Cisco IOS Debug Command Reference - Commands S through Z

Americas Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
http://www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0883

CONTENTS

CHAPTER 1 **debug saa apm through debug snmp sync** 1

- debug saa apm 3
- debug saa slm 5
- debug saa xml 7
- debug sampler 8
- debug satellite 10
- debug satellite firmware 14
- debug sccp 21
- debug sccp config 25
- debug qbm 27
- debug sdlc 29
- debug sdlc local-ack 33
- debug sdlc packet 35
- debug serial interface 36
- debug serial lead-transition 42
- debug serial packet 44
- debug service-group 45
- debug service-module 46
- debug sgbp dial-bids 47
- debug sgbp error 48
- debug sgbp hellos 50
- debug sgcp 52
- debug sgcp errors 54
- debug sgcp events 56
- debug sgcp packet 60
- debug shared-line 64
- debug smrp all 67
- debug smrp group 69

debug smrp mcache 72

debug smrp neighbor 74

debug smrp port 76

debug smrp route 78

debug smrp transaction 80

debug snasw dlc 82

debug snasw ips 84

debug snmp bulkstat 86

debug snmp detail 89

debug snmp mib nhrp 91

debug snmp overhead 94

debug snmp packet 95

debug snmp requests 98

debug snmp sync 99

debug snmp tunnel-mib 101

CHAPTER 2 **debug sntp adjust through debug tag-switching xtagatm vc** 103

debug sntp adjust 106

debug sntp packets 107

debug sntp select 109

debug software authenticity 110

debug source bridge 114

debug source error 116

debug source event 118

debug span 125

debug spanning-tree 128

debug ss7 mtp1 130

debug ss7 mtp2 134

debug ss7 sm 141

debug sse 143

debug ssg ctrl-errors 144

debug ssg ctrl-events 146

debug ssg ctrl-packets 147

debug ssg data 149

debug ssg data-nat 151

debug ssg dhcp 153

debug ssg errors 155

debug ssg events 156

debug ssg packets 158

debug ssg port-map 160

debug ssg tcp-redirect 162

debug ssg transparent login 165

debug ssl 167

debug ssl openssl 169

debug ssm 171

debug sss aaa authorization event 175

debug sss aaa authorization fsm 178

debug sss error 180

debug sss event 182

debug sss fsm 184

debug standby 186

debug standby errors 189

debug standby events 191

debug standby events icmp 195

debug standby events neighbor 197

debug standby packets 199

debug stun packet 201

debug subscriber aaa authorization 204

debug subscriber error 206

debug subscriber event 207

debug subscriber feature 208

debug subscriber fsm 211

debug subscriber packet 212

debug subscriber policy 214

debug subscriber service 217

debug subscriber testing 219

debug sw56 220

debug syscon perfdata 221

debug syscon sdp 222

debug syslog-server 223

debug tacacs 225

debug tacacs events 227

debug tag-switching atm-cos 230

debug tag-switching atm-tdp api 231

debug tag-switching atm-tdp routes 232

debug tag-switching atm-tdp states 233

debug tag-switching tdp advertisements 234

debug tag-switching tdp bindings 235

debug tag-switching tdp directed-neighbors 236

debug tag-switching tdp peer state-machine 237

debug tag-switching tdp pies received 238

debug tag-switching tdp pies sent 239

debug tag-switching tdp session io 240

debug tag-switching tdp session state-machine 241

debug tag-switching tdp transport connections 242

debug tag-switching tdp transport events 243

debug tag-switching tdp transport timers 244

debug tag-switching xtagatm cross-connect 246

debug tag-switching xtagatm errors 247

debug tag-switching xtagatm events 248

debug tag-switching xtagatm vc 249

CHAPTER 3 **debug tag-template event through debug voip application vxml** 251

debug tag-template event 254

debug tarp events 255

debug tarp packets 258

debug tbridge virtual-port 260

debug tcam_mgr 261

debug tccs signaling 263

debug tdm 265

debug telco-return msg 268

debug telnet 269

debug text-to-fax 272

debug tftp 273

debug tgrep error 274

debug tgrep events 276

debug tgrep fsm 278

debug tgrep io 281

debug tgrep messages 284

debug tgrep msgdump 286

debug tgrep timer-event 289

debug tgrep timers 293

debug tgrep tripr 295

debug tgrm 299

debug tiff reader 303

debug tiff writer 305

debug time-range ipc 306

debug token ring 307

debug topology 310

debug track 312

debug trifecta 314

debug tsp 316

debug tunnel rbscp 318

debug tunnel route-via 321

debug txconn all 322

debug txconn appc 324

debug txconn config 326

debug txconn data 328

debug txconn event 330

debug txconn tcp 332

debug txconn timer 334

debug udptn 336

debug usb driver 337

debug user-group 340

debug v120 event 342

debug v120 packet 343

debug vfi checkpoint 345

debug vg-anylan 347

debug video vicm 349

debug vlan packet 351

debug voice aaa asnl 352
debug voice all 353
debug voice cp 354
debug voice dsp crash-dump 356
debug voice eecm 358
debug voice enum 360
debug voice fastpath 363
debug voice h221 367
debug voice h324 370
debug voice mlpp 375
debug voice protocol 377
debug voice register errors 379
debug voice register events 381
debug voice signaling 385
debug voice source-group 387
debug voice statistics 389
debug voice tdsm 394
debug voice translation 396
debug voice uri 399
debug voice vofr 402
debug voip aaa 404
debug voip ais 405
debug voip application 407
debug voip application stcapp all 414
debug voip application stcapp buffer-history 421
debug voip application stcapp error 423
debug voip application stcapp events 425
debug voip application stcapp functions 427
debug voip application stcapp port 429
debug voip application vxml 437

