Flooding node information ***
  System Information::
    Flooding Protocol:   ISIS
  Header Information::
    IGP System ID:       0001.0000.0001.00
    MPLS TE Router ID:   10.106.0.6
    Flooded Links:       1
  Link ID:: 0
    Link IP Address:     10.1.0.6
    IGP Neighbor:        ID 0001.0000.0001.02
    Admin. Weight:       10
    Physical Bandwidth:  10000 kbits/sec
    Max Reservable BW:   5000 kbits/sec
    Downstream::
      Reservable Bandwidth[0]:       5000 kbits/sec
```

```
      Reservable Bandwidth[1]:       2000 kbits/sec
      Reservable Bandwidth[2]:       2000 kbits/sec
      Reservable Bandwidth[3]:       2000 kbits/sec
      Reservable Bandwidth[4]:       2000 kbits/sec
      Reservable Bandwidth[5]:       2000 kbits/sec
      Reservable Bandwidth[6]:       2000 kbits/sec
Attribute Flags: 0x00000000
```

Table 133 describes the significant fields shown in the display.

Table 133 debug isis mpls traffic-eng link-management advertisements Field Descriptions

Field	Description
Flooding Protocol	IGB that is flooding information for this area.
IGP System ID	Identification that IGP flooding uses in this area to identify this node.
MPLS TE Router ID	MPLS traffic engineering router ID.
Flooded Links	Number of links that are flooded in this area.
Link ID	Index of the link that is being described.
Link IP Address	Local IP address of this link.
IGP Neighbor	IGP neighbor on this link.
Admin. Weight	Administrative weight associated with this link.
Physical Bandwidth	Link's bandwidth capacity (in kbps).
Max Reservable BW	Maximum amount of bandwidth that is currently available for reservation at this priority.
Reservable Bandwidth	Amount of bandwidth that is available for reservation.
Attribute Flags	Attribute flags of the link being flooded.

debug mpls traffic-eng link-management bandwidth-allocation

To print detailed information about bandwidth allocation for traffic engineering LSPs, use the **debug mpls traffic-eng link-management bandwidth-allocation** privileged EXEC command. To disable debugging output, use the **no** form of this command.

debug mpls traffic-eng link-management bandwidth-allocation [**detail**] [*aclnum*]

no debug mpls traffic-eng link-management bandwidth-allocation [**detail**] [*aclnum*]

Syntax Description

detail	(Optional) Prints detailed debugging information.
aclnum	(Optional) Uses the specified access list to filter the debugging information. Prints information only for those LSPs that match the access list.

Defaults

No default behavior or values.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.05(S)	This command was introduced.
12.1(3)T	The **detail** keyword and the *aclnum* argument were added.

Examples

In the following example, information is printed about bandwidth allocation for traffic engineering LSPs:

```
debug mpls traffic-eng link-management bandwidth-allocation
TE-LM-BW:tunnel 10.106.0.6 1_10002:requesting Downstream bw hold (3000000 bps [S]) on link
Et4/0/1
TE-LM-BW:tunnel 10.106.0.6 1_10002:Downstream bw hold request succeeded
TE-LM-BW:tunnel 10.106.0.6 1_10002:requesting Downstream bw lock (3000000 bps [S]) on link
Et4/0/1
TE-LM-BW:tunnel 10.106.0.6 1_10002:Downstream bw lock request succeeded×_„Rs
```

Related Commands

Command	Description
debug mpls traffic-eng link-management admission-control	Prints information about traffic engineering LSP admission control on traffic engineering interfaces.
debug mpls traffic-eng link-management errors	Prints information about errors encountered during any traffic engineering link management procedure.

debug mpls traffic-eng link-management errors

To print information about errors encountered during any traffic engineering link management procedure, use the **debug mpls traffic-eng link-management errors** privileged EXEC command. To disable debugging output, use the **no** form of this command.

debug mpls traffic-eng link-management errors [**detail**]

no debug mpls traffic-eng link-management errors [**detail**]

Syntax Description

detail	(Optional) Prints detailed debugging information.

Defaults

No default behavior or values.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.1(3)T	This command was introduced.

Examples

In the following example, detailed debugging information is printed about errors encountered during a traffic engineering link management procedure:

```
debug mpls traffic-eng link-management errors detail

00:04:48 TE-LM-ROUTING: link Et1/1/1: neighbor 0010.0000.0012.01: add to IP peer db failed
```

Related Commands

Command	Description
debug mpls traffic-eng link-management admission-control	Prints information about traffic engineering LSP admission control on traffic engineering interfaces.
debug mpls traffic-eng link-management advertisements	Prints information about resource advertisements for traffic engineering interfaces.
debug mpls traffic-eng link-management bandwidth-allocation	Prints information about bandwidth allocation for traffic engineering LSPs.
debug mpls traffic-eng link-management events	Prints information about traffic engineering link management system events.
debug mpls traffic-eng link-management igp-neighbors	Prints information about changes to the link management databases of IGP neighbors.
debug mpls traffic-eng link-management links	Prints information about traffic engineering link management interface events.

debug mpls traffic-eng link-management events

To print information about traffic engineering link management system events, use the **debug mpls traffic-eng link-management events** privileged EXEC command. To disable debugging output, use the **no** form of this command.

debug mpls traffic-eng link-management events [**detail**]

no debug mpls traffic-eng link-management events [**detail**]

Syntax Description

detail	(Optional) Prints detailed debugging information.

Defaults

No default behavior or values.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.05(S)	This command was introduced.
12.1(3)T	The **detail** keyword was added.

Examples

In the following example, detailed debugging information is printed about traffic engineering link management system events:

```
debug mpls traffic-eng link-management events detail

TE-LM-EVENTS:stopping MPLS TE Link Management process
TE-LM-EVENTS:MPLS TE Link Management process dying now
```

debug mpls traffic-eng link-management igp-neighbors

To print information about changes to the link management database of IGP neighbors, use the **debug mpls traffic eng link-management igp-neighbors** privileged EXEC command. To disable debugging output, use the **no** form of this command.

debug mpls traffic-eng link-management igp-neighbors [**detail**]

no debug mpls traffic-eng link-management igp-neighbors [**detail**]

Syntax Description

detail	(Optional) Prints detailed debugging information.

Defaults

No default behavior or values.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.05(S)	This command was introduced.
12.1(3)T	The **detail** keyword was added.

Examples

In the following example, detailed debugging information is printed about changes to the link management database of IGP neighbors:

```
debug mpls traffic-eng link-management igp-neighbors detail

TE-LM-NBR:link AT0/0.2:neighbor 0001.0000.0002.00:created (isis level-1, 10.42.0.10,
Up)[total 2]
```

Related Commands

Command	Description
debug mpls traffic-eng link-management events	Prints information about traffic engineering-related ISIS events.

debug mpls traffic-eng link-management links

To print information about traffic engineering link management interface events, use the **debug mpls traffic-eng link-management links** privileged EXEC command. To disable debugging output, use the **no** form of this command.

debug mpls traffic-eng link-management links [**detail**]

no debug mpls traffic-eng link-management links [**detail**]

Syntax Description

detail	(Optional) Prints detailed debugging information.

Defaults

No default behavior or values.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.05(S)	This command was introduced.
12.1(3)T	The **detail** keyword was added.

Examples

In the following example, detailed debugging information is printed about traffic engineering link management interface events:

```
debug mpls traffic-eng link-management links detail

TE-LM-LINKS:link AT0/0.2:RSVP enabled
TE-LM-LINKS:link AT0/0.2:increasing RSVP bandwidth from 0 to 5000000
TE-LM-LINKS:link AT0/0.2:created [total 2]
TE-LM-LINKS:Binding MPLS TE LM Admission Control as the RSVP Policy Server on ATM0/0.2
TE-LM-LINKS:Bind attempt succeeded
TE-LM-LINKS:link AT0/0.2:LSP tunnels enabled
```

debug mpls traffic-eng link-management preemption

To print information about traffic engineering LSP preemption, use the **debug mpls traffic-eng link-management preemption** privileged EXEC command. To disable debugging output, use the **no** form of this command.

debug mpls traffic-eng link-management preemption [**detail**]

no debug mpls traffic-eng link-management preemption [**detail**]

Syntax Description

detail	(Optional) Prints detailed debugging information.

Defaults

No default behavior or values.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.1(3)T	This command was introduced.

Examples

In the following example, detailed debugging information is printed about traffic engineering LSP preemption:

```
debug mpls traffic-eng link-management preemption detail

TE-LM-BW:preempting Downstream bandwidth, 1000000, for tunnel 10.106.0.6 2_2
TE-LM-BW:building preemption list to get bandwidth, 1000000, for tunnel 10.106.0.6 2_2
(priority 0)
TE-LM-BW:added bandwidth, 3000000, from tunnel 10.106.0.6 1_2 (pri 1) to preemption list
TE-LM-BW:preemption list build to get bw, 1000000, succeeded (3000000)
TE-LM-BW:preempting bandwidth, 1000000, using plist with 1 tunnels
TE-LM-BW:tunnel 10.106.0.6 1_2:being preempted on AT0/0.2 by 10.106.0.6 2_2
TE-LM-BW:preemption of Downstream bandwidth, 1000000, succeeded
```

debug mpls traffic-eng link-management routing

To print information about traffic engineering link management routing resolutions that can be performed to help RSVP interpret explicit route objects, use the **debug mpls traffic-eng link-management routing** privileged EXEC command. Use the **no** form of this command to disable debugging output.

debug mpls traffic-eng link-management routing [**detail**]

no debug mpls traffic-eng link-management routing [**detail**]

Syntax Description

detail	(Optional) Prints detailed debugging information.

Defaults

No default behavior or values.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.05(S)	This command was introduced.
12.1(3)T	The **detail** keyword was added.

Examples

In the following example, detailed debugging information is printed about traffic engineering link management routing resolutions that can be performed to help RSVP interpret explicit route objects:

```
debug mpls traffic-eng link-management routing detail

TE-LM-ROUTING:route options to 10.42.0.10:building list (w/ nhop matching)
TE-LM-ROUTING:route options to 10.42.0.10:adding {AT0/0.2, 10.42.0.10}
TE-LM-ROUTING:route options to 10.42.0.10:completed list has 1 links
```

Related Commands

Command	Description
debug ip rsvp	Prints information about RSVP signalling events.

debug mpls traffic-eng load-balancing

To print information about unequal cost load balancing over traffic engineering tunnels, use the **debug mpls traffic-eng load-balancing** privileged EXEC command. To disable debugging output, use the **no** form of this command.

debug mpls traffic-eng load-balancing

no debug mpls traffic-eng load-balancing

Syntax Description

This command has no arguments or keywords.

Defaults

No default behavior or values.

Command Modes

Privileged EXEC

Command History

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

Examples

In the following example, information is printed about unequal cost load balancing over traffic engineering tunnels:

```
debug mpls traffic-eng load-balancing

TE-Load:10.210.0.0/16, 2 routes, loadbalancing based on MPLS TE bandwidth
TE-Load:10.200.0.0/16, 2 routes, loadbalancing based on MPLS TE bandwidth
```

debug mpls traffic-eng path

To print information about traffic engineering path calculation, use the **debug mpls traffic-eng path** privileged EXEC command. To disable debugging output, use the **no** form of this command.

debug mpls traffic-eng path {*num* | **lookup** | **spf** | **verify**}

no debug mpls traffic-eng path {*num* | **lookup** | **spf** | **verify**}

Syntax Description

num	Prints path calculation information only for the local tunneling interface with unit number *num*.
lookup	Prints information for path lookups.
spf	Prints information for shortest path first (SPF) calculations.
verify	Prints information for path verifications.

Defaults

No default behavior or values.

Command Modes

Privileged EXEC

Command History

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

Examples

In the following example, information is printed about the calculation of the traffic engineering path:

```
debug mpls traffic-eng path lookup

TE-PCALC:Tunnel1000 Path Setup to 10.110.0.10:FULL_PATH
TE-PCALC:bw 0, min_bw 0, metric:0
TE-PCALC:setup_pri 0, hold_pri 0
TE-PCALC:affinity_bits 0x0, affinity_mask 0xFFFF
TE-PCALC_PATH:create_path_hoplist:ip addr 10.42.0.6 unknown.
```

debug mpls traffic-eng topology change

To print information about traffic engineering topology change events, use the **debug mpls traffic-eng topology change** privileged EXEC command. To disable debugging output, use the **no** form of this command.

debug mpls traffic-eng topology change

no debug mpls traffic-eng topology change

Syntax Description

This command has no arguments or keywords.

Defaults

No default behavior or values.

Command Modes

Privileged EXEC

Command History

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

Examples

In the following example, information is printed about traffic engineering topology change events:

```
debug mpls traffic-eng topology change

TE-PCALC_LSA:NODE_CHANGE_UPDATE isis level-1
        link flags:LINK_CHANGE_BW
        system_id:0001.0000.0001.00, my_ip_address:10.42.0.6
        nbr_system_id:0001.0000.0002.00, nbr_ip_address 10.42.0.10
```

debug mpls traffic-eng topology lsa

To print information about traffic engineering topology link state advertisement (LSA) events, use the **debug mpls traffic-eng topology lsa** privileged EXEC command. To disable debugging output, use the **no** form of this command.

debug mpls traffic-eng topology lsa

no debug mpls traffic-eng topology lsa

Syntax Description

This command has no arguments or keywords.

Defaults

No default behavior or values.

Command Modes

Privileged EXEC

Command History

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

Examples

In the following example, information is printed about traffic engineering topology LSA events:

```
debug mpls traffic-eng topology lsa

TE-PCALC_LSA:node_lsa_add:Received a LSA:flags 0x1 !

IGP Id:0001.0000.0001.00, MPLS TE Id:10.106.0.6 is VALID has 2 links (frag_id 0)
      link[0 ]:Nbr IGP Id:0001.0000.0001.02
          frag_id 0, Intf Address:0.0.0.0
          admin_weight:10, attribute_flags:0x0

      link[1 ]:Nbr IGP Id:0001.0000.0002.00
          frag_id 0, Intf Address:10.42.0.6, Nbr Intf Address:10.42.0.10
          admin_weight:100, attribute_flags:0x0
TE-PCALC_LSA:(isis level-1):Received lsa:

IGP Id:0001.0000.0001.00, MPLS TE Id:10.106.0.6 Router Node  id 8
      link[0 ]:Nbr IGP Id:0001.0000.0002.00, nbr_node_id:9, gen:114
          frag_id 0, Intf Address:10.42.0.6, Nbr Intf Address:10.42.0.10
          admin_weight:100, attribute_flags:0x0
          physical_bw:155520 (kbps), max_reservable_bw:5000 (kbps)
               allocated_bw   reservable_bw      allocated_bw   reservable_bw
               ------------   -------------      ------------   -------------
        bw[0]:0              5000         bw[1]:3000           2000
        bw[2]:0              2000         bw[3]:0              2000
        bw[4]:0              2000         bw[5]:0              2000
        bw[6]:0              2000         bw[7]:0              2000
```

debug mpls traffic-eng tunnels errors

To print information about errors encountered during any traffic engineering tunnel management procedure, use the **debug mpls traffic-eng tunnels errors** privileged EXEC command. To disable debugging output, use the **no** form of this command.

debug mpls traffic-eng tunnels errors [**detail**]

no debug mpls traffic-eng tunnels errors [**detail**]

Syntax Description

detail	(Optional) Prints detailed debugging information.

Defaults

No default behavior or values.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.1(3)T	This command was introduced.

Examples

In the following example, detailed debugging information is printed about errors encountered during a traffic engineering tunnel management procedure:

```
debug mpls traffic-eng tunnels errors

00:04:14: LSP-TUNNEL-SIG: Tunnel10012[1]: path verification failed (unprotected) [Can't
use link 10.12.4.4 on node 10.0.0.4]
```

debug mpls traffic-eng tunnels events

To print information about traffic engineering tunnel management system events, use the **debug mpls traffic-eng tunnels events** privileged EXEC command. To disable debugging output, use the **no** form of this command.

debug mpls traffic-eng tunnels events [**detail**]

no debug mpls traffic-eng tunnels events [**detail**]

Syntax Description

detail	(Optional) Prints detailed debugging information.

Defaults

No default behavior or values.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.05(S)	This command was introduced.
12.1(3)T	The **detail** keyword was added.

Examples

In the following example, detailed debugging information is printed about traffic engineering tunnel management system events:

```
debug mpls traffic-eng tunnels events detail

LSP-TUNNEL:received event:interface admin. down [Ethernet4/0/1]
LSP-TUNNEL:posting action(s) to all-tunnels:
              check static LSPs
LSP-TUNNEL:scheduling pending actions on all-tunnels
LSP-TUNNEL:applying actions to all-tunnels, as follows:
              check static LSPs
```

debug mpls traffic-eng tunnels labels

To print information about MPLS label management for traffic engineering tunnels, use the **debug mpls traffic-eng tunnels labels** privileged EXEC command. To disable debugging output, use the **no** form of this command.

debug mpls traffic-eng tunnels labels [**detail**] [*aclnum*]

no debug mpls traffic-eng tunnels labels [**detail**] [*aclnum*]

Syntax Description

detail	(Optional) Prints detailed debugging information.
aclnum	(Optional) Uses the specified access list to filter the debugging information. Prints information only about traffic engineering tunnels that match the access list.

Defaults

No default behavior or values.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.05(S)	This command was introduced.
12.1(3)T	The **detail** keyword and the *aclnum* argument were added.

Examples

In the following example, detailed debugging information is printed about MPLS label management for traffic engineering tunnels:

```
debug mpls traffic-eng tunnels labels detail

LSP-TUNNEL-LABELS:tunnel 10.106.0.6 1 [2]:fabric PROGRAM request
LSP-TUNNEL-LABELS:tunnel 10.106.0.6 1 [2]:programming label 16 on output interface
ATM0/0.2
LSP-TUNNEL-LABELS:descriptor 71FA64:continuing "Program" request
LSP-TUNNEL-LABELS:descriptor 71FA64:set "Interface Point Out State" to, allocated
LSP-TUNNEL-LABELS:# of resource points held for "default" interfaces:2
LSP-TUNNEL-LABELS:descriptor 71FA64:set "Fabric State" to, enabled
LSP-TUNNEL-LABELS:descriptor 71FA64:set "Fabric Kind" to, default (LFIB)
LSP-TUNNEL-LABELS:descriptor 71FA64:set "Fabric State" to, set
LSP-TUNNEL-LABELS:tunnel 10.106.0.6 1 [2]:fabric PROGRAM reply
```

To restrict output to information about a single tunnel, you can configure an access list and supply it to the **debug** command. Configure the access list as follows:

```
Router(config-ext-nacl)# permit udp host scr_address host dst_address eq tun intfc
```

For example, if tunnel 10012 has destination 10.0.0.11 and source 10.0.0.4, as determined by **show mpls traffic-eng tunnels** command, the following access list could be configured and added to the **debug** command:

```
Router(config-ext-nacl)# permit udp host 10.0.0.4 10.0.0.11 eq 10012
```

debug mpls traffic-eng tunnels reoptimize

To print information about traffic engineering tunnel re-optimizations, use the **debug mpls traffic-eng tunnels reoptimize** privileged EXEC command. To disable debugging output, use the **no** form of this command.

debug mpls traffic-eng tunnels reoptimize [**detail**] [*aclnum*]

no debug mpls traffic-eng tunnels reoptimize [**detail**] [*aclnum*]

Syntax Description

detail	(Optional) Prints detailed debugging information.
aclnum	(Optional) Uses the specified access list to filter the debugging information. Prints information about only those traffic engineering tunnel reoptimizations that match the access list.

Defaults

No default behavior or values.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.05(S)	This command was introduced.
12.1(3)T	The **detail** keyword and the *aclnum* argument were added.

Examples

In the following example, detailed debugging information is printed about traffic engineering tunnel re-optimizations that match access list number 101:

```
debug mpls traffic-eng tunnels reoptimize detail 101

LSP-TUNNEL-REOPT:Tunnel1 curr option 2 (0x6175CF8C), activate new option 2
LSP-TUNNEL-REOPT:Tunnel1 new path:option 2 [10002], weight 20
LSP-TUNNEL-REOPT:Tunnel1 old path:option 2 [2], weight 110
LSP-TUNNEL-REOPT:Tunnel1 [10002] set as reopt
LSP-TUNNEL-REOPT:Tunnel1 path option 2 [10002] installing as current
LSP-TUNNEL-REOPT:Tunnel1 [2] removed as current
LSP-TUNNEL-REOPT:Tunnel1 [2] set to delayed clean
LSP-TUNNEL-REOPT:Tunnel1 [10002] removed as reopt
LSP-TUNNEL-REOPT:Tunnel1 [10002] set to current
```

debug mpls traffic-eng tunnels signalling

To print information about traffic engineering tunnel signalling operations, use the **debug mpls traffic-eng tunnels signalling** privileged EXEC command. To disable debugging output, use the **no** form of this command.

debug mpls traffic-eng tunnels signalling [**detail**] [*aclnum*]

no debug mpls traffic-eng tunnels signalling [**detail**] [*aclnum*]

Syntax Description

detail	(Optional) Prints detailed debugging information.
aclnum	(Optional) Uses the specified access list to filter the debugging information. Prints information about only those traffic engineering tunnel signalling operations that match the access list.

Defaults

No default behavior or values.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.05(S)	This command was introduced.
12.1(3)T	The **detail** keyword and the aclnum argument were added.

Examples

In the following example, detailed debugging information is printed about traffic engineering tunnel signalling operations that match access list number 101:

```
debug mpls traffic-eng tunnels signalling detail 101

LSP-TUNNEL-SIG:tunnel Tunnel1 [2]:RSVP head-end open
LSP-TUNNEL-SIG:tunnel Tunnel1 [2]:received Path NHOP CHANGE
LSP-TUNNEL-SIG:Tunnel1 [2]:first hop change:0.0.0.0 --> 10.1.0.10
LSP-TUNNEL-SIG:received ADD RESV request for tunnel 10.106.0.6 1 [2]
LSP-TUNNEL-SIG:tunnel 10.106.0.6 1 [2]:path next hop is 10.1.0.10 (Et4/0/1)
LSP-TUNNEL-SIG:Tunnel1 [2] notified of new label information
LSP-TUNNEL-SIG:sending ADD RESV reply for tunnel 10.106.0.6 1 [2]
```

debug mpls traffic-eng tunnels state

To print information about state maintenance for traffic engineering tunnels, use the **debug mpls traffic-eng tunnels state** privileged EXEC command. To disable debugging output, use the **no** form of this command.

debug mpls traffic-eng tunnels state [**detail**] [*aclnum*]

no debug mpls traffic-eng tunnels state [**detail**] [*aclnum*]

Syntax Description

detail	(Optional) Prints detailed debugging information.
aclnum	(Optional) Uses the specified access list to filter the debugging information. Prints information about state maintenance for traffic engineering tunnels that match the access list.

Defaults

No default behavior or values.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.1(3)T	This command was introduced.

Examples

In the following example, detailed debugging information is printed about state maintenance for traffic engineering tunnels that match access list number 99:

```
debug mpls traffic-eng tunnels state detail 99

LSP-TUNNEL:tunnel 10.106.0.6 1 [2]: "Connected" -> "Disconnected"
LSP-TUNNEL:Tunnel1 received event:LSP has gone down
LSP-TUNNEL:tunnel 10.106.0.6 1 [2]: "Disconnected" -> "Dead"
LSP-TUNNEL-SIG:Tunnel1:changing state from up to down
LSP-TUNNEL:tunnel 10.106.0.6 1 [2]: "Dead" -> "Connected"
```

debug mpls traffic-eng tunnels timers

To print information about traffic engineering tunnel timer management, use the **debug mpls traffic-eng tunnels timers** privileged EXEC command. To disable debugging output, use the **no** form of this command.

debug mpls traffic-eng tunnels timers [**detail**] [*aclnum*]

no debug mpls traffic-eng tunnels timers [**detail**] [*aclnum*]

Syntax Description

detail	(Optional) Prints detailed debugging information.
aclnum	(Optional) Uses the specified access list to filter the debugging information. Prints information about traffic engineering tunnel timer management that matches the access list.

Defaults

No default behavior or values.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.05(S)	This command was introduced.
12.1(3)T	The **detail** keyword and the *aclnum* argument were added.

Examples

In the following example, detailed debugging information is printed about traffic engineering tunnel timer management:

```
debug mpls traffic-eng tunnels timers detail

LSP-TUNNEL-TIMER:timer fired for Action Scheduler
LSP-TUNNEL-TIMER:timer fired for Tunnel Head Checkup
```

debug mpoa client

To display MPC debug information, use the **debug mpoa client** privileged EXEC command. The **no** form of this command disables debugging output.

debug mpoa client {**all** | **data** | **egress** | **general** | **ingress** | **keep-alives** | **platform-specific**} [**name** *mpc-name*]

no debug mpoa client {**all** | **data** | **egress** | **general** | **ingress** | **keep-alives** | **platform-specific**} [**name** *mpc-name*]

Syntax Description

all	Displays debugging information for all MPC activity.
data	Displays debugging information for data plane activity only. This option applies only to routers.
egress	Displays debugging information for egress functionality only.
general	Displays general debugging information only.
ingress	Displays debugging information for ingress functionality only.
keep-alives	Displays debugging information for keep-alive activity only.
platform-specific	Displays debugging information for specific platforms only. This option applies only to the Catalyst 5000 series ATM module.
name *mpc-name*	Specifies the name of the MPC with the specified name.

Defaults

The default is debugging turned on for all MPCs.

Command History

Release	Modification
11.3	This command was introduced.

Examples

The following shows how to turn on debugging for the MPC ip_mpc:

```
ATM# debug mpoa client all name ip_mpc
```

Related Commands

Command	Description
debug mpoa server	Displays information about the MPOA server.

debug mpoa server

To display information about the MPOA server, use the **debug mpoa server** privileged EXEC command. The **no** form of this command disables debugging output.

debug mpoa server [**name** *mps-name*]

no debug mpoa server [**name** *mps-name*]

Syntax Description

name *mps-name*	(Optional) Specifies the name of a MPOA server.

Command History

Release	Modification
11.3	This command was introduced.

Usage Guidelines

The **debug mpo server** command optionally limits the output only to the specified MPS.

Examples

The following turns on debugging only for the MPS named ip_mps:

```
Router# debug mpoa server name ip_mps
```

Related Commands

Command	Description
debug modem traffic	Displays MPC debug information.

debug mspi receive

To display debug messages for mail Service Provider Interface (SPI) receive, use the **debug mspi receive** EXEC command. To disable the debug messages, use the **no** form of this command.

debug mspi receive

no debug mspi receive

Syntax Description

This command has no arguments or keywords.

Defaults

No default behavior or values.

Command History

Release	Modification
12.1(3)XI	This command was introduced on the Cisco AS5300 access server.

Examples

The following example displays output from the **debug mspi receive** command.

```
Router# debug mspi receive

Jan  1 05:09:33.890: mspi_tel_num_trans: from: Radhika,
ph#in: fax=5271714 ph#dial: 5271714
Jan  1 05:09:33.890:  incoming destPat(5271714), matched(7), tag(22)
Jan  1 05:09:33.890:  out destPat(5......), tag(20), dgt strip enabled
Jan  1 05:09:33.890: mspi_off_new_rcpt: envlp_to [fax=5271714@rpadmana.cisco.com], 30
Jan  1 05:09:33.890:  tel_numb_dial: 5271714, subaddr:[], cover page
Jan  1 05:09:39.122: mspi_offramp_rfc822_header: msgType=0
Jan  1 05:09:39.122:  envlp_from: [Radhika], 8
Jan  1 05:09:39.122: mspi_off_put_buff: ignore mime type=1, st=CONNECTING, len=0
Jan  1 05:09:39.122: moff_save_buffer: cid=0x1F, mime=9, len=4
Jan  1 05:09:39.122:  offramp disabled receiving!
Dec 31 21:09:44.078: %ISDN-6-CONNECT: Interface Serial0:22 is now connected to 5271714
Jan  1 05:09:52.154: mspi_bridge: cid=0x1F, dst cid=0x22, data dir=OFFRAMP, conf dir=DEST
Jan  1 05:09:52.154: mspi_offramp_send_buffer: cid=0x1F, mime=9
Jan  1 05:09:52.154:  buffer with only CR/LF - set buff_len=0
Jan  1 05:09:52.154: mspi_offramp_send_buffer: cid=0x1F, mime=9 rx BUFF_END_OF_PART,
offramp rcpt enabled
Jan  1 05:09:54.126: mspi_offramp_send_buffer: cid=0x1F, mime=11
Jan  1 05:09:54.134: mspi_offramp_send_buffer: cid=0x1F, mime=11
```

Related Commands

Command	Description
debug mspi send	Displays debug messages for mail SPI send.

debug mspi send

To display debug messages for mail Service Provider Interface (SPI) send, use the **debug mspi send** EXEC command. To disable the debug messages, use the **no** form of this command.

debug mspi send

no debug mspi send

Syntax Description

This command has no arguments or keywords.

Defaults

No default behavior or values.

Command History

Release	Modification
12.1(3)XI	This command was introduced on the Cisco AS5300 access server.

Examples

The following example displays output from the **debug mspi send** command.

```
Router# debug mspi send

*Oct 16 08:40:27.515: mspi_bridge: cid=0x21, dst cid=0x26, data dir=OFFRAMP, conf dir=DEST
*Oct 16 08:40:29.143: mspi_setup_req: for cid=0x27
*Oct 16 08:40:29.147:   envelope_from=5??????@fax.cisco.com
*Oct 16 08:40:29.147:   envelope_to=ilyau@cisco.com
*Oct 16 08:40:30.147: mspi_chk_connect: cid=0x27, cnt=0,
*Oct 16 08:40:30.147:  SMTP connected to the server !
*Oct 16 08:40:30.147: mspi_bridge: cid=0x27, dst cid=0x28, data dir=ONRAMP, conf dir=SRC
*Oct 16 08:40:38.995: mspi_xmit: cid=0x27, st=CONFERENCED, src_cid=0x28, buf cnt=0
```

Related Commands

Command	Description
debug mspi receive	Displays debug messages for mail SPI receive.

debug mta receive all

To show output relating to the activity on the SMTP server, use the **debug mta receive all** EXEC command. Use the **no** form of this command to disable debugging output.

debug mta receive all

no debug mta receive all

Syntax Description

This command has no arguments or keywords.

Defaults

Disabled

Command History

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

Examples

The following example shows the messages exchanged (for example, the handshake) between the e-mail server and the off-ramp gateway.

```
Router# debug mta receive all

Jan  1 05:07:41.314: esmtp_server_work: calling helo
Jan  1 05:07:43.354: esmtp_server_work: calling mail
Jan  1 05:07:45.386: esmtp_server_work: calling rcpt
Jan  1 05:07:47.426: esmtp_server_work: calling data
Jan  1 05:07:49.514: (S)R: 'Content-Type: multipart/mixed;
boundary="------------11F7CD9D2EB3E8B8D5627C62"'
Jan  1 05:07:49.514: (S)R: ''
Jan  1 05:07:49.514: esmtp_server_engine_new_part:
Jan  1 05:07:49.514: (S)R: 'Content-Type: text/plain; charset=us-ascii'
Jan  1 05:07:49.514: (S)R: 'Content-Transfer-Encoding: 7bit'
Jan  1 05:07:49.514: (S)R: ''
Jan  1 05:07:49.514: esmtp_server_engine_new_part:
Jan  1 05:07:49.514: esmtp_server_work: freeing temp header
Jan  1 05:07:49.514: (S)R: 'Content-Type: image/tiff; name="DevTest.8.1610.tif"'
Jan  1 05:07:49.514: (S)R: 'Content-Transfer-Encoding: base64'
Jan  1 05:07:49.514: (S)R: 'Content-Disposition: inline; filename="DevTest.8.1610.tif"'
Jan  1 05:07:49.514: (S)R: ''
Jan  1 05:07:49.514: esmtp_server_engine_update_recipient_status: status=6
Jan  1 05:07:49.514: esmtp_server_engine_new_part:
Jan  1 05:07:49.518: esmtp_server_work: freeing temp header
Jan  1 05:08:03.014: esmtp_server_engine_update_recipient_status: status=7
Jan  1 05:08:04.822: esmtp_server_engine_update_recipient_status: status=6
Jan  1 05:08:33.042: esmtp_server_engine_update_recipient_status: status=7
Jan  1 05:08:34.906: esmtp_server_engine_getline: Unexpected end of file on socket 1
Jan  1 05:08:34.906: esmtp_server_work: error occured with ctx=0x61FFF710, socket=1
```

Related Commands

Command	Description
debug mta send all	Displays output for all of the on-ramp client connections.

debug mta send all

To display output for all of the on-ramp client connections, use the **debug mta send all** EXEC command. Use the **no** form of this command to disable debugging output.

debug mta send all

no debug mta send all

Syntax Description

This command has no arguments or keywords.

Defaults

Disabled

Command History

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

Examples

The following example shows the messages exchanged (for example, the handshake) between the e-mail server and the on-ramp gateway.

```
Router# debug mta send all

*Oct 16 09:04:13.055: esmtp_client_engine_open: from=5??????@fax.cisco.com,
to=ilyau@cisco.com
*Oct 16 09:04:13.055: esmtp_client_engine_add_headers: from_comment=
*Oct 16 09:04:13.111: esmtp_client_work: socket 0 attempting to connect to IP address
171.71.154.56
*Oct 16 09:04:13.111: esmtp_client_work: socket 0 readable for first time
*Oct 16 09:04:13.135: esmtp_client_work: socket 0 readable for first time
*Oct 16 09:04:13.135: (C)R: 220 quisp.cisco.com ESMTP Sendmail 8.8.4-Cisco.1/8.6.5 ready
at Wed, 27 Sep 2000 11:45:46 -0700 (PDT)
*Oct 16 09:04:13.135: (C)S: EHLO mmoip-c.cisco.com
*Oct 16 09:04:13.183: (C)R: 250-quisp.cisco.com Hello [172.22.95.16], pleased to meet you
*Oct 16 09:04:13.183: (C)R: 250-EXPN
*Oct 16 09:04:13.183: (C)R: 250-VERB
```

Related Commands

Command	Description
debug mta receive all	Displays output for all of the off-ramp client connections.
debug mta send rcpt-to	Displays output for a specific on-ramp SMTP client connection during an e-mail transmission.

debug mta send rcpt-to

To display output for a specific on-ramp SMTP client connection during an e-mail transmission, use the **debug mta send rcpt-to** EXEC command. Use the **no** form of this command to disable debugging output.

debug mta send rcpt-to *string*

[no] debug mta send rcpt-to *string*

Syntax Description

string	Specifies the e-mail address.

Defaults

Disabled

Command History

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

Examples

The following example shows debugging information displayed when the **debug mmoip send email** command has been enabled and the SMTP client is sending an e-mail message.