CHAPTER 4 debug voip avlist through debug vpm signaling 445
debug voip application lpcor 448
debug voip avlist 450
debug voip ccapi 451

debug voip ccapi error 467
debug voip ccapi inout 468
debug voip confmsp 477
debug voip dcapi 480
debug voip dialpeer 482
debug voip dsm 486
debug voip dsmp 492
debug voip dspapi 500
debug voip dump-file-acct 507
debug voip eddri 509
debug voip enum 511
debug voip event-log 514
debug voip fastpath 516
debug voip fileacct 520
debug voip fpi call-rate 522
debug voip h221 523
debug voip h324 526
debug voip hpi 531
debug voip ipipgw 538
debug voip ivr 542
debug voip ivr all 543
debug voip ivr applib 544
debug voip ivr callsetup 545
debug voip ivr digitcollect 546
debug voip ivr dynamic 547
debug voip ivr error 548
debug voip ivr redirect 549
debug voip ivr script 550
debug voip ivr settlement 551
debug voip ivr states 552
debug voip ivr supplementary-service 553
debug voip ivr tclcommands 554
debug voip lpcor 555
debug voip profile fax 557
debug voip profile help 561

debug voip profile modem 564

debug voip profile voice 567

debug voip rawmsg 570

debug voip rtcp 573

debug voip rtp 575

debug voip settlement all 578

debug voip settlement enter 580

debug voip settlement error 581

debug voip settlement exit 586

debug voip settlement misc 587

debug voip settlement network 588

debug voip settlement security 591

debug voip settlement ssl 592

debug voip settlement transaction 594

debug voip tsp 595

debug voip vtsp 601

debug voip xcodemsp 612

debug vpdn 614

debug vpdn pppoe-data 632

debug vpdn pppoe-error 635

debug vpdn pppoe-events 638

debug vpdn pppoe-packet 640

debug vpdn redundancy 642

debug vpm all 649

debug vpm dsp 651

debug vpm error 652

debug vpm port 654

debug vpm signal 656

debug vpm signaling 657

CHAPTER 5 **debug vpm spi through voice call debug** 659

debug vpm spi 662

debug vpm trunk_sc 664

debug vpm voaal2 all 666

debug vpm voaal2 type1 668

debug vpm voaal2 type3 670
debug vrf 672
debug vrrp all 674
debug vrrp authentication 676
debug vrrp error 677
debug vrrp events 679
debug vrrp ha 681
debug vrrp packets 683
debug vrrp state 685
debug vrrp vrrs 687
debug vrrs all 689
debug vrrs accounting 691
debug vrrs database 692
debug vrrs infra 694
debug vrrs log 696
debug vrrs pathway 697
debug vrrs plugin 700
debug vsi api 702
debug vsi errors 704
debug vsi events 707
debug vsi packets 710
debug vsi param-groups 712
debug vtemplate 715
debug vtemplate subinterface 718
debug vtsp 720
debug vtsp all 723
debug vtsp dsp 729
debug vtsp error 731
debug vtsp event 733
debug vtsp port 736
debug vtsp rtp 739
debug vtsp send-nse 742
debug vtsp session 744
debug vtsp stats 748
debug vtsp tone 750

debug vtsp vofr subframe 751
debug vwic-mft firmware controller 753
debug vxml 756
debug waas 764
debug waas accelerator cifs-express 767
debug waas accelerator http-express 769
debug waas accelerator ssl-express 771
debug warm-reboot 773
debug wccp 775
debug webvpn 778
debug webvpn dtls 784
debug webvpn license 786
debug wlccp ap 788
debug wlccp ap rm enhanced-neighbor-list 790
debug wlccp packet 791
debug wlccp rmlib 792
debug wlccp wds 793
debug wsma agent 795
debug wsma profile 797
debug wsapi 799
debug x25 802
debug x25 annexg 811
debug x25 aodi 813
debug x25 interface 815
debug x25 vc 821
debug x25 xot 827
debug x28 833
debug xcctsp all 834
debug xcctsp error 835
debug xcctsp session 836
debug xconnect 837
debug xcsp 839
debug xdsl application 842
debug xdsl driver 845
debug xdsl eoc 847

debug xdsl error 849

debug xmpp profile 851

debug zone 852

show memory debug incremental 854

set memory debug incremental starting-time 857

show memory debug leaks 859

show memory debug references 867

show memory debug unused 869

show crypto debug-condition 871

show debugging 874

show debugging condition 877

voice call debug 879

debug saa apm through debug snmp sync

- debug saa apm, page 3
- debug saa slm, page 5
- debug saa xml, page 7
- debug sampler, page 8
- debug satellite, page 10
- debug satellite firmware, page 14
- debug sccp, page 21
- debug sccp config, page 25
- debug qbm, page 27
- debug sdlc, page 29
- debug sdlc local-ack, page 33
- debug sdlc packet, page 35
- debug serial interface, page 36
- debug serial lead-transition, page 42
- debug serial packet, page 44
- debug service-group, page 45
- debug service-module, page 46
- debug sgbp dial-bids, page 47
- debug sgbp error, page 48
- debug sgbp hellos, page 50
- debug sgcp, page 52
- debug sgcp errors, page 54
- debug sgcp events, page 56
- debug sgcp packet, page 60

- debug shared-line, page 64
- debug smrp all, page 67
- debug smrp group, page 69
- debug smrp mcache, page 72
- debug smrp neighbor, page 74
- debug smrp port, page 76
- debug smrp route, page 78
- debug smrp transaction, page 80
- debug snasw dlc, page 82
- debug snasw ips, page 84
- debug snmp bulkstat, page 86
- debug snmp detail, page 89
- debug snmp mib nhrp, page 91
- debug snmp overhead, page 94
- debug snmp packet, page 95
- debug snmp requests, page 98
- debug snmp sync, page 99
- debug snmp tunnel-mib, page 101

debug saa apm

Note Effective with Cisco IOS Release 12.3(14)T, the **debug saa apm**command is replaced by the **debug ip sla monitor apm**command. See the **debug ip sla monitor apm**command for more information.