```
Router# debug mta send all
All email send debugging is on
Router# debug mmoip send email ilyau@company.com
Router# socket 0 attempting to connect to IP address 172.69.95.82
socket 0 readable for first time - let's try to read it
R:220 quisp.cisco.com ESMTP Sendmail 8.8.4-Cisco.1/8.6.5 ready at Tue, 6
Apr 1999 13:35:39 -0700 (PDT)
S:EHLO mmoip-c.cisco.com
R:250-quisp.cisco.com Hello [172.22.95.16], pleased to meet you
R:250-EXPN
R:250-VERB
R:250-8BITMIME
R:250-SIZE
R:250-DSN
R:250-ETRN
R:250-XUSR
R:250 HELP
S:MAIL FROM:<testing@> RET=HDRS
R:250 <testing@>... Sender ok
S:RCPT TO:<ilyau@cisco.com> NOTIFY=SUCCESS ORCPT=rfc822;testing@
R:250 <ilyau@cisco.com>... Recipient ok
R:354 Enter mail, end with "." on a line by itself
S:Received:(Cisco Powered Fax System) by mmoip-c.cisco.com for
<ilyau@cisco.com> (with Cisco NetWorks); Fri, 17 Oct 1997 14:54:27 +0800
S:To: <ilyau@cisco.com>
S:Message-ID:<000F1997145427146@mmoip-c.cisco.com>
S:Date:Fri, 17 Oct 1997 14:54:27 +0800
S:Subject:mmoip-c subject here
S:X-Mailer:IOS (tm) 5300 Software (C5300-IS-M)
S:MIME-Version:1.0
S:Content-Type:multipart/mixed;
S: boundary="yradnuoB=_000E1997145426826.mmoip-ccisco.com"
```

```
S:From:"Test User" <testing@>
S:--yradnuoB=_000E1997145426826.mmoip-ccisco.com
S:Content-ID:<00101997145427150@mmoip-c.cisco.com>
S:--yradnuoB=_000E1997145426826.mmoip-ccisco.com--
Sending terminating dot ...(socket=0)
S:.
R:250 NAA09092 Message accepted for delivery
S:QUIT
R:221 quisp.cisco.com closing connection
Freeing SMTP ctx at 0x6121D454
returned from work_routine, context freed
```

Related Commands

Command	Description
debug mta send all	Displays output for all of the on-ramp client connections.

debug ncia circuit

To display circuit-related information between the native client interface architecture (NCIA) server and client, use the **debug ncia circuit** privileged EXEC command. The **no** form of this command disables debugging output.

debug ncia circuit [**error** | **event** | **flow-control** | **state**]

no debug ncia circuit [**error** | **event** | **flow-control** | **state**]

Syntax Description

error	(Optional) Displays the error situation for each circuit.
event	(Optional) Displays the packets received and sent for each circuit.
flow-control	(Optional) Displays the flow control information for each circuit.
state	(Optional) Displays the state changes for each circuit.

Usage Guidelines

NCIA is an architecture developed by Cisco for accessing SNA applications. This architecture allows native SNA interfaces on hosts and clients to access TCP/IP backbones.

You cannot enable debugging output for a particular client or particular circuit.

Caution Do not enable the **debug ncia circuit** command during normal operation because this command generates a substantial amount of output messages and could slow down the router.

Examples

The following is sample output from the **debug ncia circuit error** command. In this example, the possible errors are displayed. The first error message indicates that the router is out of memory. The second message indicates that the router has an invalid circuit control block. The third message indicates that the router is out of memory. The remaining messages identify errors related to the finite state machine.

```
Router# debug ncia circuit error

NCIA: ncia_circuit_create memory allocation fail
NCIA: ncia_send_ndlc: invalid circuit control block
NCIA: send_ndlc: fail to get buffer for ndlc primitive xxx
NCIA: ncia circuit fsm: Invalid input
NCIA: ncia circuit fsm: Illegal state
NCIA: ncia circuit fsm: Illegal input
NCIA: ncia circuit fsm: Unexpected input
NCIA: ncia circuit fsm: Unknown error rtn code
```

The following is sample output from the **debug ncia circuit event** command. In this example, a session start-up sequence is displayed.

```
Router# debug ncia circuit event

NCIA(IN): Ver_Id: 0x81, MsgType: NDLC_START_DL, Len: 24, tmac: 4000.1060.1000,
          tsap: 4, csap 8, oid: 8A91E8, tid 0, lfs 16, ws 1
NCIA: create circuit: saddr 4000.1060.1000, ssap 4, daddr 4000.3000.0003, dsap 8 sid:
          8B09A8
NCIA: send NDLC_DL_STARTED to client 10.2.20.3 for ckt: 8B09A8
NCIA(OUT): Ver_Id: 0x81, MsgType: NDLC_DL_STARTED, Len: 2,4 tmac: 4000.1060.1000,
```

```
          tsap: 4, csap 8, oid: 8A91E8, tid 8B09A8, lfs 16, ws 1
NCIA(IN): Ver_Id: 0x81, MsgType: NDLC_XID_FRAME, Len: 12, sid: 8B09A8, FC 0x81
NCIA: send NDLC_XID_FRAME to client 10.2.20.3 for ckt: 8B09A8
NCIA(OUT): Ver_Id: 0x81, MsgType: NDLC_XID_FRAME, Len: 12, sid: 8A91E8, FC 0xC1
NCIA(IN): Ver_Id: 0x81, MsgType: NDLC_XID_FRAME, Len: 18, sid: 8B09A8, FC 0xC1
NCIA: send NDLC_CONTACT_STN to client 10.2.20.3 for ckt: 8B09A8
NCIA(OUT): Ver_Id: 0x81, MsgType: NDLC_CONTACT_STN, Len: 12, sid: 8A91E8, FC 0xC1
NCIA(IN): Ver_Id: 0x81, MsgType: NDLC_STN_CONTACTED, Len: 12, sid: 8B09A8, FC 0xC1
NCIA: send NDLC_INFO_FRAME to client 10.2.20.3 for ckt: 8B09A8
NCIA(OUT): Ver_Id: 0x81, MsgType: NDLC_INFO_FRAME, Len: 30, sid: 8A91E8, FC 0xC1
```

Table 134 describes the significant fields in the output.

Table 134 debug ncia circuit event Field Descriptions

Field	Description
IN	Incoming message from client.
OUT	Outgoing message to client.
Ver_Id	NDLC version ID.
MsgType	NDLC message type.
Len	NDLC message length.
tmac	Target MAC.
tsap	Target SAP.
csap	Client SAP.
oid	Origin ID.
tid	Target ID.
lfs	Largest frame size flag.
ws	Window size.
saddr	Source MAC address.
ssap	Source SAP.
daddr	Destination MAC address.
dsap	Destination SAP.
sid	Session ID.
FC	Flow control flag.

In the following messages, an NDLC_START_DL messages is received from a client. to start a data-link session:

```
NCIA(IN): Ver_Id: 0x81, MsgType: NDLC_START_DL, Len: 24, tmac: 4000.1060.1000,
          tsap: 4, csap 8, oid: 8A91E8, tid 0, lfs 16, ws 1
NCIA: create circuit: saddr 4000.1060.1000, ssap 4, daddr 4000.3000.0003, dsap 8 sid:
          8B09A8
```

The next two messages indicate that an NDLC_DL_STARTED message is sent to a client. The server informs the client that a data-the link session is started.

```
NCIA: send NDLC_DL_STARTED to client 10.2.20.3 for ckt: 8B09A8
NCIA(OUT): Ver_Id: 0x81, MsgType: NDLC_DL_STARTED, Len: 2,4 tmac: 4000.1060.1000,
          tsap: 4, csap 8, oid: 8A91E8, tid 8B09A8, lfs 16, ws 1
```

In the following two messages, an NDLC_XID_FRAME message is received from a client, and the client starts an XID exchange:

```
NCIA(IN): Ver_Id: 0x81, MsgType: NDLC_XID_FRAME, Len: 12, sid: 8B09A8, FC 0x81
NCIA: send NDLC_XID_FRAME to client 10.2.20.3 for ckt: 8B09A8
```

In the following two messages, an NDLC_XID_FRAME message is sent from a client, and an DLC_XID_FRAME message is received from a client:

```
NCIA(OUT): Ver_Id: 0x81, MsgType: NDLC_XID_FRAME, Len: 12, sid: 8A91E8, FC 0xC1
NCIA(IN): Ver_Id: 0x81, MsgType: NDLC_XID_FRAME, Len: 18, sid: 8B09A8, FC 0xC1
```

The next two messages show that an NDLC_CONTACT_STN message is sent to a client:

```
NCIA: send NDLC_CONTACT_STN to client 10.2.20.3 for ckt: 8B09A8
NCIA(OUT): Ver_Id: 0x81, MsgType: NDLC_CONTACT_STN, Len: 12, sid: 8A91E8, FC 0xC1
```

In the following message, an NDLC_STN_CONTACTED message is received from a client. The client informs the server that the station has been contacted.

```
NCIA(IN): Ver_Id: 0x81, MsgType: NDLC_STN_CONTACTED, Len: 12, sid: 8B09A8, FC 0xC1
```

In the last two messages, an NDLC_INFO_FRAME is sent to a client, and the server sends data to the client:

```
NCIA: send NDLC_INFO_FRAME to client 10.2.20.3 for ckt: 8B09A8
NCIA(OUT): Ver_Id: 0x81, MsgType: NDLC_INFO_FRAME, Len: 30, sid: 8A91E8, FC 0xC1
```

The following is sample output from the **debug ncia circuit flow-control** command. In this example, the flow control in a session startup sequence is displayed:

```
Router# debug ncia circuit flow-control

NCIA: no flow control in NDLC_DL_STARTED frame
NCIA: receive Increment Window Op for circuit 8ADE00
NCIA: ncia_flow_control_in FC 0x81, IW 1 GP 2 CW 2, Client IW 1 GP 0 CW 1
NCIA: grant client more packet by sending Repeat Window Op
NCIA: ncia_flow_control_out FC: 0xC1, IW 1 GP 2 CW 2, Client IW 1 GP 2 CW 2
NCIA: receive FCA for circuit 8ADE00
NCIA: receive Increment Window Op for circuit 8ADE00
NCIA: ncia_flow_control_in FC 0xC1, IW 1 GP 5 CW 3, Client IW 1 GP 2 CW 2
NCIA: grant client more packet by sending Repeat Window Op
NCIA: ncia_flow_control_out FC: 0xC1, IW 1 GP 5 CW 3, Client IW 1 GP 5 CW 3
NCIA: receive FCA for circuit 8ADE00
NCIA: receive Increment Window Op for circuit 8ADE00
NCIA: ncia_flow_control_in FC 0xC1, IW 1 GP 9 CW 4, Client IW 1 GP 5 CW 3
NCIA: grant client more packet by sending Repeat Window Op
NCIA: ncia_flow_control_out FC: 0xC1, IW 1 GP 8 CW 4, Client IW 1 GP 9 CW 4
NCIA: reduce ClientGrantPacket by 1 (Granted: 8)
NCIA: receive FCA for circuit 8ADE00
NCIA: receive Increment Window Op for circuit 8ADE00
```

Table 135 describes the significant fields shown in the display.

Table 135 debug ncia circuit flow-control Field Descriptions

Field	Description
IW	Initial window size.
GP	Granted packet number.
CW	Current window size.

The following is sample output from the **debug ncia circuit state** command. In this example, a session startup sequence is displayed:

```
Router# debug ncia circuit state

NCIA: pre-server fsm: event CONN_OPENED
NCIA: pre-server fsm: event NDLC_PRIMITIVES
NCIA: server event: WAN - STDL state: CLSOED
NCIA: ncia server fsm action 32
NCIA: circuit state: CLOSED -> START_DL_RCVD
NCIA: server event: DLU - TestStn.Rsp state: START_DL_RCVD
NCIA: ncia server fsm action 17
NCIA: circuit state: START_DL_RCVD -> DL_STARTED_SND
NCIA: pre-server fsm: event NDLC_PRIMITIVES
NCIA: server event: WAN - XID state: DL_STARTED_SND
NCIA: ncia server fsm action 33
NCIA: circuit state: DL_STARTED_SND -> DL_STARTED_SND
NCIA: server event: DLU - ReqOpnStn.Req state: DL_STARTED_SND
NCIA: ncia server fsm action 33
NCIA: circuit state: DL_STARTED_SND -> OPENED
NCIA: server event: DLU - Id.Rsp state: OPENED
NCIA: ncia server fsm action 11
NCIA: circuit state: OPENED -> OPENED
NCIA: pre-server fsm: event NDLC_PRIMITIVES
NCIA: server event: WAN - XID state: OPENED
NCIA: ncia server fsm action 33
NCIA: circuit state: OPENED -> OPENED
NCIA: server event: DLU - Connect.Req state: OPENED
NCIA: ncia server fsm action 6
NCIA: circuit state: OPENED -> CONNECT_PENDING
NCIA: pre-server fsm: event NDLC_PRIMITIVES
NCIA: server event: WAN - CONR state: CONNECT_PENDING
NCIA: ncia server fsm action 33 --> CLS_CONNECT_CNF sets NciaClsBusy
NCIA: circuit state: CONNECT_PENDING -> CONNECTED
NCIA: server event: DLU - Flow.Req (START) state: CONNECTED
NCIA: ncia server fsm action 25 --> unset NciaClsBusy
NCIA: circuit state: CONNECTED -> CONNECTED
NCIA: server event: DLU - Data.Rsp state: CONNECTED
NCIA: ncia server fsm action 8
NCIA: circuit state: CONNECTED -> CONNECTED
```

Table 136 describes the significant fields shown in the display.

Table 136 debug ncia circuit state Field Descriptions

Field	Description
WAN	Event from WAN (client).
DLU	Event from upstream module—dependent logical unit (DLU).
ADMIN	Administrative event.
TIMER	Timer event.

Related Commands

Command	Description
debug dmsp fax-to-doc	Enables debugging of DLSw+.
debug ncia client	Displays debug information for all NCIA client processing that occurs in the router.
debug ncia server	Displays debug information for the NCIA server and its upstream software modules.

debug ncia client

To display debug information for all native client interface architecture (NCIA) client processing that occurs in the router, use the **debug ncia client** privileged EXEC command. The **no** form of this command disables debugging output.

debug ncia client [*ip-address* | **error** [*ip-address*] | **event** [*ip-address*] | **message** [*ip-address*]]

no debug ncia client [*ip-address* | **error** [*ip-address*] | **event** [*ip-address*] | **message** [*ip-address*]]

Syntax Description

ip-address	(Optional) The remote client IP address.
error	(Optional) Triggers the recording of messages only when errors occur. The current state and event of an NCIA client are normally included in the message. If you do not specify an IP address, the error messages are logged for all active clients.
event	(Optional) Triggers the recording of messages that describe the current state and event—and sometimes the action that just completed—for the NCIA client. If you do not specify an IP address, the messages are logged for all active clients.
message	(Optional) Triggers the recording of messages that contain up to the first 32 bytes of data in a TCP packet sent to or received from an NCIA client. If you do not specify an IP address, the messages are logged for all active clients.

Usage Guidelines

NCIA is an architecture developed by Cisco for accessing SNA applications. This architecture allows native SNA interfaces on hosts and clients to access TCP/IP backbones.

Use the **debug ncia client error** command to see only certain error conditions that occur.

Use the **debug ncia client event** command to determine the sequences of activities that occur while a NCIA client is in different processing states.

Use the **debug ncia client message** command to see only the first 32 bytes of data in a TCP packet sent to or received from an NCIA client.

The **debug ncia client** command can be used in conjunction with the **debug ncia server** and **debug ncia circuit** commands to get a complete picture of NCIA activity.

Examples

The following is sample output from the **debug ncia circuit** command. Following the example is a description of each sample output message.

```
Router# debug ncia client

NCIA: Passive open 10.2.20.123(1088) -> 1973
NCIA: index for client hash queue is 27
NCIA: number of element in client hash queue 27 is 1
NCIA: event PASSIVE_OPEN, state NCIA_CLOSED for client 10.2.20.123
NCIA: Rcvd msg type NDLC_CAP_XCHG in tcp packet for client 10.2.20.123
NCIA: First 17 byte of data rcvd: 8112001100000000000000400050104080C
NCIA: Sent msg type NDLC_CAP_XCHG in tcp packet to client 10.2.20.123
NCIA: First 17 byte of data sent: 81120011100000001000040005010408OC
NCIA: event CAP_CMD_RCVD, state NCIA_CAP_WAIT, for client 10.2.20.123, cap xchg cmd sent
```

```
NCIA: Rcvd msg type NDLC_CAP_XCHG in tcp packet for client 10.2.20.123
NCIA: First 17 byte of data rcvd: 81120011100000001000000050104080C
NCIA: event CAP_RSP_RCVD, state NCIA_CAP_NEG for client 10.2.20.123

NCIA: Rcvd msg type NDLC_PEER_TEST_REQ in tcp packet for client 10.2.20.123
NCIA: First 4 byte of data rcvd: 811D0004
NCIA: event KEEPALIVE_RCVD, state NCIA_OPENED for client 10.2.20.123
NCIA: Sent msg type NDLC_PEER_TEST_RSP in tcp packet to client 10.2.20.123
NCIA: First 4 byte of data sent: 811E0004IA

NCIA: event TIME_OUT, state NCIA_OPENED, for client 10.2.20.123, keepalive_count = 0
NCIA: Sent msg type NDLC_PEER_TEST_REQ, in tcp packet to client 10.2.20.123
NCIA: First 4 byte of data sent: 811D0004
NCIA: Rcvd msg type NDLC_PEER_TEST_RSP in tcp packet for client 10.2.20.123
NCIA: First 4 byte of data rcvd: 811E0004
NCIA: event KEEPALIVE_RSP_RCVD, state NCIA_OPENED for client 10.2.20.123

NCIA: Error, event PASIVE_OPEN, state NCIA_OPENED, for client 10.2.20.123, should not have
occurred.
NCIA: Error, active_open for pre_client_fsm while client 10.2.20.123 is active or not
configured, registered.
```

Messages in lines 1 through 12 show the events that occur when a client connects to the router (the NCIA server). These messages show a passive_open process.

Messages in lines 13 to 17 show the events that occur when a TIME_OUT event is detected by a client PC workstation. The workstation sends an NDLC_PEER_TEST_REQ message to the NCIA server, and the router responds with an NDLC_PEER_TEST_RSP message.

Messages in lines 18 to 23 show the events that occur when a TIME_OUT event is detected by the router (the NCIA server). The router sends an NDLC_PEER_TEST_REQ message to the client PC workstation, and the PC responds with an NDLC_PEER_TEST_RSP message.

When you use the **debug ncia client message** command, the messages shown on lines 6, 8, 11, 14, 17, 20, and 22 are output in addition to other messages not shown in this example.

When you use the **debug ncia client error** command, the messages shown on lines 24 and 25 are output in addition to other messages not shown in this example.

Related Commands

Command	Description
debug ncia circuit	Displays debug information for all NCIA client processing that occurs in the router.
debug ncia server	Displays debug information for the NCIA server and its upstream software modules.

debug ncia server

To display debug information for the native client interface architecture (NCIA) server and its upstream software modules, use the **debug ncia server** privileged EXEC command. The **no** form of this command disables debugging output.

debug ncia server

no debug ncia server

Syntax Description

This command has no arguments or keywords.

Usage Guidelines

NCIA is an architecture developed by Cisco for accessing SNA applications. This architecture allows native SNA interfaces on hosts and clients to access TCP/IP backbones.

The **debug ncia server** command displays all Cisco Link Services (CLS) messages between the NCIA server and its upstream modules, such as data-link switching (DLSw) and downstream physical units (DSPUs). Use this command when a problem exists between the NCIA server and other software modules within the router.

You cannot enable debugging output for a particular client or particular circuit.

Examples

The following is sample output from the **debug ncia server** command. In this example, a session startup sequence is displayed. Following the example is a description of each group of sample output messages.