To enable debugging output for Cisco IOS IP Service Level Agreements (SLAs) Application Performance Monitor (APM) operations, use the **debug saa apm** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug saa apm

no debug saa apm

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(2)T	This command was introduced.
12.3(14)T	This command was replaced by the **debug ip sla monitor apm**command.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Examples

The following is sample output from the **debug saa apm** command:

```
Router# debug saa apm
Router# configure terminal
Router(config)# saa apm operation 123 start ftp://apm/config/iptv.cf
21:40:27: SAA-APM-123: downloading file (apm/config/iptv.cf) of size (534)
21:40:29: SAA-APM-123: downloading file (apm/scheduler/master.sch) of size (2500)
21:40:30: SAA-APM-123: downloading file (apm/scripts/iptv.scr) of size (1647)
21:40:32: SAA-APM-123: downloading file (apm/data/iptv.dat) of size (118)
21:40:32: SAA-APM-123: sending APM_CAPABILITIES_REQUEST message
21:40:32: sending control msg:
21:40:32: Ver: 1 ID: 29 Len: 48
21:40:32: SAA-APM-123: apm_engine version: major<1>, minor<0>
21:40:32: SAA-APM-123: sending APM_SCRIPT_DNLD message
21:40:32: sending control msg:
21:40:32: Ver: 1 ID: 30 Len: 148
21:40:37: SAA-APM-123: sending APM_SCRIPT_DNLD_STATUS message
21:40:37: sending control msg:
21:40:37: Ver: 1 ID: 31 Len: 148
21:40:38: SAA-APM-123: starting the operation
21:40:38: SAA-APM-123: sending APM_SCRIPT_START message
21:40:38: sending control msg:
21:40:38: Ver: 1 ID: 32 Len: 148
21:40:41: SAA-APM: 0,2144,0
```

```
.
.
.
21:49:42: SAA-APM-123: waiting for ageout timer to expire
21:55:13: SAA-APM-123: sending APM_SCRIPT_DONE message
21:55:13: sending control msg:
21:55:13: Ver: 1 ID: 42 Len: 148
21:55:13: SAA-APM-123: operation done
Router(config)# no saa apm
21:55:13: SAA-APM-123: sending APM_SCRIPT_DONE message
21:55:13: sending control msg:
21:55:13: Ver: 1 ID: 42 Len: 148
21:55:13: SAA-APM-123: operation done
```

debug saa slm

Note Effective with Cisco IOS Release 12.3(14)T, the **debug saa slm**command is replaced by the **debug ip sla monitor slm**command. See the **debug ip sla monitor slm**command for more information.

To enable debugging output of detailed event messages for Cisco IOS IP Service Level Agreements (SLAs) Service Level Monitoring (SLM) Asynchronous Transfer Mode (ATM) operations, use the **debug saa slm**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug saa slm

no debug saa slm

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(11)T	This command was introduced.
12.3(14)T	This command was replaced by the **debug ip sla monitor slm**command.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

IP SLAs SLM ATM performance statistics cannot be retrieved from Cisco IOS devices using Simple Network Management Protocol (SNMP). The IP SLAs SLM ATM feature was designed to provide data by responding to extensible markup language (XML) requests.

Note This command may generate a large number of debugging messages.

Examples

In the following example, debugging is enabled for the IP SLAs SLM ATM feature and the IP SLAs XML feature for the purposes of debugging the XML requests and responses:

```
debug saa slm
debug saa xml
```

Related Commands

Command	Description
debug saa xml	Enables debugging output of XML requests and responses for IP SLAs operations.

debug saa xml

Note Effective with Cisco IOS Release 12.3(14)T, the **debug saa xml**command is replaced by the **debug ip sla monitor xml**command. See the **debug ip sla monitor xml**command for more information.

To enable debugging output of eXtensible Markup Language (XML) requests and responses for Cisco IOS IP Service Level Agreements (SLAs) operations, use the **debug saa xml**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug saa xml

no debug saa xml

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(11)T	This command was introduced.
12.3(14)T	This command was replaced by the **debug ip sla monitor xml**command.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Examples

In the following example, debugging is enabled for the IP SLAs SLM ATM feature and the IP SLAs eXtensible Markup Language (XML) feature for the purposes of debugging the XML requests and responses:

```
debug saa slm
debug saa xml
```

Related Commands

Command	Description
debug saa slm	Enables debugging output of detailed event messages for IP SLAs SLM ATM operations.

debug sampler

To enable debugging output for Flexible NetFlow samplers, use the **debug sampler** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug sampler [**detailed**| **error**| **[name]** *sampler-name* [**detailed**| **error**| **sampling** *samples*]]

no debug sampler [**detailed**| **error**| **[name]** *sampler-name* [**detailed**| **error**| **sampling**]]

Syntax Description

detailed	(Optional) Enables detailed debugging for sampler elements.
error	(Optional) Enables debugging for sampler errors.
name	(Optional) Specifies the name of a sampler.
sampler-name	(Optional) Name of a sampler that was previously configured.
sampling *samples*	(Optional) Enables debugging for sampling and specifies the number of samples to debug.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.4(9)T	This command was introduced.
12.2(31)SB2	This command was integrated into Cisco IOS Release 12.2(31)SB2.
12.0(33)S	This command was implemented on the Cisco 12000 series routers.
12.2(33)SRC	Support for this command was added for Cisco 7200 series routers.
12.2(33)SRE	This command was integrated into Cisco IOS Release 12.2(33)SRE for the Cisco 7300 Network Processing Engine (NPE) series routers.
12.2(50)SY	This command was integrated into Cisco IOS Release 12.2(50)SY.

Examples

The following sample output shows that the debug process has obtained the ID for the sampler named SAMPLER-1:

```
Router# debug sampler detailed
```

```
*Oct 28 04:14:30.883: Sampler: Sampler(SAMPLER-1: flow monitor FLOW-MONITOR-1 (ip,Et1/0,O)
 get ID succeeded:1
*Oct 28 04:14:30.971: Sampler: Sampler(SAMPLER-1: flow monitor FLOW-MONITOR-1 (ip,Et0/0,I)
 get ID succeeded:1
```

Related Commands

Command	Description
clear sampler	Clears the Flexible NetFlow sampler statistics.

debug satellite

To enable debugging output for the Cisco IP VSAT satellite WAN network module (NM-1VSAT-GILAT), use the **debug satellite** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug satellite {**all**| **errors**| **events**| **hsrp**| **rbcp**}

no debug satellite {**all**| **errors**| **events**| **hsrp**| **rbcp**}

Syntax Description

all	Displays all types of satellite debug information.
errors	Displays debug information for satellite error events.
events	Displays debug information for software events.
hsrp	Displays debug information for satellite Hot Standby Router Protocol (HSRP) events.
rbcp	Displays debug information for satellite Router Blade Control Protocol (RBCP) messages.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

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

Usage Guidelines

The **debug satellite errors** command is useful for catching unusual conditions when troubleshooting unexpected behavior. Because this command typically generates very little output, you can enter the **debug satellite errors** command every time you troubleshoot satellite network connectivity.

Examples

This section provides the following examples:

Examples

Every 2 minutes, the NM-1VSAT-GILAT network module sends the router an RBCP message requesting any updates to the routing table. The following example shows how to monitor the route-update messages:

```
Router# debug satellite rbcp
```

...
The NM-1VSAT-GILAT network module requests IP route information:

```
*May 16 09:18:54.475:Satellite1/0 RBCP Request  msg Recd:IPROUTE_REQ(0x22)
```

The Cisco IOS software acknowledges that it received the message from the NM-1VSAT-GILAT network module:

```
*May 16 09:18:54.475:Satellite1/0 RBCP Response msg Sent:IPROUTE_REQ(0x22)
```

The Cisco IOS software sends the IP route information to the NM-1VSAT-GILAT network module:

```
*May 16 09:18:54.475:Satellite1/0 RBCP Request  msg Sent:IPROUTE_UPD(0x23)
```

The NM-1VSAT-GILAT network module acknowledges that it received the routing update from the Cisco IOS software:

```
*May 16 09:18:54.475:Satellite1/0 RBCP Response msg Recd:IPROUTE_UPD(0x23)
```

Examples

The following example shows how to monitor the periodic heartbeats that the NM-1VSAT-GILAT network module sends to the Cisco IOS software:

```
Router# debug satellite events

satellite major software events debugging is on
.Dec 16 12:57:52.108:Satellite1/0 FSM transition LINK_UP-->LINK_UP, ev=got_heartbeat
.Dec 16 12:58:08.888:Satellite1/0 FSM transition LINK_UP-->LINK_UP, ev=got_heartbeat
.Dec 16 12:58:25.664:Satellite1/0 FSM transition LINK_UP-->LINK_UP, ev=got_heartbeat
.Dec 16 12:58:42.440:Satellite1/0 FSM transition LINK_UP-->LINK_UP, ev=got_heartbeat
```

Examples

The following example shows the **debug satellite hsrp** command messages that appear when the active router is forced to standby status because the HSRP-tracked satellite interface is shut down:

```
Router# configure terminal

Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# interface satellite 1/0

Router(config-if)# shutdown

Router(config-if)# end

Router#
01:03:48:%SYS-5-CONFIG_I:Configured from console by console
01:03:49:%LINK-5-CHANGED:Interface Satellite1/0, changed state to administratively down
01:03:50:%LINEPROTO-5-UPDOWN:Line protocol on Interface Satellite1/0, changed state to down
01:04:22:%HSRP-6-STATECHANGE:FastEthernet0/0 Grp 1 state Active -> Speak
01:04:22:HSRP-sat:IPred group grp-x update state ACTIVE --> SPEAK
01:04:22:Satellite1/0 HSRP-sat:fsm crank ACTIVE-->STANDBY
01:04:22:Satellite1/0 HSRP-sat:send standby msg STANDBY
01:04:32:HSRP-sat:IPred group grp-x update state SPEAK --> STANDBY
01:04:32:Satellite1/0 HSRP-sat:fsm crank STANDBY-->STANDBY
01:04:32:Satellite1/0 HSRP-sat:send standby msg STANDBY
01:04:42:Satellite1/0 HSRP-sat:send standby msg STANDBY
01:04:52:Satellite1/0 HSRP-sat:standby msg STANDBY deferred, not in operational state
01:05:02:Satellite1/0 HSRP-sat:standby msg STANDBY deferred, not in operational state
01:05:12:Satellite1/0 HSRP-sat:standby msg STANDBY deferred, not in operational state
01:05:22:Satellite1/0 HSRP-sat:standby msg STANDBY deferred, not in operational state
01:05:32:Satellite1/0 HSRP-sat:standby msg STANDBY not sent, already in state
01:06:47:%VSAT-5-STANDBY_MODE:Satellite1/0 module configured for standby mode
01:09:32:Satellite1/0 HSRP-sat:fsm crank STANDBY-->STANDBY-UP
```