```
Router# debug ncia server

NCIA: send CLS_TEST_STN_IND to DLU
NCIA: Receive TestStn.Rsp
NCIA: send CLS_ID_STN_IND to DLU
NCIA: Receive ReqOpnStn.Req
NCIA: send CLS_REQ_OPNSTN_CNF to DLU
NCIA: Receive Id.Rsp
NCIA: send CLS_ID_IND to DLU
NCIA: Receive Connect.Req
NCIA: send CLS_CONNECT_CNF to DLU
NCIA: Receive Flow.Req
NCIA: Receive Data.Req
NCIA: send CLS_DATA_IND to DLU
NCIA: send CLS_DISC_IND to DLU
NCIA: Receive Disconnect.Rsp
```

In the following messages, the client is sending a test message to the host and the test message is received by the host:

```
NCIA: send CLS_TEST_STN_IND to DLU
NCIA: Receive TestStn.Rsp
```

In the next message, the server is sending an XID message to the host:

```
NCIA: send CLS_ID_STN_IND to DLU
```

In the next two messages, the host opens the station and the server responds:

```
NCIA: Receive ReqOpnStn.Req
NCIA: send CLS_REQ_OPNSTN_CNF to DLU
```

In the following two messages, the client is performing an XID exchange with the host:

```
NCIA: Receive Id.Rsp
NCIA: send CLS_ID_IND to DLU
```

In the next group of messages, the host attempts to establish a session with the client:

```
NCIA: Receive Connect.Req
NCIA: send CLS_CONNECT_CNF to DLU
NCIA: Receive Flow.Req
```

In the next two messages, the host sends data to the client:

```
NCIA: Receive Data.Req
NCIA: send CLS_DATA_IND to DLU
```

In the last two messages, the client closes the session:

```
NCIA: send CLS_DISC_IND to DLU
NCIA: Receive Disconnect.Rsp
```

Related Commands

Command	Description
debug dmsp fax-to-doc	Enables debugging of DLSw+.
debug ncia circuit	Displays circuit-related information between the NCIA server and client.
debug ncia client	Displays debug information for all NCIA client processing that occurs in the router.

debug netbios error

To display information about Network Basic Input/Output System (NetBIOS) protocol errors, use the **debug netbios error** privileged EXEC command. The **no** form of this command disables debugging output.

debug netbios error

no debug netbios error

Syntax Description

This command has no arguments or keywords.

Usage Guidelines

For complete information on the NetBIOS process, use the **debug netbios packet** command along with the **debug netbios error** command.

Examples

The following is sample output from the **debug netbios error** command. This example shows that an illegal packet has been received on the asynchronous interface.

```
Router# debug netbios error

Async1 nbf Bad packet
```

Related Commands

Command	Description
debug netbios-name-cache	Displays name caching activities on a router.
debug netbios packet	Displays general information about NetBIOS packets.

debug netbios-name-cache

To display name caching activities on a router, use the **debug netbios-name-cache** privileged EXEC command. The **no** form of this command disables debugging output.

debug netbios-name-cache

no debug netbios-name-cache

Syntax Description

This command has no arguments or keywords.

Usage Guidelines

Examine the display to diagnose problems in NetBIOS name caching.

Examples

The following is sample output from the **debug netbios-name-cache** command:

```
Router# debug netbios-name-cache

NETBIOS: L checking name ORINDA, vrn=0
NetBIOS name cache table corrupted at offset 13
NetBIOS name cache table corrupted at later offset, at location 13
NETBIOS: U chk name=ORINDA, addr=1000.4444.5555, idb=TR1, vrn=0, type=1
NETBIOS: U upd name=ORINDA,addr=1000.4444.5555,idb=TR1,vrn=0,type=1
NETBIOS: U add name=ORINDA,addr=1000.4444.5555,idb=TR1,vrn=0,type=1
NETBIOS: U no memory to add cache entry. name=ORINDA,addr=1000.4444.5555
NETBIOS: Invalid structure detected in netbios_name_cache_ager
NETBIOS: flushed name=ORINDA, addr=1000.4444.5555
NETBIOS: expired name=ORINDA, addr=1000.4444.5555
NETBIOS: removing entry. name=ORINDA,addr=1000.4444.5555,idb=TR1,vrn=0
NETBIOS: Tossing ADD_NAME/STATUS/NAME/ADD_GROUP frame
NETBIOS: Lookup Failed -- not in cache
NETBIOS: Lookup Worked, but split horizon failed
NETBIOS: Could not find RIF entry
NETBIOS: Cannot duplicate packet in netbios_name_cache_proxy
```

Note The sample display is a composite output. Debugging output that you actually see would not necessarily occur in this sequence.

Table 137 describes the significant fields shown in the display.

Table 137 debug netbios-name-cache Field Descriptions

Field	Description
NETBIOS	NetBIOS name caching debugging output.
L, U	L means lookup; U means update.
addr=1000.4444.5555	MAC address of machine being looked up in NetBIOS name cache.
idb=TR1	Indicates that the name of machine was learned from Token Ring interface number 1; idb is into interface data block.

Table 137 debug netbios-name-cache Field Descriptions (continued)

Field	Description
vrn=0	Packet comes from virtual ring number 0. This packet actually comes from a real Token Ring interface, because virtual ring number 0 is not valid.
type=1	Indicates the way that the router learned about the specified machine. The possible values are as follows: • 1 - Learned from traffic • 2 - Learned from a remote peer • 4, 8 - Statically entered via the configuration of the router

With the first line of output, the router declares that it has examined the NetBIOS name cache table for the machine name ORINDA and that the packet that prompted the lookup came from virtual ring 0. In this case, this packet comes from a real interface—virtual ring number 0 is not valid.

```
NETBIOS: L checking name ORINDA, vrn=0
```

The following two lines indicate that an invalid NetBIOS entry exists and that the corrupted memory was detected. The invalid memory will be removed from the table; no action is needed.

```
NetBIOS name cache table corrupted at offset 13
NetBIOS name cache table corrupted at later offset, at location 13
```

The following line indicates that the router attempted to check the NetBIOS cache table for the name ORINDA with MAC address 1000.4444.5555. This name was obtained from Token Ring interface 1. The type field indicates that the name was learned from traffic.

```
NETBIOS: U chk name=ORINDA, addr=1000.4444.5555, idb=TR1, vrn=0, type=1
```

The following line indicates that the NetBIOS name ORINDA is in the name cache table and was updated to the current value:

```
NETBIOS: U upd name=ORINDA,addr=1000.4444.5555,idb=TR1,vrn=0,type=1
```

The following line indicates that the NetBIOS name ORINDA is not in the table and must be added to the table:

```
NETBIOS: U add name=ORINDA,addr=1000.4444.5555,idb=TR1,vrn=0,type=1
```

The following line indicates that there was insufficient cache buffer space when the router tried to add this name:

```
NETBIOS: U no memory to add cache entry. name=ORINDA,addr=1000.4444.5555
```

The following line indicates that the NetBIOS ager detects an invalid memory in the cache. The router clears the entry; no action is needed.

```
NETBIOS: Invalid structure detected in netbios_name_cache_ager
```

The following line indicates that the entry for ORINDA was flushed from the cache table:

```
NETBIOS: flushed name=ORINDA, addr=1000.4444.5555
```

The following line indicates that the entry for ORINDA timed out and was flushed from the cache table:

```
NETBIOS: expired name=ORINDA, addr=1000.4444.5555
```

The following line indicates that the router removed the ORINDA entry from its cache table:

```
NETBIOS: removing entry. name=ORINDA,addr=1000.4444.5555,idb=TR1,vrn=0
```

The following line indicates that the router discarded a NetBIOS packet of type ADD_NAME, STATUS, NAME_QUERY, or ADD_GROUP. These packets are discarded when multiple copies of one of these packet types are detected during a certain period of time.

```
NETBIOS: Tossing ADD_NAME/STATUS/NAME/ADD_GROUP frame
```

The following line indicates that the system could not find a NetBIOS name in the cache:

```
NETBIOS: Lookup Failed -- not in cache
```

The following line indicates that the system found the destination NetBIOS name in the cache, but located on the same ring from which the packet came. The router will drop this packet because the packet should not leave this ring.

```
NETBIOS: Lookup Worked, but split horizon failed
```

The following line indicates that the system found the NetBIOS name in the cache, but the router could not find the corresponding RIF. The packet will be sent as a broadcast frame.

```
NETBIOS: Could not find RIF entry
```

The following line indicates that no buffer was available to create a NetBIOS name cache proxy. A proxy will not be created for the packet, which will be forwarded as a broadcast frame.

```
NETBIOS: Cannot duplicate packet in netbios_name_cache_proxy
```

Related Commands

Command	Description
debug netbios error	Displays information about NetBIOS protocol errors.
debug netbios packet	Displays general information about NetBIOS packets.

debug netbios packet

To display general information about NetBIOS packets, use the **debug netbios packet** privileged EXEC command. The **no** form of this command disables debugging output.

debug netbios packet

no debug netbios packet

Syntax Description

This command has no arguments or keywords.

Usage Guidelines

For complete information on the NetBIOS process, use the **debug netbios error** command along with the **debug netbios packet** command.

Examples

The following is sample output from the **debug netbios packet** and **debug netbios error** commands. This example shows the LLC header for an asynchronous interface followed by the NetBIOS information. For additional information on the NetBIOS fields, refer to *IBM LAN Technical Reference IEEE 802.2*.

```
Router# debug netbios packet

Async1 (i) U-format UI C_R=0x0
(i) NETBIOS_ADD_NAME_QUERY
 Resp_correlator= 0x6F 0x0
 Src name=CS-NT-1

Async1 (i) U-format UI C_R=0x0
(i) NETBIOS_ADD_GROUP_QUERY
 Resp_correlator= 0x6F 0x0
 Src name=COMMSERVER-WG

Async1 (i) U-format UI C_R=0x0
(i) NETBIOS_ADD_NAME_QUERY
 Resp_correlator= 0x6F 0x0
 Src name=CS-NT-1

Ethernet0 (i) U-format UI C_R=0x0
(i) NETBIOS_DATAGRAM
 Length= 0x2C 0x0
 Dest name=COMMSERVER-WG
 Src name=CS-NT-3
```

Related Commands

Command	Description
debug netbios error	Displays information about NetBIOS protocol errors.
debug netbios-name-cache	Displays name caching activities on a router.

debug nhrp

To display information about Next Hop Resolution Protocol (NHRP) activity, use the **debug nhrp** privileged EXEC command. The **no** form of this command disables debugging output.

debug nhrp

no debug nhrp

Syntax Description

This command has no arguments or keywords.

Usage Guidelines

Use this command when some nodes on a TCP/IP or IPX network are not responding. Output from the command shows whether the router is sending or receiving NHRP packets.

Examples

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

```
Router# debug nhrp

NHRP: Cache update 172.19.145.57 None
NHRP: Sent request src 172.19.145.56 dst 255.255.255.255
NHRP M: id 0 src 172.19.145.56 dst 172.19.145.57
NHRP: Encapsulation succeeded. MAC addr ffff.ffff.ffff.
NHRP: O 86 bytes out Ethernet1 dest 255.255.255.255
NHRP: Recv reply Size 64
NHRP M: id 0 src 172.19.145.56 dst 172.19.145.57
NHRP: Cache update 172.19.145.57 0000.0c14.59d3.
```

Table 138 describes the significant fields shown in the display.

Table 138 debug nhrp Field Descriptions

Field	Descriptions
NHRP and NHRP M	NHRP debugging output and mandatory header debugging output.
Cache update	NHRP cache is being revised.
Sent request src dst	NHRP request packet was sent from the specified source address. NHRP packet was sent to the specified destination address.
id	Sequence number of the packet.
src	Sequence number of the source address.
dst	Sequence number of the destination address.
Encapsulation succeeded.	NHRP packet was encapsulated.
MAC addr	Link-layer address used as the destination address for the NHRP packet.

Table 138 debug nhrp Field Descriptions (continued)

Field	Descriptions
O 86 bytes out Ethernet1 dest	Size of the NHRP packet (in this case, the output was 86 bytes). Interface that the packet was sent out on, and the network-layer destination address.
Recv reply Size	Indicates receipt of an NHRP reply packet and the size of the packet excluding the link-layer header.

Related Commands

Command	Description
debug nhrp options	Displays information about NHRP option processing.
debug nhrp packet	Displays a dump of NHRP packets.

debug nhrp extension

To display the extensions portion of a NHRP packet, use the **debug nhrp extension** privileged EXEC command. The **no** form of this command disables debugging output.

debug nhrp extension

no debug nhrp extension

Syntax Description

This command has no arguments or keywords.

Examples

The following is sample output from the **debug nhrp extension** command:

```
Router# debug nhrp extension

NHRP extension processing debugging is on
Router#
Forward Transit NHS Record Extension(4):
 (C-1) code: no error(0)
       prefix: 0, mtu: 9180, hd_time: 7200
       addr_len: 20(NSAP), subaddr_len: 0(NSAP), proto_len: 4, pref: 0
       client NBMA: 47.0091810000000002ba08e101.525354555354.01
       client protocol: 135.206.58.54
Reverse Transit NHS Record Extension(5):
Responder Address Extension(3):
 (C) code: no error(0)
       prefix: 0, mtu: 9180, hd_time: 7200
       addr_len: 20(NSAP), subaddr_len: 0(NSAP), proto_len: 4, pref: 0
       client NBMA: 47.0091810000000002ba08e101.525354555355.01
       client protocol: 135.206.58.55
Forward Transit NHS Record Extension(4):
 (C-1) code: no error(0)
       prefix: 0, mtu: 9180, hd_time: 7200
       addr_len: 20(NSAP), subaddr_len: 0(NSAP), proto_len: 4, pref: 0
       client NBMA: 47.0091810000000002ba08e101.525354555354.01
       client protocol: 135.206.58.54
Reverse Transit NHS Record Extension(5):
Responder Address Extension(3):
Forward Transit NHS Record Extension(4):
Reverse Transit NHS Record Extension(5):
```

debug nhrp options

To display information about NHRP option processing, use the **debug nhrp options** privileged EXEC command. The **no** form of this command disables debugging output.

debug nhrp options

no debug nhrp options

Syntax Description

This command has no arguments or keywords.

Usage Guidelines

Use this command to show you whether there are problems or error situations with NHRP option processing (for example, unknown options).

Examples

The following is sample output from the **debug nhrp options** command:

```
Router# debug nhrp options

NHRP-OPT: MASK 4
NHRP-OPT-MASK: FFFFFFFF
NHRP-OPT: NETID 4
NHRP-OPT: RESPONDER 4
NHRP-OPT: RECORD 0
NHRP-OPT: RRECORD 0
```

Table 139 describes the significant fields shown in the display.

Table 139 debug nhrp options Field Descriptions

Field	Descriptions
NHRP-OPT	NHRP options debugging output.
MASK 4	Number of bytes of information in the destination prefix option.
NHRP-OPT-MASK	Contents of the destination prefix option.
NETID	Number of bytes of information in the subnetwork identifier option.
RESPONDER	Number of bytes of information in the responder address option.
RECORD	Forward record option.
RRECORD	Reverse record option.

Related Commands

Command	Description
debug nhrp	Displays information about NHRP activity.
debug nhrp packet	Displays a dump of NHRP packets.

debug nhrp packet

To display a dump of NHRP packets, use the **debug nhrp packet** privileged EXEC command. The **no** form of this command disables debugging output.

debug nhrp packet

no debug nhrp packet

Syntax Description

This command has no arguments or keywords.

Examples

The following is sample output from the **debug nhrp packet** command:

```
Router# debug nhrp packet

NHRP activity debugging is on
Router#
NHRP: Send Purge Request via ATM3/0.1, packet size: 72
 src: 135.206.58.55, dst: 135.206.58.56
 (F) afn: NSAP(3), type: IP(800), hop: 255, ver: 1
     shtl: 20(NSAP), sstl: 0(NSAP)
 (M) flags: "reply required", reqid: 2
     src NBMA: 47.0091810000000002ba08e101.525354555355.01
     src protocol: 135.206.58.55, dst protocol: 135.206.58.56
 (C-1) code: no error(0)
       prefix: 0, mtu: 9180, hd_time: 0
       addr_len: 0(NSAP), subaddr_len: 0(NSAP), proto_len: 4, pref: 0
       client protocol: 135.206.58.130
NHRP: Receive Purge Reply via ATM3/0.1, packet size: 72
 (F) afn: NSAP(3), type: IP(800), hop: 254, ver: 1
     shtl: 20(NSAP), sstl: 0(NSAP)
 (M) flags: "reply required", reqid: 2
     src NBMA: 47.0091810000000002ba08e101.525354555355.01
     src protocol: 135.206.58.55, dst protocol: 135.206.58.56
 (C-1) code: no error(0)
       prefix: 0, mtu: 9180, hd_time: 0
       addr_len: 0(NSAP), subaddr_len: 0(NSAP), proto_len: 4, pref: 0
       client protocol: 135.206.58.130
```

debug nhrp rate

To display information about NHRP traffic rate limits, use the **debug nhrp rate** privileged EXEC command. The **no** form of this command disables debugging output.

debug nhrp rate

no debug nhrp rate

Syntax Description

This command has no arguments or keywords.

Usage Guidelines

Use this command to verify that the traffic is consistent with the setting of the NHRP commands (such as **ip nhrp use** and **ip max-send** commands).

Examples

The following is sample output from the **debug nhrp rate** command:

```
Router# debug nhrp rate

NHRP-RATE: Sending initial request
NHRP-RATE: Retransmitting request (retrans ivl 2)
NHRP-RATE: Retransmitting request (retrans ivl 4)
NHRP-RATE: Ethernet1: Used 3
```

Table 140 describes the significant fields shown in the display.

Table 140 debug nhrp rate Field Descriptions

Field	Descriptions
NHRP-RATE	NHRP rate debugging output.
Sending initial request	First time an attempt was made to send an NHRP packet to a particular destination.
Retransmitting request	Indicates that the NHRP packet was re-sent, and shows the time interval (in seconds) to wait before the NHRP packet is re-sent again.
Ethernet1:	Interface over which the NHRP packet was sent.
Used 3	Number of packets sent out of the default maximum five (in this case, three were sent).

Related Commands

Command	Description
debug nhrp	Displays information about NHRP activity.
debug nhrp options	Displays information about NHRP option processing

debug ntp

To display debug messages for Network Time Protocol (NTP) features, use the **debug ntp** command. To stop the output of ntp debugging messages, use the **no** form of this command.

debug ntp {**adjust** | **authentication** | **events** | **loopfilter** | **packets** | **params** | **refclock** | **select** | **sync** | **validity**}

no debug ntp {**adjust** | **authentication** | **events** | **loopfilter** | **packets** | **params** | **refclock** | **select** | **sync** | **validity**}

Syntax Description

adjust	Displays debugging information on NTP clock adjustments.
authentication	Displays debugging information on NTP authentication.
events	Displays debugging information on NTP events.
loopfilter	Displays debugging information on NTP loop filters.
packets	Displays debugging information on NTP packets.
params	Displays debugging information on NTP clock parameters.
refclock	Displays debugging information on NTP reference clocks.
select	Displays debugging information on NTP clock selection.
sync	Displays debugging information on NTP clock synchronization.
validity	Displays debugging information on NTP peer clock validity.

Defaults

Debug commands are disabled by default.

Command History

Release	Modification
12.0 T	This command was introduced in a release prior to Cisco IOS Release 12.1.

Related Commands

Command	Description
ntp refclock	Configures an external clock source for use with NTP services.

debug oam

To display operation and maintenance (OAM) events, use the **debug oam** privileged EXEC command. The **no** form of this command disables debugging output.

debug oam

no debug oam

Syntax Description

This command has no arguments or keywords.

Examples

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

```
Router# debug oam

4/0(O): VCD:0x0 DM:0x300 *OAM Cell* Length:0x39
0000 0300 0070 007A 0018 0100 0000 05FF FFFF FFFF FFFF FFFF FFFF FFFF
FFFF FFFF FFFF FFFF FF6A 6A6A 6A6A 6A6A 6A6A 6A6A 6A6A 6A6A 6A00 0000
```

Table 141 describes the significant fields in the display.

Table 141 debug oam Field Descriptions

Field	Description
0000	VCD Special OAM indicator.
0300	Descriptor MODE bits for the AIP.
0	GFC (4 bits).
07	VPI (8 bits).
0007	VCI (16 bits).
A	Payload type field (PTI) (4 bits).
00	Header Error Correction (8 bits).
1	OAM Fault mangement cell (4 bits).
8	OAM LOOPBACK indicator (4 bits).
01	Loopback indicator value, always 1 (8 bits).
00000005	Loopback unique ID, sequence number (32 bits).
FF6A	Fs and 6A required in the remaining cell, per UNI3.0.

debug packet

To display per-packet debugging output, use the **debug packet** privileged EXEC command. The **no** form of this command disables debugging output.

debug packet [**interface** *number* [**vcd** *vcd-number*] | **vc** *vpi/vci* | *vc-name*]

no debug packet [**interface** *number* [**vcd** *vcd-number*] | **vc** *vpi/vci* | *vc-name*]

Syntax Description

interface *number*	(Optional) interface or subinterface number.
vcd *vcd-number*	(Optional) Number of the virtual circuit designator (VCD).
vc *vpi/vci*	(Optional) VPI and VCI numbers of the VC.
vc-name	(Optional) Name of the PVC or SVC.

Usage Guidelines

The **debug packet** command displays all process-level packets for both outbound and inbound packets. This command is useful for determining whether packets are being received and sent correctly. The output reports information online when a packet is received or a transmission is attempted.

For sent packets, the information is displayed only after the protocol data unit (PDU) is entirely encapsulated and a next hop VC is found. If information is not displayed, the address translation probably failed during encapsulation. When a next hop VC is found, the packet is displayed exactly as it will be presented on the wire. Having a display indicates that the packets are properly encapsulated for transmission.

For received packets, information is displayed for all incoming frames. The display can show whether the sending station properly encapsulates the frames. Because all incoming frames are displayed, this information is useful when performing back-to-back testing and corrupted frames cannot be dropped by an intermediary switch.

The **debug packet** command also displays the initial bytes of the actual PDU in hexadecimal. This information can be decoded only by qualified support or engineering personnel.

Caution Because the **debug packet** command generates a substantial amount of output for every packet processed, use it only when traffic on the network is low, so other activity on the system is not adversely affected.

Examples

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

```
Router# debug packet

2/0.5(I): VCD:0x9 VCI:0x23 Type:0x0 SAP:AAAA CTL:03 OUI:000000 TYPE:0800 Length0x70
4500 002E 0000 0000 0209 92ED 836C A26E FFFF FFFF 1108 006D 0001 0000 0000
A5CC 6CA2 0000 000A 0000 6411 76FF 0100 6C08 00FF FFFF 0003 E805 DCFF 0105
```

Table 142 describes the significant fields in the display.

Table 142 debug packet Field Descriptions

Field	Description
2/0.5	Indicates the subinterface that generated this packet.
(I)	Indicates a receive packet. (O) indicates an output packet.
VCD: 0x*n*	Indicates the virtual circuit associated with this packet, where *n* is some value.
DM: 0x*nnnn*	Indicates the descriptor mode bits on output only, where *nnnn* is a hexadecimal value.
TYPE:*n*	Displays the encapsulation type for this packet.
Length:*n*	Displays the total length of the packet including the headers.

The following two lines of output are the binary data, which are the contents of the protocol PDU before encapsulation:

```
4500 002E 0000 0000 0209 92ED 836C A26E FFFF FFFF 1108 006D 0001 0000 0000
A5CC 6CA2 0000 000A 0000 6411 76FF 0100 6C08 00FF FFFF 0003 E805 DCFF 0105
```

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

```
Router# debug packet

Ethernet0: Unknown ARPA, src 0000.0c00.6fa4, dst ffff.ffff.ffff, type 0x0a0
data 00000c00f23a00000c00ab45, len 60
Serial3: Unknown HDLC, size 64, type 0xaaaa, flags 0x0F00
Serial2: Unknown PPP, size 128
Serial7: Unknown FRAME-RELAY, size 174, type 0x5865, DLCI 7a
Serial0: compressed TCP/IP packet dropped
```

Table 143 describes the significant fields shown in the display.

Table 143 debug packet Field Descriptions

Field	Description
Ethernet0	Name of the Ethernet interface that received the packet.
Unknown	Network could not classify this packet. Examples include packets with unknown link types.

Table 143 debug packet Field Descriptions (continued)

Field	Description
ARPA	Packet uses ARPA-style encapsulation. Possible encapsulation styles vary depending on the media command mode (MCM) and encapsulation style. Ethernet (MCM)—Encapsulation Style: • APOLLO • ARP • ETHERTALK • ISO1 • ISO3 • LLC2 • NOVELL-ETHER • SNAP
	FDDI (MCM)—Encapsulation Style: • APOLLO • ISO1 • ISO3 • LLC2 • SNAP Frame Relay—Encapsulation Style: • BRIDGE • FRAME-RELAY

Table 143 debug packet Field Descriptions (continued)

Field	Description
	Serial (MCM)—Encapsulation Style: • BFEX25 • BRIDGE • DDN-X25 • DDNX25-DCE • ETHERTALK • FRAME-RELAY • HDLC • HDH • LAPB • LAPBDCE • MULTI-LAPB • PPP • SDLC-PRIMARY • SDLC-SECONDARY • SLIP • SMDS • STUN • X25 • X25-DCE
	Token Ring (MCM)—Encapsulation Style: • 3COM-TR • ISO1 • ISO3 • MAC • LLC2 • NOVELL-TR • SNAP • VINES-TR
src 0000.0c00.6fa4	MAC address of the node generating the packet.
dst.ffff.ffff.ffff	MAC address of the destination node for the packet.
type 0x0a0	Packet type.
data...	First 12 bytes of the datagram following the MAC header.
len 60	Length of the message (in bytes) that the interface received from the wire.
size 64	Length of the message (in bytes) that the interface received from the wire. Equivalent to the len field.

Table 143 debug packet Field Descriptions (continued)

Field	Description
flags 0x0F00	HDLC or PP flags field.
DLCI 7a	The DLCI number on Frame Relay.
compressed TCP/IP packet dropped	TCP header compression is enabled on an interface and the packet is not HDLC or X25.

debug pots

To display information on the telephone interfaces, use the **debug pots** privileged EXEC command. Use the **no** form of this command to disable debugging output.

debug pots {**driver** | **csm**} [**1** | **2**]

no debug pots {**driver** | **csm**} [**1** | **2**]

Syntax Description

driver	Displays driver debug information.
csm	Displays CSM debug information.
1	(Optional) Displays information for telephone port 1 only.
2	(Optional) Displays information for telephone port 2 only.

Usage Guidelines

The **debug pots** command displays driver and CSM debug information for telephone ports 1 and 2.

Examples

The following is a sample display from the **debug pots driver 1** command. This sample display indicates that the telephone port driver is not receiving caller ID information from the ISDN line. Therefore, the analog caller ID device attached to the telephone port does not display caller ID information.

```
Router# debug pots driver 1

00:01:51:POTS DRIVER port=1 activate ringer: cadence=0 callerId=Unknown
00:01:51:POTS DRIVER port=1 state=Idle drv_event=RING_EVENT
00:01:51:POTS DRIVER port=1 enter_ringing
00:01:51:POTS DRIVER port=1 cmd=19
00:01:51:POTS DRIVER port=1 activate disconnect
00:01:51:POTS DRIVER port=1 state=Ringing drv_event=DISCONNECT_EVENT
00:01:51:POTS DRIVER port=1 cmd=1A
00:01:51:POTS DRIVER port=1 enter_idle
00:01:51:POTS DRIVER port=1 ts connect: 0 0
00:01:51:POTS DRIVER port=1 cmd=D
00:01:51:POTS DRIVER port=1 report onhook
00:01:51:POTS DRIVER port=1 activate tone=SILENCE_TONE
00:01:51:POTS DRIVER port=1 state=Idle drv_event=TONE_EVENT
00:01:51:POTS DRIVER port=1 activate tone=SILENCE_TONE
00:01:51:POTS DRIVER port=1 state=Idle drv_event=TONE_EVENT
00:01:53:POTS DRIVER port=1 activate ringer: cadence=0 callerId=Unknown
00:01:53:POTS DRIVER port=1 state=Idle drv_event=RING_EVENT
00:01:53:POTS DRIVER port=1 enter_ringing
00:01:53:POTS DRIVER port=1 cmd=19
00:01:55:POTS DRIVER port=1 cmd=1A
00:02:49:POTS DRIVER port=1 state=Ringing drv_event=OFFHOOK_EVENT
00:02:49:POTS DRIVER port=1 cmd=1A
00:02:49:POTS DRIVER port=1 enter_suspend
00:02:49:POTS DRIVER port=1 cmd=A
00:02:49:POTS DRIVER port=1 report offhook
00:02:49:POTS DRIVER port=1 activate connect: endpt=1 calltype=TWO_PARTY_CALL
00:02:49:POTS DRIVER port=1 state=Suspend drv_event=CONNECT_EVENT
00:02:49:POTS DRIVER port=1 enter_connect: endpt=1 calltype=0
00:02:49:POTS DRIVER port=1 cmd=A
00:02:49:POTS DRIVER port=1 ts connect: 1 0
00:02:49:POTS DRIVER port=1 activate connect: endpt=1 calltype=TWO_PARTY_CALL
```

```
00:02:49:POTS DRIVER port=1 state=Connect drv_event=CONNECT_EVENT
00:02:49:POTS DRIVER port=1 enter_connect: endpt=1 calltype=0
00:02:49:POTS DRIVER port=1 cmd=A
00:02:49:POTS DRIVER port=1 ts connect: 1 0
00:02:55:POTS DRIVER port=1 state=Connect drv_event=ONHOOK_EVENT
00:02:55:POTS DRIVER port=1 enter_idle
00:02:55:POTS DRIVER port=1 ts connect: 0 0
00:02:55:POTS DRIVER port=1 cmd=D
00:02:55:POTS DRIVER port=1 report onhook
00:02:55:POTS DRIVER port=1 activate tone=SILENCE_TONE
00:02:55:POTS DRIVER port=1 state=Idle drv_event=TONE_EVENT
00:02:55:POTS DRIVER port=1 activate tone=SILENCE_TONE
00:02:55:POTS DRIVER port=1 state=Idle drv_event=TONE_EVENT
```

The following is sample display from the **debug pots csm 1** command. This sample display indicates that a dial peer contains an invalid destination pattern (555-1111).

```
Router# debug pots csm 1

01:57:28:EVENT_FROM_ISDN:dchanidb=0x66CB38, call_id=0x11, ces=0x2 bchan=0x0, event=0x1,
cause=0x0
01:57:28:Dial peer not found, route call to port 1
01:57:28:CSM_PROC_IDLE:CSM_EVENT_ISDN_CALL, call_id=0x11, port=1
01:57:28:Calling number '5551111'
01:57:40:CSM_PROC_RINGING:CSM_EVENT_VDEV_OFFHOOK, call_id=0x11, port=1
01:57:40:EVENT_FROM_ISDN:dchan_idb=0x66CB38, call_id=0x11, ces=0x2 bchan=0x0, event=0x4,
cause=0x0
01:57:40:CSM_PROC_CONNECTING:CSM_EVENT_ISDN_CONNECTED, call_id=0x11, port=1
01:57:47:CSM_PROC_CONNECTING:CSM_EVENT_VDEV_ONHOOK, call_id=0x11, port=1
01:57:201863503872: %ISDN-6-DISCONNECT:Interface BRI0:1 disconnected from unknown, call
lasted 5485 seconds
01:57:47: %ISDN-6-DISCONNECT:Interface BRI0:1 disconnected from unknown, call lasted 5485
seconds
01:57:47:EVENT_FROM_ISDN:dchan _idb=0x66CB38, call_id=0x11, ces=0x2 bchan=0xFFFFFFFF,
event=0x0, cause=0x1
01:57:47:CSM_PROC_NEAR_END_DISCONNECT:CSM_
```

debug pots csm

To activate events from which an application can determine and display the status and progress of calls to and from POTS ports, use the **debug pots csm** EXEC command.

debug pots csm

Syntax Description

This command has no arguments or keywords.

Command Modes

EXEC

Command History

Release	Modification
12.1.(2)XF	This command was introduced on the Cisco 800 series routers.

Usage Guidelines

To see debug messages, enter the **logging console** global configuration mode command as follows:

```
router(config)# logging console

router(config)# exit
```

Debug messages are displayed in one of two formats that are relevant to the POTS dial feature:

```
hh:mm:ss: CSM_STATE: CSM_EVENT, call id = ??, port = ?
```

or

```
hh:mm:ss: EVENT_FROM_ISDN:dchan_idb=0x???????, call_id=0x????, ces=? bchan=0x????????,
event=0x?, cause=0x??
```

Table 144 describes the significant fields shown in the display.

Table 144 debug pots csm Field Descriptions:

Command Elements	Description
hh:mm:ss	Timestamp (in hours, minutes, and seconds).
CSM_STATE	One of the call CSM states listed in Table 145.
CSM_EVENT	One of the CSM events listed in Table 146.
call id	Hexadecimal value from 0x00 to 0xFF.
port	Telephone port 1 or 2.
EVENT_FROM_ISDN	A CSM event. Table 146 shows a list of CSM events.
dchan_idb	Internal data structure address.
ces	Connection end point suffix used by ISDN.
bchan	Channel used by the call. A value of 0xFFFFFFFF indicates that a channel is not assigned.

Command Elements	Description
event	A hexadecimal value that is translated into a CSM event. Table 147 shows a list of events and the corresponding CSM events.
cause	A hexadecimal value that is given to call-progressing events. Table 148 shows a list of cause values and definitions.

Table 145 shows the values for CSM states.

Table 145 CSM States

CSM State	Description
CSM_IDLE_STATE	Telephone on the hook.
CSM_RINGING	Telephone ringing.
CSM_SETUP	Setup for outgoing call in progress.
CSM_DIALING	Dialing number of outgoing call.
CSM_IVR_DIALING	Interactive voice response (IVR) for Japanese telephone dialing.
CSM_CONNECTING	Waiting for carrier to connect the call.
CSM_CONNECTED	Call connected.
CSM_DISCONNECTING	Waiting for carrier to disconnect the call.
CSM_NEAR_END_DISCONNECTING	Waiting for carrier to disconnect the call.
CSM_HARD_HOLD	Call on hard hold.
CSM_CONSULTATION_HOLD	Call on consultation hold.
CSM_WAIT_FOR_HOLD	Waiting for carrier to put call on hard hold.
CSM_WAIT_FOR_CONSULTATION_HOLD	Waiting for carrier to put call on consultation hold.
CSM_CONFERENCE	Waiting for carrier to complete call conference.
CSM_TRANSFER	Waiting for carrier to transfer call.
CSM_APPLIC_DIALING	Call initiated from Cisco IOS CLI.

Table 146 shows the values for CSM events.

Table 146 CSM Events

CSM Events	Description
CSM_EVENT_INTER_DIGIT_TIMEOUT	Time waiting for dial digits has expired.
CSM_EVENT_TIMEOUT	Near- or far-end disconnect timeout.
CSM_EVENT_ISDN_CALL	Incoming call.
CSM_EVENT_ISDN_CONNECTED	Call connected.
CSM_EVENT_ISDN_DISCONNECT	Far end disconnected.
CSM_EVENT_ISDN_DISCONNECTED	Call disconnected.
CSM_EVENT_ISDN_SETUP	Outgoing call requested.

Table 146 CSM Events (continued)

CSM Events	Description
CSM_EVENT_ISDN_SETUP_ACK	Outgoing call accepted.
CSM_EVENT_ISDN_PROC	Call proceeding and dialing completed.
CSM_EVENT_ISDN_CALL_PROGRESSING	Call being received in band tone.
CSM_EVENT_ISDN_HARD_HOLD	Call on hard hold.
CSM_EVENT_ISDN_HARD_HOLD_REJ	Hold attempt rejected.
CSM_EVENT_ISDN_CHOLD	Call on consultation hold.
CSM_EVENT_ISDN_CHOLD_REJ	Consultation hold attempt rejected.
CSM_EVENT_ISDN_RETRIEVED	Call retrieved.
CSM_EVENT_ISDN_RETRIEVE_REJ	Call retrieval attempt rejected.
CSM_EVENT_ISDN_TRANSFERRED	Call transferred.
CSM_EVENT_ISDN_TRANSFER_REJ	Call transfer attempt rejected.
CSM_EVENT_ISDN_CONFERENCE	Call conference started.
CSM_EVENT_ISDN_CONFERENCE_REJ	Call conference attempt rejected.
CSM_EVENT_ISDN_IF_DOWN	ISDN interface down.
CSM_EVENT_ISDN_INFORMATION	ISDN information element received (used by NTT IVR application).
CSM_EVENT_VDEV_OFFHOOK	Telephone off the hook.
CSM_EVENT_VDEV_ONHOOK	Telephone on the hook.
CSM_EVENT_VDEV_FLASHHOOK	Telephone hook switch has flashed.
CSM_EVENT_VDEV_DIGIT	DTMF digit has been detected.
CSM_EVENT_VDEV_APPLICATION_CALL	Call initiated from Cisco IOS CLI.

Table 147 shows the values for events that are translated into CSM events.

Table 147 Event Values

Hexadecimal Value	Event	CSM Event
0x0	DEV_IDLE	CSM_EVENT_ISDN_DISCONNECTED
0x1	DEV_INCALL	CSM_EVENT_ISDN_CALL
0x2	DEV_SETUP_ACK	CSM_EVENT_ISDN_SETUP_ACK
0x3	DEV_CALL_PROC	CSM_EVENT_ISDN_PROC
0x4	DEV_CONNECTED	CSM_EVENT_ISDN_CONNECTED
0x5	DEV_CALL_PROGRESSING	CSM_EVENT_ISDN_CALL_PROGRESSING
0x6	DEV_HOLD_ACK	CSM_EVENT_ISDN_HARD_HOLD
0x7	DEV_HOLD_REJECT	CSM_EVENT_ISDN_HARD_HOLD_REJ
0x8	DEV_CHOLD_ACK	CSM_EVENT_ISDN_CHOLD
0x9	DEV_CHOLD_REJECT	CSM_EVENT_ISDN_CHOLD_REJ

Table 147 Event Values (continued)

Hexadecimal Value	Event	CSM Event
0xa	DEV_RETRIEVE_ACK	CSM_EVENT_ISDN_RETRIEVED
0xb	DEV_RETRIEVE_REJECT	CSM_EVENT_ISDN_RETRIEVE_REJ
0xc	DEV_CONFR_ACK	CSM_EVENT_ISDN_CONFERENCE
0xd	DEV_CONFR_REJECT	CSM_EVENT_ISDN_CONFERENCE_REJ
0xe	DEV_TRANS_ACK	CSM_EVENT_ISDN_TRANSFERRED
0xf	DEV_TRANS_REJECT	CSM_EVENT_ISDN_TRANSFER_REJ

Table 148 shows cause values that are assigned only to call-progressing events.

Table 148 Cause Values

Hexadecimal Value	Cause Definitions
0x01	UNASSIGNED_NUMBER
0x02	NO_ROUTE
0x03	NO_ROUTE_DEST
0x04	NO_PREFIX
0x06	CHANNEL_UNACCEPTABLE
0x07	CALL_AWARDED
0x08	CALL_PROC_OR_ERROR
0x09	PREFIX_DIALED_ERROR
0x0a	PREFIX_NOT_DIALED
0x0b	EXCESSIVE_DIGITS
0x0d	SERVICE_DENIED
0x10	NORMAL_CLEARING
0x11	USER_BUSY
0x12	NO_USER_RESPONDING
0x13	NO_USER_ANSWER
0x15	CALL_REJECTED
0x16	NUMBER_CHANGED
0x1a	NON_SELECTED_CLEARING
0x1b	DEST_OUT_OF_ORDER
0x1c	INVALID_NUMBER_FORMAT
0x1d	FACILITY_REJECTED
0x1e	RESP_TO_STAT_ENQ
0x1f	UNSPECIFIED_CAUSE
0x22	NO_CIRCUIT_AVAILABLE
0x26	NETWORK_OUT_OF_ORDER

Table 148 Cause Values (continued)

Hexadecimal Value	Cause Definitions
0x29	TEMPORARY_FAILURE
0x2a	NETWORK_CONGESTION
0x2b	ACCESS_INFO_DISCARDED
0x2c	REQ_CHANNEL_NOT_AVAIL
0x2d	PRE_EMPTED
0x2f	RESOURCES_UNAVAILABLE
0x32	FACILITY_NOT_SUBSCRIBED
0x33	BEARER_CAP_INCOMPAT
0x34	OUTGOING_CALL_BARRED
0x36	INCOMING_CALL_BARRED
0x39	BEARER_CAP_NOT_AUTH
0x3a	BEAR_CAP_NOT_AVAIL
0x3b	CALL_RESTRICTION
0x3c	REJECTED_TERMINAL
0x3e	SERVICE_NOT_ALLOWED
0x3f	SERVICE_NOT_AVAIL
0x41	CAP_NOT_IMPLEMENTED
0x42	CHAN_NOT_IMPLEMENTED
0x45	FACILITY_NOT_IMPLEMENT
0x46	BEARER_CAP_RESTRICTED
0x4f	SERV_OPT_NOT_IMPLEMENT
0x51	INVALID_CALL_REF
0x52	CHAN_DOES_NOT_EXIST
0x53	SUSPENDED_CALL_EXISTS
0x54	NO_CALL_SUSPENDED
0x55	CALL_ID_IN_USE
0x56	CALL_ID_CLEARED
0x58	INCOMPATIBLE_DEST
0x5a	SEGMENTATION_ERROR
0x5b	INVALID_TRANSIT_NETWORK
0x5c	CS_PARAMETER_NOT_VALID
0x5f	INVALID_MSG_UNSPEC
0x60	MANDATORY_IE_MISSING
0x61	NONEXISTENT_MSG
0x62	WRONG_MESSAGE
0x63	BAD_INFO_ELEM

Table 148 Cause Values (continued)

Hexadecimal Value	Cause Definitions
0x64	INVALID_ELEM_CONTENTS
0x65	WRONG_MSG_FOR_STATE
0x66	TIMER_EXPIRY
0x67	MANDATORY_IE_LEN_ERR
0x6f	PROTOCOL_ERROR
0x7f	INTERWORKING_UNSPEC

Examples

This section provides debug output examples for three call scenarios, displaying the sequence of events that occur during a POTS dial call or POTS disconnect call.

Call Scenario 1

In this example call scenario, port 1 is on the hook, the application dial is set to call 4085552221, and the far-end successfully connects.

```
Router# debug pots csm

Router# test pots 1 dial 4085552221#

Router#
```

The following screen output shows an event indicating that port 1 is being used by the dial application:

```
01:58:27: CSM_PROC_IDLE: CSM_EVENT_VDEV_APPLICATION_CALL, call id = 0x0, port = 1
```

The following screen output shows events indicating that the CSM is receiving the application digits of the number to dial:

```
01:58:27: CSM_PROC_APPLIC_DIALING: CSM_EVENT_VDEV_DIGIT, call id = 0x0, port = 1
01:58:27: CSM_PROC_APPLIC_DIALING: CSM_EVENT_VDEV_DIGIT, call id = 0x0, port = 1
01:58:27: CSM_PROC_APPLIC_DIALING: CSM_EVENT_VDEV_DIGIT, call id = 0x0, port = 1
01:58:27: CSM_PROC_APPLIC_DIALING: CSM_EVENT_VDEV_DIGIT, call id = 0x0, port = 1
01:58:27: CSM_PROC_APPLIC_DIALING: CSM_EVENT_VDEV_DIGIT, call id = 0x0, port = 1
01:58:27: CSM_PROC_APPLIC_DIALING: CSM_EVENT_VDEV_DIGIT, call id = 0x0, port = 1
01:58:27: CSM_PROC_APPLIC_DIALING: CSM_EVENT_VDEV_DIGIT, call id = 0x0, port = 1
01:58:27: CSM_PROC_APPLIC_DIALING: CSM_EVENT_VDEV_DIGIT, call id = 0x0, port = 1
01:58:27: CSM_PROC_APPLIC_DIALING: CSM_EVENT_VDEV_DIGIT, call id = 0x0, port = 1
01:58:27: CSM_PROC_APPLIC_DIALING: CSM_EVENT_VDEV_DIGIT, call id = 0x0, port = 1
```

The following screen output shows that the telephone connected to port 1 is off the hook:

```
    01:58:39: CSM_PROC_APPLIC_DIALING: CSM_EVENT_VDEV_OFFHOOK, call id = 0x0, port = 1
```

The following screen output shows a call-proceeding event pair indicating that the router ISDN software has sent the dialed digits to the ISDN switch:

```
01:58:40: EVENT_FROM_ISDN:dchan_idb=0x280AF38, call_id=0x8004, ces=0x1 bchan=0x0,
event=0x3, cause=0x0
01:58:40: CSM_PROC_ENBLOC_DIALING: CSM_EVENT_ISDN_PROC, call id =
0x8004, port = 1
```

The following screen output shows the call-progressing event pair indicating that the telephone at the far end is ringing:

```
01:58:40: EVENT_FROM_ISDN:dchan_idb=0x280AF38, call_id=0x8004, ces=0x1 bchan=0xFFFFFFFF,
event=0x5, cause=0x0
01:58:40: CSM_PROC_ENBLOC_DIALING: CSM_EVENT_ISDN_CALL_PROGRESSING, call id = 0x8004, port
= 1
```

The following screen output shows a call-connecting event pair indicating that the telephone at the far end has answered:

```
01:58:48: EVENT_FROM_ISDN:dchan_idb=0x280AF38, call_id=0x8004, ces=0x1 bchan=0xFFFFFFFF,
event=0x4, cause=0x0
01:58:48: CSM_PROC_CONNECTING: CSM_EVENT_ISDN_CONNECTED, call id = 0x8004, port = 1
```

The following screen output shows a call-progressing event pair indicating that the telephone at the far end has hung up and that the calling telephone is receiving an in-band tone from the ISDN switch:

```
01:58:55: EVENT_FROM_ISDN:dchan_idb=0x280AF38, call_id=0x8004, ces=0x1 bchan=0xFFFFFFFF,
event=0x5, cause=0x10
01:58:55: CSM_PROC_CONNECTED: CSM_EVENT_ISDN_CALL_PROGRESSING, call id = 0x8004, port = 1
```

The following screen output shows that the telephone connected to port 1 has hung up:

```
01:58:57: CSM_PROC_CONNECTED: CSM_EVENT_VDEV_ONHOOK, call id = 0x8004, port = 1
```

The following screen output shows an event pair indicating that the call has been terminated:

```
01:58:57: EVENT_FROM_ISDN:dchan_idb=0x280AF38, call_id=0x8004, ces=0x1 bchan=0xFFFFFFFF,
event=0x0, cause=0x0
01:58:57: CSM_PROC_NEAR_END_DISCONNECT: CSM_EVENT_ISDN_DISCONNECTED, call id = 0x8004,
port = 1
813_local#
```

Call Scenario 2

In this example scenario, port 1 is on the hook, the application dial is set to call 4085552221, and the destination number is busy.

```
Router# debug pots csm

Router# test pots 1 dial 4085552221#

Router#
```

The following screen output shows that port 1 is used by the dial application:

```
01:59:42: CSM_PROC_IDLE: CSM_EVENT_VDEV_APPLICATION_CALL, call id = 0x0, port = 1
```

The following screen output shows the events indicating that the CSM is receiving the application digits of the number to call:

```
01:59:42: CSM_PROC_APPLIC_DIALING: CSM_EVENT_VDEV_DIGIT, call id = 0x0, port = 1
01:59:42: CSM_PROC_APPLIC_DIALING: CSM_EVENT_VDEV_DIGIT, call id = 0x0, port = 1
01:59:42: CSM_PROC_APPLIC_DIALING: CSM_EVENT_VDEV_DIGIT, call id = 0x0, port = 1
01:59:42: CSM_PROC_APPLIC_DIALING: CSM_EVENT_VDEV_DIGIT, call id = 0x0, port = 1
01:59:42: CSM_PROC_APPLIC_DIALING: CSM_EVENT_VDEV_DIGIT, call id = 0x0, port = 1
01:59:42: CSM_PROC_APPLIC_DIALING: CSM_EVENT_VDEV_DIGIT, call id = 0x0, port = 1
01:59:42: CSM_PROC_APPLIC_DIALING: CSM_EVENT_VDEV_DIGIT, call id = 0x0, port = 1
01:59:42: CSM_PROC_APPLIC_DIALING: CSM_EVENT_VDEV_DIGIT, call id = 0x0, port = 1
01:59:42: CSM_PROC_APPLIC_DIALING: CSM_EVENT_VDEV_DIGIT, call id = 0x0, port = 1
01:59:42: CSM_PROC_APPLIC_DIALING: CSM_EVENT_VDEV_DIGIT, call id = 0x0, port = 1
```

The following screen output shows an event indicating that the telephone connected to port 1 is off the hook:

```
01:59:52: CSM_PROC_APPLIC_DIALING: CSM_EVENT_VDEV_OFFHOOK, call id = 0x0, port = 1
```

The following screen output shows a call-proceeding event pair indicating that the telephone at the far end is busy:

```
01:59:52: EVENT_FROM_ISDN:dchan_idb=0x280AF38, call_id=0x8005, ces=0x1 bchan=0x0,
event=0x3, cause=0x11
01:59:52: CSM_PROC_ENBLOC_DIALING: CSM_EVENT_ISDN_PROC, call id = 0x8005, port = 1
```

The following screen output shows a call-progressing event pair indicating that the calling telephone is receiving an in-band busy tone from the ISDN switch:

```
01:59:58: EVENT_FROM_ISDN:dchan_idb=0x280AF38, call_id=0x8005, ces=0x1 bchan=0xFFFFFFFF,
event=0x5, cause=0x0
01:59:58: CSM_PROC_ENBLOC_DIALING: CSM_EVENT_ISDN_CALL_PROGRESSING, call id = 0x8005, port
= 1
```

The following screen output shows an event indicating that the calling telephone has hung up:

```
02:00:05: CSM_PROC_ENBLOC_DIALING: CSM_EVENT_VDEV_ONHOOK, call id = 0x8005, port = 1
```

The following screen output shows an event pair indicating that the call has been terminated:

```
02:00:05: EVENT_FROM_ISDN:dchan_idb=0x280AF38, call_id=0x8005, ces=0x1 bchan=0xFFFFFFFF,
event=0x0, cause=0x0
02:00:05: CSM_PROC_NEAR_END_DISCONNECT: CSM_EVENT_ISDN_DISCONNECTED, call id = 0x8005,
port = 1
```

Call Scenario 3

In this example call scenario, port 1 is on the hook, the application dial is set to call 408-666-1112, the far end successfully connects, and the command **test pots disconnect** terminates the call:

```
Router# debug pots csm

Router# test pots 1 dial 4086661112

Router#
```

The following screen output follows the same sequence of events as shown in Call Scenario 1:

```
1d03h: CSM_PROC_IDLE: CSM_EVENT_VDEV_APPLICATION_CALL, call id = 0x0, port = 1
1d03h: CSM_PROC_APPLIC_DIALING: CSM_EVENT_VDEV_DIGIT, call id = 0x0, port = 1
1d03h: CSM_PROC_APPLIC_DIALING: CSM_EVENT_VDEV_DIGIT, call id = 0x0, port = 1
1d03h: CSM_PROC_APPLIC_DIALING: CSM_EVENT_VDEV_DIGIT, call id = 0x0, port = 1
1d03h: CSM_PROC_APPLIC_DIALING: CSM_EVENT_VDEV_DIGIT, call id = 0x0, port = 1
1d03h: CSM_PROC_APPLIC_DIALING: CSM_EVENT_VDEV_DIGIT, call id = 0x0, port = 1
1d03h: CSM_PROC_APPLIC_DIALING: CSM_EVENT_VDEV_DIGIT, call id = 0x0, port = 1
1d03h: CSM_PROC_APPLIC_DIALING: CSM_EVENT_VDEV_DIGIT, call id = 0x0, port = 1
1d03h: CSM_PROC_APPLIC_DIALING: CSM_EVENT_VDEV_DIGIT, call id = 0x0, port = 1
1d03h: CSM_PROC_APPLIC_DIALING: CSM_EVENT_VDEV_DIGIT, call id = 0x0, port = 1
1d03h: CSM_PROC_APPLIC_DIALING: CSM_EVENT_VDEV_DIGIT, call id = 0x0, port = 1

1d03h: CSM_PROC_APPLIC_DIALING: CSM_EVENT_VDEV_OFFHOOK, call id = 0x0, port = 1

1d03h: EVENT_FROM_ISDN:dchan_idb=0x2821F38, call_id=0x8039, ces=0x1
   bchan=0x0, event=0x3, cause=0x0
1d03h: CSM_PROC_ENBLOC_DIALING: CSM_EVENT_ISDN_PROC, call id = 0x8039, port = 1

1d03h: EVENT_FROM_ISDN:dchan_idb=0x2821F38, call_id=0x8039, ces=0x1
   bchan=0xFFFFFFFF, event=0x5, cause=0x0

1d03h: CSM_PROC_ENBLOC_DIALING: CSM_EVENT_ISDN_CALL_PROGRESSING, call id = 0x8039,
    port = 1
```

```
Router# test pots 1 disconnect
```

The **test pots disconnect** command disconnects the call before you physically need to put the telephone back on the hook:

```
1d03h: CSM_PROC_CONNECTING: CSM_EVENT_VDEV_APPLICATION_HANGUP_CALL, call id = 0x8039,
     port = 1
1d03h: EVENT_FROM_ISDN:dchan_idb=0x2821F38, call_id=0x8039, ces=0x1
   bchan=0xFFFFFFFF, event=0x0, cause=0x0

1d03h: CSM_PROC_DISCONNECTING: CSM_EVENT_ISDN_DISCONNECTED, call id = 0x8039,
    port = 1
1d03h: CSM_PROC_DISCONNECTING: CSM_EVENT_TIMEOUT, call id = 0x8039, port = 1
```

debug ppp

To display information on traffic and exchanges in an internetwork implementing the PPP, use the **debug ppp** privileged EXEC command. The **no** form of this command disables debugging output.

debug ppp {**packet** | **negotiation** | **error** | **authentication** | **compression** | **cbcp**}

no debug ppp {**packet** | **negotiation** | **error** | **authentication** | **compression** | **cbcp**}

Syntax Description

packet	Displays PPP packets being sent and received. (This command displays low-level packet dumps.)
negotiation	Displays PPP packets sent during PPP startup, where PPP options are negotiated.
error	Displays protocol errors and error statistics associated with PPP connection negotiation and operation.
authentication	Displays authentication protocol messages, including Challenge Authentication Protocol (CHAP) packet exchanges and Password Authentication Protocol (PAP) exchanges.
compression	Displays information specific to the exchange of PPP connections using MPPC. This command is useful for obtaining incorrect packet sequence number information where MPPC compression is enabled.
cbcp	Displays protocol errors and statistics associated with PPP connection negotiations using MSCB.

Usage Guidelines

Use the **debug ppp** command when trying to find the following:

- The Network Control Protocols (NCPs) that are supported on either end of a PPP connection
- Any loops that might exist in a PPP internetwork
- Nodes that are (or are not) properly negotiating PPP connections
- Errors that have occurred over the PPP connection
- Causes for CHAP session failures
- Causes for PAP session failures
- Information specific to the exchange of PPP connections using the Callback Control Protocol (CBCP), used by Microsoft clients
- Incorrect packet sequence number information where MPPC compression is enabled

Refer to Internet RFCs 1331, 1332, and 1333 for details concerning PPP-related nomenclature and protocol information.

Caution The **debug ppp compression** command is CPU-intensive and should be used with caution. This command should be disabled immediately after debugging.

Examples

The following is sample output from the **debug ppp packet** command as seen from the Link Quality Monitor (LQM) side of the connection. This display example depicts packet exchanges under normal PPP operation.