Examples

The following example shows HSRP-related debug output for both the router and the NM-1VSAT-GILAT network module when the router goes from standby to active state because the HSRP-tracked satellite interface is reenabled:

```
Router# show debugging

SATCOM:
  satellite HSRP events debugging is on
HSRP:
  HSRP Errors debugging is on
  HSRP Events debugging is on
  HSRP Packets debugging is on
```

The satellite interface is reenabled:

```
Router# configure terminal

Router(config)# interface satellite 1/0

Router(config-if)# no shutdown

Router(config-if)# end

Router#
```

The effective HSRP priority of the router changes as the tracked satellite interface comes up:

```
02:14:37:HSRP:Fa0/0 Grp 1 Hello  in  10.123.96.2 Active  pri 90 vIP 10.123.96.100
02:14:39:HSRP:Fa0/0 API 10.1.0.6 is not an HSRP address
02:14:39:HSRP:Fa0/0 Grp 1 Hello  out 10.123.96.3 Standby pri 90 vIP 10.123.96.100
02:14:39:HSRP:Fa0/0 Grp 1 Track 1 object changed, state Down -> Up
02:14:39:HSRP:Fa0/0 Grp 1 Priority 90 -> 100
Router#
```

The router changes from standby to active state because its priority is now highest in the hot standby group, and preemption is enabled:

```
02:14:40:HSRP:Fa0/0 Grp 1 Hello  in  10.123.96.2 Active  pri 90 vIP 10.123.96.100
02:14:40:HSRP:Fa0/0 Grp 1 Standby:h/Hello rcvd from lower pri Active router (90/10.123.96.2)
02:14:40:HSRP:Fa0/0 Grp 1 Active router is local, was 10.123.96.2
02:14:40:HSRP:Fa0/0 Grp 1 Standby router is unknown, was local
02:14:40:HSRP:Fa0/0 Redirect adv out, Active, active 1 passive 3
02:14:40:HSRP:Fa0/0 Grp 1 Coup   out 10.123.96.3 Standby pri 100 vIP 10.123.96.100
02:14:40:HSRP:Fa0/0 Grp 1 Standby -> Active
02:14:40:%HSRP-6-STATECHANGE:FastEthernet0/0 Grp 1 state Standby -> Active
```

The HSRP status of the satellite interface also changes from standby to active state because the **service-module ip redundancy** command was previously entered to link the HSRP status of the satellite interface to the primary HSRP interface, Fast Ethernet 0/0.

```
02:14:40:HSRP:Fa0/0 Grp 1 Redundancy "grp-x" state Standby -> Active
02:14:40:HSRP-sat:IPred group grp-x update state STANDBY --> ACTIVE
02:14:40:Satellite1/0 HSRP-sat:fsm crank STANDBY-UP-->ACTIVE-COND
02:14:40:HSRP:Fa0/0 Redirect adv out, Active, active 1 passive 2
02:14:40:HSRP:Fa0/0 Grp 1 Hello  out 10.123.96.3 Active  pri 100 vIP 10.123.96.100
02:14:40:HSRP:Fa0/0 REDIRECT adv in, Passive, active 0, passive 2, from 10.123.96.2
02:14:40:HSRP:Fa0/0 REDIRECT adv in, Passive, active 0, passive 1, from 10.123.96.15
02:14:40:HSRP:Fa0/0 Grp 1 Hello  in  10.123.96.2 Speak   pri 90 vIP 10.123.96.100
```

Line protocols come up, and HSRP states become fully active:

```
02:14:41:%LINK-3-UPDOWN:Interface Satellite1/0, changed state to up
02:14:42:%LINEPROTO-5-UPDOWN:Line protocol on Interface Satellite1/0, changed state to up
02:14:43:HSRP:Fa0/0 Grp 1 Hello  out 10.123.96.3 Active  pri 100 vIP 10.123.96.100
02:14:43:HSRP:Fa0/0 Grp 1 Redundancy group grp-x state Active -> Active
02:14:43:HSRP-sat:IPred group grp-x update state ACTIVE --> ACTIVE
02:14:43:Satellite1/0 HSRP-sat:fsm crank ACTIVE-COND-->ACTIVE-COND
```

```
02:14:43:HSRP:Fa0/0 Grp 1 Hello  in  10.123.96.2 Speak   pri 90 vIP 10.123.96.100
02:14:46:HSRP:Fa0/0 Grp 1 Hello  out 10.123.96.3 Active  pri 100 vIP 10.123.96.100
02:14:46:HSRP:Fa0/0 Grp 1 Redundancy group grp-x state Active -> Active
02:14:46:HSRP-sat:IPred group grp-x update state ACTIVE --> ACTIVE
02:14:46:Satellite1/0 HSRP-sat:fsm crank ACTIVE-COND-->ACTIVE-COND
02:14:46:HSRP:Fa0/0 Grp 1 Hello  in  10.123.96.2 Speak   pri 90 vIP 10.123.96.100
02:14:49:HSRP:Fa0/0 Grp 1 Hello  out 10.123.96.3 Active  pri 100 vIP 10.123.96.100
02:14:49:HSRP:Fa0/0 Grp 1 Hello  in  10.123.96.2 Speak   pri 90 vIP 10.123.96.100
02:14:50:HSRP:Fa0/0 Grp 1 Hello  in  10.123.96.2 Standby pri 90 vIP 10.123.96.100
02:14:50:HSRP:Fa0/0 Grp 1 Standby router is 10.123.96.2
02:14:51:Satellite1/0 HSRP-sat:send standby msg ACTIVE
02:14:52:HSRP:Fa0/0 Grp 1 Hello  out 10.123.96.3 Active  pri 100 vIP 10.123.96.100
02:14:53:HSRP:Fa0/0 Grp 1 Hello  in  10.123.96.2 Standby pri 90 vIP 10.123.96.100
02:14:55:HSRP:Fa0/0 Grp 1 Hello  out 10.123.96.3 Active  pri 100 vIP 10.123.96.100
```

Related Commands

Command	Description
debug satellite firmware	Enables debugging output for the Cisco IP VSAT satellite WAN network module (NM-1VSAT-GILAT) firmware.
debug standby	Displays all HSRP errors, events, and packets.

debug satellite firmware

To enable debugging output for the Cisco IP VSAT satellite WAN network module (NM-1VSAT-GILAT) firmware, use the **debug satellite firmware**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug satellite firmware {**all**| **level** *number*| *option*}

no debug satellite firmware

Syntax Description

all	Displays all satellite firmware events.
level *number*	Satellite debug level. The debug level affects what information is displayed for subsequently entered **debug satellite firmware** commands. See the table below.
option	One of the following options. See the table below. • **bb** --Satellite backbone events • **buf** --Satellite buffer events • **en** --Satellite firmware encryption events • **ip** --Satellite IP events • **rbcp** --Satellite RBCP events • **rpa** --Satellite Remote Page Acceleration (RPA) events • **sat** --Satellite inbound and outbound packet statistics • **tcp** --Satellite TCP events • **trc** --Satellite backbone traces

Command Default

No default behavior or values.

Command Modes

Privileged EXEC

Command History

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

Usage Guidelines

The output from this command is generally useful for diagnostic tasks performed by technical support.

The level number affects which debug messages the system displays for subsequently entered **debug satellite firmware** commands. The table below describes what each command option displays at each debug level.

Note Level 3 debugging produces significant amounts of output that may negatively impact the performance of both the NM-1VSAT-GILAT network module and the router. When you enter debug level 3, a warning message and confirmation prompt appear.

Table 1: debug satellite firmware Command Level Options

Option	Level 1 Output	Level 2 Output	Level 3 Output
bb	Backbone link information	Frame statistics for the backbone link to the hub	--
buf	Buffer information	Buffer owners	--
en	Satellite firmware-based encryption events	--	--
ip	IP statistics	--	Driver transmission statistics
rbcp	Number of transmitted and received RBCP messages	--	Satellite Control Protocol (SCP) message summaries
rpa	RPA statistics	Tunnel connect and disconnect events	--
tcp	TCP statistics	TCP connection information	TCP statistics and TCP connection information
sat	Inbound and outbound packet statistics	Inbound and outbound packet statistics	Inbound and outbound packet statistics
trc	--	--	Backbone receive and transmit traces

Examples

This section provides the following sample output for the **debug satellite firmware**command:

Examples

The following example shows all satellite firmware events and statistics:

```
Router# debug satellite firmware all
2d06h: Satellite2/0
```

```
buffers 4856 min 4486 list_str 683798 list_end 6885c8
emp 686030 fil 685de0 start 6885c8 end fb4fe8
2d06h: Satellite2/0
TCP stats: NetRXBytes=223 NetTXBytes=4775126 NetRxPkts=104213 ToIOSPkts=104166
2d06h: Satellite2/0
SAT stats: OUTbound_pkts=114131, INbound_pkts=182347
2d06h: Satellite2/0
RBCP statistics: TXcount=975 RXCount=975
2d06h: Satellite2/0
RPA stats: ToTunnel=0 FromTunnel=0
TunnelGets=0 TunnelNotGets=0
BlksUsed=0 BlksIn-Use=0 Max=300
2d06h: Satellite2/0
EN:
RX encrypted bytes received = 0
RX: compressed=0 -> Uncompressed=0
TX: compressed=0 -> Uncompressed=0
2d06h: Satellite2/0
BB 6 LINK state=INFO_STATE
     Status = 0x79,  LOW NOT READY,  HI PRI READY
     RSP Q free=230, Max HI=228, Max LOW=224, Max DG=232
     IN RA mode
     Curr DG BW=50000, HighDG BW=100000, Curr BW=98094
    MaxDG BW=1250000, Max BW=2500000
     PD Queue lengths:
        q_wtog=0, q_wtos=57,  q_wtos_high=0, q_defrag=d
     DG Queue lengths:
        q_dg_wtos=0, q_dg_wtos_hi=0, q_dg_defrag=0
     Congestion Levels:        TX LOCAL = 7, TX NET = 0
2d06h: Satellite2/0
IP stats: ToIOS_Pkts=234193, ToIOS_Bytes=183444492 FromIOS_Pkts=143 From_IOS_Bytes=12204
2d06h: Satellite2/0 NO Trace at levels 1 or 2
2d06h: Satellite2/0 NO Trace at levels 1 or 2
```

Examples

The following example shows backbone link information:

```
Router# debug satellite firmware level 1

Router# debug satellite firmware bb

satellite BackBone events debugging is on
Router#
2d06h: Satellite2/0
BB 6 LINK state=INFO_STATE
     Status = 0x79,  LOW NOT READY,  HI PRI READY
     RSP Q free=240, Max HI=228, Max LOW=224, Max DG=232
     IN RA mode
     Curr DG BW=50000, HighDG BW=100000, Curr BW=96188
    MaxDG BW=1250000, Max BW=2500000
     PD Queue lengths:
        q_wtog=0, q_wtos=95,  q_wtos_high=0, q_defrag=d
     DG Queue lengths:
        q_dg_wtos=0, q_dg_wtos_hi=0, q_dg_defrag=0
     Congestion Levels:        TX LOCAL = 7, TX NET = 0
2d06h: Satellite2/0
BB 6 LINK state=INFO_STATE
     Status = 0x7b,  LOW READY,  HI PRI READY
     RSP Q free=27, Max HI=228, Max LOW=224, Max DG=232
     IN RA mode
     Curr DG BW=50000, HighDG BW=100000, Curr BW=92376
    MaxDG BW=1250000, Max BW=2500000
     PD Queue lengths:
        q_wtog=0, q_wtos=24,  q_wtos_high=0, q_defrag=d
     DG Queue lengths:
        q_dg_wtos=0, q_dg_wtos_hi=0, q_dg_defrag=0
     Congestion Levels:        TX LOCAL = 4, TX NET = 0
```