```
Router# debug ppp packet

PPP Serial4(o): lcp_slqr() state = OPEN magic = D21B4, len = 48
PPP Serial4(i): pkt type 0xC025, datagramsize 52
PPP Serial4(i): lcp_rlqr() state = OPEN magic = D3454, len = 48
PPP Serial4(i): pkt type 0xC021, datagramsize 16
PPP Serial4: I LCP ECHOREQ(9) id 3 (C) magic D3454
PPP Serial4: input(C021) state = OPEN code = ECHOREQ(9) id = 3 len = 12
PPP Serial4: O LCP ECHOREP(A) id 3 (C) magic D21B4
PPP Serial4(o): lcp_slqr() state = OPEN magic = D21B4, len = 48
PPP Serial4(i): pkt type 0xC025, datagramsize 52
PPP Serial4(i): lcp_rlqr() state = OPEN magic = D3454, len = 48
PPP Serial4(i): pkt type 0xC021, datagramsize 16
PPP Serial4: I LCP ECHOREQ(9) id 4 (C) magic D3454
PPP Serial4: input(C021) state = OPEN code = ECHOREQ(9) id = 4 len = 12
PPP Serial4: O LCP ECHOREP(A) id 4 (C) magic D21B4
PPP Serial4(o): lcp_slqr() state = OPEN magic = D21B4, len = 48
PPP Serial4(i): pkt type 0xC025, datagramsize 52
PPP Serial4(i): lcp_rlqr() state = OPEN magic = D3454, len = 48
PPP Serial4(i): pkt type 0xC021, datagramsize 16
PPP Serial4: I LCP ECHOREQ(9) id 5 (C) magic D3454
PPP Serial4: input(C021) state = OPEN code = ECHOREQ(9) id = 5 len = 12
PPP Serial4: O LCP ECHOREP(A) id 5 (C) magic D21B4
PPP Serial4(o): lcp_slqr() state = OPEN magic = D21B4, len = 48
PPP Serial4(i): pkt type 0xC025, datagramsize 52
PPP Serial4(i): lcp_rlqr() state = OPEN magic = D3454, len = 48
PPP Serial4(i): pkt type 0xC021, datagramsize 16
PPP Serial4: I LCP ECHOREQ(9) id 6 (C) magic D3454
PPP Serial4: input(C021) state = OPEN code = ECHOREQ(9) id = 6 len = 12
PPP Serial4: O LCP ECHOREP(A) id 6 (C) magic D21B4
PPP Serial4(o): lcp_slqr() state = OPEN magic = D21B4, len = 48
PPP Serial4(i): pkt type 0xC025, datagramsize 52
PPP Serial4(i): lcp_rlqr() state = OPEN magic = D3454, len = 48
PPP Serial4(i): pkt type 0xC021, datagramsize 16
PPP Serial4: I LCP ECHOREQ(9) id 7 (C) magic D3454
PPP Serial4: input(C021) state = OPEN code = ECHOREQ(9) id = 7 len = 12
PPP Serial4: O LCP ECHOREP(A) id 7 (C) magic D21B4
PPP Serial4(o): lcp_slqr() state = OPEN magic = D21B4, len = 48
```

Table 149 describes the significant fields shown in the display.

Table 149 debug ppp packet Field Descriptions

Field	Description
PPP	PPP debugging output.
Serial4	Interface number associated with this debugging information.
(o), O	Packet was detected as an output packet.
(i), I	Packet was detected as an input packet.
lcp_slqr()	Procedure name; running LQM, send a Link Quality Report (LQR).
lcp_rlqr()	Procedure name; running LQM, received an LQR.
input (C021)	Router received a packet of the specified packet type (in hexadecimal notation). A value of C025 indicates packet of type LQM.
state = OPEN	PPP state; normal state is OPEN.

Table 149 debug ppp packet Field Descriptions (continued)

Field	Description
magic = D21B4	Magic Number for indicated node; when output is indicated, this is the Magic Number of the node on which debugging is enabled. The actual Magic Number depends on whether the packet detected is indicated as I or O.
datagramsize 52	Packet length including header.
code = ECHOREQ(9)	Identifies the type of packet received. Both forms of the packet, string and hexadecimal, are presented.
len = 48	Packet length without header.
id = 3	ID number per Link Control Protocol (LCP) packet format.
pkt type 0xC025	Packet type in hexadecimal notation; typical packet types are C025 for LQM and C021 for LCP.
LCP ECHOREQ(9)	Echo Request; value in parentheses is the hexadecimal representation of the LCP type.
LCP ECHOREP(A)	Echo Reply; value in parentheses is the hexadecimal representation of the LCP type.

To elaborate on the displayed output, consider the partial exchange. This sequence shows that one side is using ECHO for its keepalives and the other side is using LQRs.

```
Router# debug ppp packet

PPP Serial4(o): lcp_slqr() state = OPEN magic = D21B4, len = 48
PPP Serial4(i): pkt type 0xC025, datagramsize 52
PPP Serial4(i): lcp_rlqr() state = OPEN magic = D3454, len = 48
PPP Serial4(i): pkt type 0xC021, datagramsize 16
PPP Serial4: I LCP ECHOREQ(9) id 3 (C) magic D3454
PPP Serial4: input(C021) state = OPEN code = ECHOREQ(9) id = 3 len = 12
PPP Serial4: O LCP ECHOREP(A) id 3 (C) magic D21B4
PPP Serial4(o): lcp_slqr() state = OPEN magic = D21B4, len = 48
```

The first line states that the router with debugging enabled has sent an LQR to the other side of the PPP connection:

```
PPP Serial4(o): lcp_slqr() state = OPEN magic = D21B4, len = 48
```

The next two lines indicate that the router has received a packet of type C025 (LQM) and provides details about the packet:

```
PPP Serial4(i): pkt type 0xC025, datagramsize 52
PPP Serial4(i): lcp_rlqr() state = OPEN magic = D3454, len = 48
```

The next two lines indicate that the router received an ECHOREQ of type C021 (LCP). The other side is sending ECHOs. The router on which debugging is configured for LQM but also responds to ECHOs.

```
PPP Serial4(i): pkt type 0xC021, datagramsize 16
PPP Serial4: I LCP ECHOREQ(9) id 3 (C) magic D3454
```

Next, the router is detected to have responded to the ECHOREQ with an ECHOREP and is preparing to send out an LQR:

```
PPP Serial4: O LCP ECHOREP(A) id 3 (C) magic D21B4
PPP Serial4(o): lcp_slqr() state = OPEN magic = D21B4, len = 48
```

The following is sample output from the **debug ppp negotiation** command. This is a normal negotiation, where both sides agree on Network Control Program (NCP) parameters. In this case, protocol type IP is proposed and acknowledged.

```
Router# debug ppp negotiation

ppp: sending CONFREQ, type = 4 (CI_QUALITYTYPE), value = C025/3E8
ppp: sending CONFREQ, type = 5 (CI_MAGICNUMBER), value = 3D56CAC
ppp: received config for type = 4 (QUALITYTYPE) acked
ppp: received config for type = 5 (MAGICNUMBER) value = 3D567F8 acked (ok)
PPP Serial4: state = ACKSENT fsm_rconfack(C021): rcvd id 5
ppp: config ACK received, type = 4 (CI_QUALITYTYPE), value = C025
ppp: config ACK received, type = 5 (CI_MAGICNUMBER), value = 3D56CAC
ppp: ipcp_reqci: returning CONFACK.
 (ok)
PPP Serial4: state = ACKSENT fsm_rconfack(8021): rcvd id 4
```

Table 150 describes significant fields shown in the display.

Table 150 debug ppp Command Negotiation Field Descriptions

Field	Description
ppp	PPP debugging output.
sending CONFREQ	Router sent a configuration request.
type = 4 (CI_QUALITYTYPE)	Type of LCP configuration option that is being negotiated and a descriptor. A type value of 4 indicates Quality Protocol negotiation; a type value of 5 indicates Magic Number negotiation.
value = C025/3E8	For Quality Protocol negotiation, indicates NCP type and reporting period. In the example, C025 indicates LQM; 3E8 is a hexadecimal value translating to about 10 seconds (in hundredths of a second).
value = 3D56CAC	For Magic Number negotiation, indicates the Magic Number being negotiated.
received config	Receiving node has received the proposed option negotiation for the indicated option type.
acked	Acknowledgment and acceptance of options.
state = ACKSENT	Specific PPP state in the negotiation process.
ipcp_reqci	IPCP notification message; sending CONFACK.
fsm_rconfack (8021)	Procedure fsm_rconfack processes received CONFACKs, and the protocol (8021) is IP.

The first two lines indicate that the router is trying to bring up LCP and will use the indicated negotiation options (Quality Protocol and Magic Number). The value fields are the values of the options themselves. C025/3E8 translates to Quality Protocol LQM. 3E8 is the reporting period (in hundredths of a second). 3D56CAC is the value of the Magic Number for the router.

```
ppp: sending CONFREQ, type = 4 (CI_QUALITYTYPE), value = C025/3E8
ppp: sending CONFREQ, type = 5 (CI_MAGICNUMBER), value = 3D56CAC
```

The next two lines indicate that the other side negotiated for options 4 and 5 as requested and acknowledged both. If the responding end does not support the options, a CONFREJ is sent by the responding node. If the responding end does not accept the value of the option, a CONFNAK is sent with the value field modified.

```
ppp: received config for type = 4 (QUALITYTYPE) acked
ppp: received config for type = 5 (MAGICNUMBER) value = 3D567F8 acked (ok)
```

The next three lines indicate that the router received a CONFACK from the responding side and displays accepted option values. Use the rcvd id field to verify that the CONFREQ and CONFACK have the same ID field.

```
PPP Serial4: state = ACKSENT fsm_rconfack(C021): rcvd id 5
ppp: config ACK received, type = 4 (CI_QUALITYTYPE), value = C025
ppp: config ACK received, type = 5 (CI_MAGICNUMBER), value = 3D56CAC
```

The next line indicates that the router has IP routing enabled on this interface and that the IPCP NCP negotiated successfully:

```
ppp: ipcp_reqci: returning CONFACK.
```

In the last line, the state of the router is listed as ACKSENT.

```
PPP Serial4: state = ACKSENT fsm_rconfack(C021): rcvd id 5\
```

The following is sample output from when the **debug ppp packet** and **debug ppp negotiation** commands are enabled at the same time.

```
router# debug ppp negotiation
router# debug ppp packet

ppp: sending CONFREQ, type = 4 (CI_QUALITYTYPE), value = C025/3E8
ppp: sending CONFREQ, type = 5 (CI_MAGICNUMBER), value = D4C64
PPP Serial4: O LCP CONFREQ(1) id 4 (12) QUALITYTYPE (8) 192 37 0 0 3 232
   MAGICNUMBER (6) 0 13 76 100
PPP Serial4(i): pkt type 0xC021, datagramsize 22
PPP Serial4: I LCP CONFREQ(1) id 4 (12) QUALITYTYPE (8) 192 37 0 0 3 232
   MAGICNUMBER (6) 0 13 84 240
PPP Serial4: input(C021) state = REQSENT code = CONFREQ(1) id = 4 len = 18
ppp: received config for type = 4 (QUALITYTYPE) acked
ppp: received config for type = 5 (MAGICNUMBER) value = D54F0 acked
PPP Serial4: O LCP CONFACK(2) id 4 (12) QUALITYTYPE (8) 192 37 0 0 3 232
   MAGICNUMBER (6) 0 13 84 240 (ok)
PPP Serial4(i): pkt type 0xC021, datagramsize 22
PPP Serial4: I LCP CONFACK(2) id 4 (12) QUALITYTYPE (8) 192 37 0 0 3 232
   MAGICNUMBER (6) 0 13 76 100
PPP Serial4: input(C021) state = ACKSENT code = CONFACK(2) id = 4 len = 18
PPP Serial4: state = ACKSENT fsm_rconfack(C021): rcvd id 4
ppp: config ACK received, type = 4 (CI_QUALITYTYPE), value = C025
ppp: config ACK received, type = 5 (CI_MAGICNUMBER), value = D4C64
ipcp: sending CONFREQ, type = 3 (CI_ADDRESS), Address = 2.1.1.2
PPP Serial4: O IPCP CONFREQ(1) id 3 (10) Type3 (6) 2 1 1 2
PPP Serial4: I IPCP CONFREQ(1) id 3 (10) Type3 (6) 2 1 1 1
PPP Serial4(i): pkt type 0x8021, datagramsize 14
PPP Serial4: input(8021) state = REQSENT code = CONFREQ(1) id = 3 len = 10
ppp Serial4: Negotiate IP address: her address 2.1.1.1 (ACK)
ppp: ipcp_reqci: returning CONFACK.
PPP Serial4: O IPCP CONFACK(2) id 3 (10) Type3 (6) 2 1 1 1 (ok)
PPP Serial4: I IPCP CONFACK(2) id 3 (10) Type3 (6) 2 1 1 2
PPP Serial4: input(8021) state = ACKSENT code = CONFACK(2) id = 3 len = 10
PPP Serial4: state = ACKSENT fsm_rconfack(8021): rcvd id 3
ipcp: config ACK received, type = 3 (CI_ADDRESS), Address = 2.1.1.2
PPP Serial4(o): lcp_slqr() state = OPEN magic = D4C64, len = 48
PPP Serial4(i): pkt type 0xC025, datagramsize 52
PPP Serial4(i): lcp_rlqr() state = OPEN magic = D54F0, len = 48
PPP Serial4(i): pkt type 0xC025, datagramsize 52
PPP Serial4(i): lcp_rlqr() state = OPEN magic = D54F0, len = 48
PPP Serial4(o): lcp_slqr() state = OPEN magic = D4C64, len = 48
```

This field shows a decimal representation of the Magic Number.

This field shows a decimal representation of the NCP value.

This field shows a decimal representation of the reporting period.

This exchange represents a successful PPP negotiation for support of NCP type IPCP.

S2877

The following is sample output from the **debug ppp negotiation** command when the remote side of the connection is unable to respond to LQM requests:

```
Router# debug ppp negotiation

ppp: sending CONFREQ, type = 4 (CI_QUALITYTYPE), value = C025/3E8
ppp: sending CONFREQ, type = 5 (CI_MAGICNUMBER), value = 44B7010
ppp: sending CONFREQ, type = 4 (CI_QUALITYTYPE), value = C025/3E8
ppp: sending CONFREQ, type = 5 (CI_MAGICNUMBER), value = 44B7010
ppp: sending CONFREQ, type = 4 (CI_QUALITYTYPE), value = C025/3E8
ppp: sending CONFREQ, type = 5 (CI_MAGICNUMBER), value = 44B7010
ppp: sending CONFREQ, type = 4 (CI_QUALITYTYPE), value = C025/3E8
ppp: sending CONFREQ, type = 5 (CI_MAGICNUMBER), value = 44B7010
ppp: sending CONFREQ, type = 4 (CI_QUALITYTYPE), value = C025/3E8
ppp: sending CONFREQ, type = 5 (CI_MAGICNUMBER), value = 44B7010
ppp: sending CONFREQ, type = 4 (CI_QUALITYTYPE), value = C025/3E8
ppp: sending CONFREQ, type = 5 (CI_MAGICNUMBER), value = 44B7010
ppp: sending CONFREQ, type = 4 (CI_QUALITYTYPE), value = C025/3E8
ppp: sending CONFREQ, type = 5 (CI_MAGICNUMBER), value = 44B7010
ppp: sending CONFREQ, type = 4 (CI_QUALITYTYPE), value = C025/3E8
ppp: sending CONFREQ, type = 5 (CI_MAGICNUMBER), value = 44B7010
ppp: sending CONFREQ, type = 4 (CI_QUALITYTYPE), value = C025/3E8
ppp: sending CONFREQ, type = 5 (CI_MAGICNUMBER), value = 44B7010
ppp: sending CONFREQ, type = 4 (CI_QUALITYTYPE), value = C025/3E8
ppp: sending CONFREQ, type = 5 (CI_MAGICNUMBER), value = 44B7010
ppp: sending CONFREQ, type = 4 (CI_QUALITYTYPE), value = C025/3E8
ppp: sending CONFREQ, type = 5 (CI_MAGICNUMBER), value = 44B7010
ppp: sending CONFREQ, type = 4 (CI_QUALITYTYPE), value = C025/3E8
ppp: sending CONFREQ, type = 5 (CI_MAGICNUMBER), value = 44C1488
```

The following is sample output when no response is detected for configuration requests (with both the **debug ppp negotiation** and **debug ppp packet** command enabled):