Examples

The following example shows frame statistics for the backbone link to the hub:

```
Router# debug satellite firmware level 2

Router# debug satellite firmware bb

satellite BackBone events debugging is on
Router#
2d06h: Satellite2/0 BB link statistics
    Frame Type          # Received      # Transmitted
  ------------          ----------      -------------
   INFORMATION           00096238          00184811
   UNNUMBERED            00000000          00000067
   RETRANSMITTED         00000000          00000000
   POLLS                 00000000          00000000
   ACKS                  00006640          00000455
   NAKS                  00000000          00000000
   PACKS                 00000000          00000000
   UA                    00000001          00000000
   SABME                 00000000          00000001
   DISC                  00000000          00000000
```

Examples

The following example shows buffer information:

```
Router# debug satellite firmware level 1

Router# debug satellite firmware buf

*May 13 15:58:54.498:Satellite1/0
buffers 4951 min 4945 list_str 681858 list_end 686688
emp 683abc fil 6839e8 start 686688 end fb30a8
```

Examples

The following example shows buffer owners:

```
Router# debug satellite firmware level 2

Router# debug satellite firmware buf

*May 13 15:59:13.438:Satellite1/0 inuse 49 free 4951
Trace byte  1
Trace byte = 0x169    Count =   49
Trace byte  2
Trace byte = 0x  0    Count =   49
   0 buffers with BB Rel only
   0 buffers with in lower layer set
   0 buffers with do not transmit set
   0 buffers on BB retransmit queues
```

Examples

The following example shows IP statistics:

```
Router# debug satellite firmware level 1

Router# debug satellite firmware ip

*Nov  7 08:27:56.440: Satellite3/0
IP stats: ToIOS_Pkts=0, ToIOS_Bytes=0 FromIOS_Pkts=84751 From_IOS_Bytes=5941124
```

Examples

The following example shows the number of RBCP messages transmitted and received since the most recent reset of the Cisco IOS software on the router or the VSAT software on the NM-1VSAT-GILAT network module:

```
Router# debug satellite firmware level 1
```

```
Router# debug satellite firmware rbcp
```

```
RBCP statistics:TXcount=301154 RXCount=301155
```

Examples

The following example shows RPA statistics:

```
Router# debug satellite firmware level 1
```

```
Router# debug satellite firmware rpa
```

```
  *Nov  7 08:27:13.488:Satellite3/0
  RPA stats:ToTunnel=0 FromTunnel=0
  TunnelGets=0 TunnelNotGets=0
  BlksUsed=0 BlksIn-Use=0 Max=400
```

Examples

The following example shows a tunnel being disconnected:

```
Router# debug satellite firmware level 2
```

```
Router# debug satellite firmware rpa
```

```
*May 13 18:27:59.779:Satellite1/0 RPA Tunnel DOWN
RPA:InitTunnelConn Successful locIP e000006 locPort 1090, RemIP c0a80186,
RemPort 9876
RPA Tunnel DOWN
RPA:InitTunnelConn Successful locIP e000006 locPort 1091, RemIP c0a80186,
RemPort 9876
RPA Tunnel DOWN
RPA:InitTunnelConn Successful locIP e000006 locPort 1092, RemIP c0a80186,
RemPort 9876
RPA Tunnel DOWN
RPA:InitTunnelConn Successful locIP e000006 locPort 1093, RemIP c0a80186,
RemPort 9876
RPA Tunnel DOWN
RPA:InitTunnelConn Successful locIP e000006 locPort 1094, RemIP c0a80186,
RemPort 9876
```

Examples

The following example shows inbound and outbound packet statistics. Note that for all levels, the debug output is the same for the **sat** option.

```
Router# debug satellite firmware level 1
```

```
Router# debug satellite firmware sat
```

```
satellite related trace events debugging is on
Router#
1d16h: Satellite2/0
SAT stats: OUTbound_pkts=25660796, INbound_pkts=3235932
1d16h: Satellite2/0
SAT stats: OUTbound_pkts=25660800, INbound_pkts=3235934
1d16h: Satellite2/0
SAT stats: OUTbound_pkts=25660803, INbound_pkts=3235934
```

```
1d16h: Satellite2/0
SAT stats: OUTbound_pkts=25660803, INbound_pkts=3235934
```

Examples

The following example shows TCP statistics:

```
Router# debug satellite firmware level 1

Router# debug satellite firmware tcp

satellite tcp events debugging is on
Router#
2d06h: Satellite2/0
TCP stats: NetRXBytes=631292 NetTXBytes=4009436 NetRxPkts=49244 ToIOSPkts=49246
2d06h: Satellite2/0
TCP stats: NetRXBytes=1154356 NetTXBytes=4086106 NetRxPkts=49621 ToIOSPkts=49629
```