```
Router# debug ppp negotiation

Router# debug ppp packet

ppp: sending CONFREQ, type = 4 (CI_QUALITYTYPE), value = C025/3E8
ppp: sending CONFREQ, type = 5 (CI_MAGICNUMBER), value = 44DFDC8
PPP Serial4: O LCP CONFREQ(1) id 14 (12) QUALITYTYPE (8) 192 37 0 0 3 232
   MAGICNUMBER (6) 4 77 253 200
ppp: TIMEout: Time= 44E0980 State= 3
ppp: sending CONFREQ, type = 4 (CI_QUALITYTYPE), value = C025/3E8
ppp: sending CONFREQ, type = 5 (CI_MAGICNUMBER), value = 44DFDC8
PPP Serial4: O LCP CONFREQ(1) id 15 (12) QUALITYTYPE (8) 192 37 0 0 3 232
   MAGICNUMBER (6) 4 77 253 200
ppp: TIMEout: Time= 44E1828 State= 3
ppp: sending CONFREQ, type = 4 (CI_QUALITYTYPE), value = C025/3E8
ppp: sending CONFREQ, type = 5 (CI_MAGICNUMBER), value = 44DFDC8
PPP Serial4: O LCP CONFREQ(1) id 16 (12) QUALITYTYPE (8) 192 37 0 0 3 232
   MAGICNUMBER (6) 4 77 253 200
ppp: TIMEout: Time= 44E27C8 State= 3
ppp: sending CONFREQ, type = 4 (CI_QUALITYTYPE), value = C025/3E8
ppp: sending CONFREQ, type = 5 (CI_MAGICNUMBER), value = 44DFDC8
PPP Serial4: O LCP CONFREQ(1) id 17 (12) QUALITYTYPE (8) 192 37 0 0 3 232
   MAGICNUMBER (6) 4 77 253 200
ppp: TIMEout: Time= 44E3768 State= 3
```

The following is sample output from the **debug ppp error** command. These messages might appear when the Quality Protocol option is enabled on an interface that is already running PPP.

```
Router# debug ppp error

PPP Serial3(i): rlqr receive failure. successes = 15
PPP: myrcvdiffp = 159 peerxmitdiffp = 41091
PPP: myrcvdiffo = 2183 peerxmitdiffo = 1714439
PPP: threshold = 25
```

```
PPP Serial4(i): rlqr transmit failure. successes = 15
PPP: myxmitdiffp = 41091 peerrcvdiffp = 159
PPP: myxmitdiffo = 1714439 peerrcvdiffo = 2183
PPP: l->OutLQRs = 1 LastOutLQRs = 1
PPP: threshold = 25
PPP Serial3(i): lqr_protrej() Stop sending LQRs.
PPP Serial3(i): The link appears to be looped back.
```

Table 151 describes the significant fields shown in the display.

Table 151 debug ppp Error Field Descriptions

Field	Description
PPP	PPP debugging output.
Serial3(i)	Interface number associated with this debugging information; indicates that this is an input packet.
rlqr receive failure	Request to negotiate the Quality Protocol option is not accepted.
myrcvdiffp = 159	Number of packets received over the time period.
peerxmitdiffp = 41091	Number of packets sent by the remote node over this period.
myrcvdiffo = 2183	Number of octets received over this period.
peerxmitdiffo = 1714439	Number of octets sent by the remote node over this period.
threshold = 25	Maximum error percentage acceptable on this interface. This percentage is calculated by the threshold value entered in the **ppp quality** *number* interface configuration command. A value of 100 – *number* (100 minus *number*) is the maximum error percentage. In this case, a *number* of 75 was entered. This means that the local router must maintain a minimum 75 percent non-error percentage, or the PPP link will be considered down.
OutLQRs = 1	Local router's current send LQR sequence number.
LastOutLQRs = 1	The last sequence number that the remote node side has seen from the local node.

The following is sample output from the **debug ppp authentication** command. Use this **debug** command to determine why an authentication fails.

```
Router# debug ppp authentication

Serial0: Unable to authenticate. No name received from peer
Serial0: Unable to validate CHAP response. USERNAME pioneer not found.
Serial0: Unable to validate CHAP response. No password defined for USERNAME pioneer
Serial0: Failed CHAP authentication with remote.
Remote message is Unknown name
Serial0: remote passed CHAP authentication.
Serial0: Passed CHAP authentication with remote.
Serial0: CHAP input code = 4 id = 3 len = 48
```

In general, these messages are self-explanatory. Fields that can show optional output are outlined in Table 152.

Table 152 debug ppp authentication Field Descriptions

Field	Description
Serial0	Interface number associated with this debugging information and CHAP access session in question.
USERNAME pioneer not found.	The name *pioneer* in this example is the name received in the CHAP response. The router looks up this name in the list of usernames that are configured for the router.
Remote message is Unknown name	The following messages can appear: • No name received to authenticate • Unknown name • No secret for given name • Short MD5 response received • MD compare failed
code = 4	Specific CHAP type packet detected. Possible values are as follows: • 1—Challenge • 2—Response • 3—Success • 4—Failure
id = 3	ID number per LCP packet format.
len = 48	Packet length without header.

The following shows sample output from the **debug ppp** command using the **cbcp** keyword. This output depicts packet exchanges under normal PPP operation where the Cisco access server is waiting for the remote PC to respond to the MSCB request. The router also has **debug ppp negotiation** and **service timestamps msec** commands enabled.

```
Router# debug ppp cbcp

Dec 17 00:48:11.302: As8 MCB: User mscb Callback Number - Client ANY
Dec 17 00:48:11.306: Async8 PPP: O MCB Request(1) id 1 len 9
Dec 17 00:48:11.310: Async8 MCB: O  1  1  0  9  2  5  0  1  0
Dec 17 00:48:11.314: As8 MCB: O Request Id 1 Callback Type Client-Num delay 0
Dec 17 00:48:13.342: As8 MCB: Timeout in state WAIT_RESPONSE
Dec 17 00:48:13.346: Async8 PPP: O MCB Request(1) id 2 len 9
Dec 17 00:48:13.346: Async8 MCB: O  1  2  0  9  2  5  0  1  0
Dec 17 00:48:13.350: As8 MCB: O Request Id 2 Callback Type Client-Num delay 0
Dec 17 00:48:15.370: As8 MCB: Timeout in state WAIT_RESPONSE
Dec 17 00:48:15.374: Async8 PPP: O MCB Request(1) id 3 len 9
Dec 17 00:48:15.374: Async8 MCB: O  1  3  0  9  2  5  0  1  0
Dec 17 00:48:15.378: As8 MCB: O Request Id 3 Callback Type Client-Num delay 0
Dec 17 00:48:17.398: As8 MCB: Timeout in state WAIT_RESPONSE
Dec 17 00:48:17.402: Async8 PPP: O MCB Request(1) id 4 len 9
Dec 17 00:48:17.406: Async8 MCB: O  1  4  0  9  2  5  0  1  0
Dec 17 00:48:17.406: As8 MCB: O Request Id 4 Callback Type Client-Num delay 0
Dec 17 00:48:19.426: As8 MCB: Timeout in state WAIT_RESPONSE
Dec 17 00:48:19.430: Async8 PPP: O MCB Request(1) id 5 len 9
Dec 17 00:48:19.430: Async8 MCB: O  1  5  0  9  2  5  0  1  0
Dec 17 00:48:19.434: As8 MCB: O Request Id 5 Callback Type Client-Num delay 0
Dec 17 00:48:21.454: As8 MCB: Timeout in state WAIT_RESPONSE
```

```
Dec 17 00:48:21.458: Async8 PPP: O MCB Request(1) id 6 len 9
Dec 17 00:48:21.462: Async8 MCB: O  1  6  0  9  2  5  0  1  0
Dec 17 00:48:21.462: As8 MCB: O Request Id 6 Callback Type Client-Num delay 0
Dec 17 00:48:23.482: As8 MCB: Timeout in state WAIT_RESPONSE
Dec 17 00:48:23.486: Async8 PPP: O MCB Request(1) id 7 len 9
Dec 17 00:48:23.490: Async8 MCB: O  1  7  0  9  2  5  0  1  0
Dec 17 00:48:23.490: As8 MCB: O Request Id 7 Callback Type Client-Num delay 0
Dec 17 00:48:25.510: As8 MCB: Timeout in state WAIT_RESPONSE
Dec 17 00:48:25.514: Async8 PPP: O MCB Request(1) id 8 len 9
Dec 17 00:48:25.514: Async8 MCB: O  1  8  0  9  2  5  0  1  0
Dec 17 00:48:25.518: As8 MCB: O Request Id 8 Callback Type Client-Num delay 0
Dec 17 00:48:26.242: As8 PPP: I pkt type 0xC029, datagramsize 18
Dec 17 00:48:26.246: Async8 PPP: I MCB Response(2) id 8 len 16
Dec 17 00:48:26.250: Async8 MCB: I  2  8  0  10  2  C  C  1  32  34  39  32  36  31  33  0
Dec 17 00:48:26.254: As8 MCB: Received response
Dec 17 00:48:26.258: As8 MCB: Response CBK-Client-Num 2 12 12,  addr 1-2492613
Dec 17 00:48:26.262: Async8 PPP: O MCB Ack(3) id 9 len 16
Dec 17 00:48:26.266: Async8 MCB: O  3  9  0  10  2  C  C  1  32  34  39  32  36  31  33  0
Dec 17 00:48:26.270: As8 MCB: O Ack Id 9 Callback Type Client-Num delay 12
Dec 17 00:48:26.270: As8 MCB: Negotiated MCB with peer
Dec 17 00:48:26.390: As8 LCP: I TERMREQ [Open] id 4 len 8 (0x00000000)
Dec 17 00:48:26.390: As8 LCP: O TERMACK [Open] id 4 len 4
Dec 17 00:48:26.394: As8 MCB: Peer terminating the link
Dec 17 00:48:26.402: As8 MCB: Initiate Callback for mscb at 2492613 using Async
```

The following is sample output from the **debug ppp compression** command with **service timestamps** enabled and shows a typical PPP packet exchange between the router and Microsoft client where the MPPC header sequence numbers increment correctly:

```
Router# debug ppp compression

00:04:14: BR0:1 MPPC: Decomp - hdr/exp_cc# 0x2003/0x0003
00:04:14: BR0:1 MPPC: Decomp - hdr/exp_cc# 0x2004/0x0004
00:04:14: BR0:1 MPPC: Decomp - hdr/exp_cc# 0x2005/0x0005
00:04:14: BR0:1 MPPC: Decomp - hdr/exp_cc# 0x2006/0x0006
00:04:14: BR0:1 MPPC: Decomp - hdr/exp_cc# 0x2007/0x0007
```

Table 153 describes the fields for the **debug ppp compression** output.

Table 153 debug ppp compression Field Descriptions

Field	Description
interface	Interface enabled with MPPC.
Decomp - hdr/	Decompression header and bit settings.
exp_cc#	Expected coherency count.
0x2003	Received sequence number.
0x0003	Expected sequence number.

The following shows sample output from **debug ppp negotiation** and **debug ppp error** commands, which can be used to troubleshoot initial PPP negotiation and setup errors. This example shows a virtual interface (virtual interface 1) during normal PPP operation and CCP negotiation.

```
Router# debug ppp negotiation error

Vt1 PPP: Unsupported or un-negotiated protocol. Link arp
VPDN: Chap authentication succeeded for p5200
Vi1 PPP: Phase is DOWN, Setup
Vi1 VPDN: Virtual interface created for dinesh@cisco.com
```

```
Vi1 VPDN: Set to Async interface
Vi1 PPP: Phase is DOWN, Setup
Vi1 VPDN: Clone from Vtemplate 1 filterPPP=0 blocking
Vi1 CCP: Re-Syncing history using legacy method
%LINK-3-UPDOWN: Interface Virtual-Access1, changed state to up
Vi1 PPP: Treating connection as a dedicated line
Vi1 PPP: Phase is ESTABLISHING, Active Open
Vi1 LCP: O CONFREQ [Closed] id 1 len 25
Vi1 LCP:    ACCM 0x000A0000 (0x0206000A0000)
Vi1 LCP:    AuthProto CHAP (0x0305C22305)
Vi1 LCP:    MagicNumber 0x000FB69F (0x0506000FB69F)
Vi1 LCP:    PFC (0x0702)
Vi1 LCP:    ACFC (0x0802)
Vi1 VPDN: Bind interface direction=2
Vi1 PPP: Treating connection as a dedicated line
Vi1 LCP: I FORCED CONFREQ len 21
Vi1 LCP:    ACCM 0x000A0000 (0x0206000A0000)
Vi1 LCP:    AuthProto CHAP (0x0305C22305)
Vi1 LCP:    MagicNumber 0x12A5E4B5 (0x050612A5E4B5)
Vi1 LCP:    PFC (0x0702)
Vi1 LCP:    ACFC (0x0802)
Vi1 VPDN: PPP LCP accepted sent & rcv CONFACK
Vi1 PPP: Phase is AUTHENTICATING, by this end
Vi1 CHAP: O CHALLENGE id 1 len 27 from "l_4000"
Vi1 CHAP: I RESPONSE id 20 len 37 from "dinesh@cisco.com"
Vi1 CHAP: O SUCCESS id 20 len 4
Vi1 PPP: Phase is UP
Vi1 IPCP: O CONFREQ [Closed] id 1 len 10
Vi1 IPCP:    Address 15.2.2.3 (0x03060F020203)
Vi1 CCP: O CONFREQ [Not negotiated] id 1 len 10
Vi1 CCP:    MS-PPC supported bits 0x00000001 (0x120600000001)
Vi1 IPCP: I CONFREQ [REQsent] id 1 len 34
Vi1 IPCP:    Address 0.0.0.0 (0x030600000000)
Vi1 IPCP:    PrimaryDNS 0.0.0.0 (0x810600000000)
Vi1 IPCP:    PrimaryWINS 0.0.0.0 (0x820600000000)
Vi1 IPCP:    SecondaryDNS 0.0.0.0 (0x830600000000)
Vi1 IPCP:    SecondaryWINS 0.0.0.0 (0x840600000000)
Vi1 IPCP: Using the default pool
Vi1 IPCP: Pool returned 11.2.2.5
Vi1 IPCP: O CONFREJ [REQsent] id 1 len 16
Vi1 IPCP:    PrimaryWINS 0.0.0.0 (0x820600000000)
Vi1 IPCP:    SecondaryWINS 0.0.0.0 (0x840600000000)
Vi1 CCP: I CONFREQ [REQsent] id 1 len 15
Vi1 CCP:    MS-PPC supported bits 0x00000001 (0x120600000001)
Vi1 CCP:    Stacker history 1 check mode EXTENDED (0x1105000104)
Vi1 CCP: Already accepted another CCP option, rejecting this STACKER
Vi1 CCP: O CONFREJ [REQsent] id 1 len 9
Vi1 CCP:    Stacker history 1 check mode EXTENDED (0x1105000104)
Vi1 IPCP: I CONFACK [REQsent] id 1 len 10
Vi1 IPCP:    Address 15.2.2.3 (0x03060F020203)
Vi1 CCP: I CONFACK [REQsent] id 1 len 10
Vi1 CCP:    MS-PPC supported bits 0x00000001 (0x120600000001)
Vi1 CCP: I CONFREQ [ACKrcvd] id 2 len 10
Vi1 CCP:    MS-PPC supported bits 0x00000001 (0x120600000001)
Vi1 CCP: O CONFACK [ACKrcvd] id 2 len 10
Vi1 CCP:    MS-PPC supported bits 0x00000001 (0x120600000001)
Vi1 CCP: State is Open
Vi1 IPCP: I CONFREQ [ACKrcvd] id 2 len 22
Vi1 IPCP:    Address 0.0.0.0 (0x030600000000)
Vi1 IPCP:    PrimaryDNS 0.0.0.0 (0x810600000000)
Vi1 IPCP:    SecondaryDNS 0.0.0.0 (0x830600000000)
Vi1 IPCP: O CONFNAK [ACKrcvd] id 2 len 22
Vi1 IPCP:    Address 11.2.2.5 (0x03060B020205)
Vi1 IPCP:    PrimaryDNS 171.69.1.148 (0x8106AB450194)
```

```
Vi1 IPCP:    SecondaryDNS 171.69.2.132 (0x8306AB450284)
Vi1 IPCP: I CONFREQ [ACKrcvd] id 3 len 22
Vi1 IPCP:    Address 11.2.2.5 (0x03060B020205)
Vi1 IPCP:    PrimaryDNS 171.69.1.148 (0x8106AB450194)
Vi1 IPCP:    SecondaryDNS 171.69.2.132 (0x8306AB450284)
Vi1 IPCP: O CONFACK [ACKrcvd] id 3 len 22
Vi1 IPCP:    Address 11.2.2.5 (0x03060B020205)
Vi1 IPCP:    PrimaryDNS 171.69.1.148 (0x8106AB450194)
Vi1 IPCP:    SecondaryDNS 171.69.2.132 (0x8306AB450284)
Vi1 IPCP: State is Open
Vi1 IPCP: Install route to 11.2.2.5
```