Examples

The following example shows the TCP connections:

```
Router# debug satellite firmware level 2

Router# debug satellite firmware tcp

satellite tcp events debugging is on
Router#
2d06h: Satellite2/0 TCP connections:
ID=48, locIP=192.168.107.2 remIP=172.25.1.2, locP=2962, remP=21 state=17 iosQ=0
ID=49, locIP=192.168.107.2 remIP=172.25.1.2, locP=2963, remP=20 state=17 iosQ=0
ID=58, locIP=192.168.107.2 remIP=172.25.1.28, locP=2972, remP=21 state=17 iosQ=0
ID=59, locIP=192.168.107.2 remIP=172.25.1.28, locP=2973, remP=20 state=17 iosQ=7
2d06h: Satellite2/0 TCP connections:
ID=48, locIP=192.168.107.2 remIP=172.25.1.2, locP=2962, remP=21 state=17 iosQ=0
ID=49, locIP=192.168.107.2 remIP=172.25.1.2, locP=2963, remP=20 state=7 iosQ=0
ID=60, locIP=192.168.107.2 remIP=172.25.1.28, locP=2974, remP=21 state=3 iosQ=0
```

Examples

The following example shows TCP statistics and connections:

```
Router# debug satellite firmware level 3

Output may be extensive and affect performance. Continue? [yes]: yes

Router# debug satellite firmware tcp

satellite tcp events debugging is on
Router#
2d06h: Satellite2/0
TCP stats: NetRXBytes=279 NetTXBytes=9436111 NetRxPkts=64991 ToIOSPkts=64999
2d06h: Satellite2/0 TCP connections:
ID=48, locIP=192.168.107.2 remIP=172.25.1.2, locP=2962, remP=21 state=7 iosQ=0
ID=49, locIP=192.168.107.2 remIP=172.25.1.2, locP=2963, remP=20 state=7 iosQ=0
ID=62, locIP=192.168.107.2 remIP=172.25.1.28, locP=2976, remP=21 state=7 iosQ=0
2d06h: Satellite2/0
TCP stats: NetRXBytes=382 NetTXBytes=9582924 NetRxPkts=64993 ToIOSPkts=65001
2d06h: Satellite2/0 TCP connections:
ID=48, locIP=192.168.107.2 remIP=172.25.1.2, locP=2962, remP=21 state=17 iosQ=0
ID=49, locIP=192.168.107.2 remIP=172.25.1.2, locP=2963, remP=20 state=17 iosQ=0
ID=62, locIP=192.168.107.2 remIP=172.25.1.28, locP=2976, remP=21 state=7 iosQ=0
```

Examples

The following example shows detailed receive and transmit traces for the backbone link:

```
Router# debug satellite firmware level 3

Output may be extensive and affect performance. Continue? [yes]: yes
```

```
Router# debug satellite firmware trc

satellite BackBone trace debugging is on
Router#
2d06h: Satellite2/0 strrec 0, rec 0, count 256, trc 1a6dd78, str 1a5c600, end 1a
74600
count 4096, emp 1a6dd78, fil 1a6d8b0, lnknum=6
   0 xmt  6 len  951  9 pd    con 0 PF  3 ns  169 nr   15  a c12 0   0.000
   1 xmt  6 len  951  9 pd    con 0 PF  3 ns  170 nr   15  a c12 0   0.010
   2 xmt  6 len  951  9 pd    con 0 PF  3 ns  171 nr   15  a c12 0   0.010
   3 xmt  6 len  951  9 pd    con 0 PF  3 ns  172 nr   15  a c12 0   0.010
   4 xmt  6 len  951  9 pd    con 0 PF  3 ns  173 nr   15  a c12 0   0.030
   5 xmt  6 len
2d06h: Satellite2/0  951
2d06h: Satellite2/0  9 pd    con 0 PF  3 ns  174 nr   15  a c12 0   0.010
   6 xmt  6 len  951  9 pd    con 0 PF  3 ns  175 nr   15  a c12 0   0.010
   7 xmt  6 len  951  9 pd    con 0 PF  3 ns  176 nr   15  a c12 0   0.010
   8 xmt  6 len  951  9 pd    con 0 PF  3 ns  177 nr   15  a c12 0   0.010
   9 xmt  6 len  951  9 pd    con 0 PF  3 ns  178 nr   15  a c12 0   0.010
  10 xmt  6 len  951  9 pd    con 0 PF  3 ns  179 nr   15  a c12 0   0.010
  11 xmt  6 len  951  9 pd    con 0 PF  3 ns  180 nr   15  a c12 0   0.010
```

Related Commands

Command	Description
debug satellite	Enables debugging output for the Cisco IP VSAT satellite WAN network module (NM-1VSAT-GILAT).

debug sccp

To display debugging information for Simple Client Control Protocol (SCCP) and its related applications (transcoding and conferencing), use the **debug sccp**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug sccp {**all**| **errors**| **events**| **packets**| **parser**}

no debug sccp

Syntax Description

all	All SCCP debug-trace information.
errors	SCCP errors.
events	SCCP events.
packets	SCCP packets.
parser	SCCP parser and builder.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.1(5)YH	This command was introduced on the Cisco VG200.
12.2(13)T	This command was implemented on the Cisco 2600 series, Cisco 3620, Cisco 3640, Cisco 3660, and Cisco 3700 series.

Usage Guidelines

The router on which this command is used must be equipped with one or more digital T1/E1 packet voice trunk network modules (NM-HDVs) or high-density voice (HDV) transcoding and conferencing digital signal processor (DSP) farms (NM-HDV-FARMs) to provide DSP resources.

Debugging is turned on for all DSP farm service sessions. You can debug multiple sessions simultaneously, with different levels of debugging for each.

Examples

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

```
Router# debug sccp events

Skinny Client Control Protocol events debugging is on
*Mar  1 00:46:29: sccp_create_application: send keepalive msg, appl 6248F760, appl_type 1,
 count 0
*Mar  1 00:46:29: sccp_keepalive: send keepalive id 0, len 4
```

```
*Mar  1 00:46:29: sccp_process_mtp_pdu: appl - 6248F760, mbuf - 6248F7D4
*Mar  1 00:46:29: sccp_process_mtp_pdu: msg_ptr 6248F7DC, len 4, offset 12, msg_id 256
*Mar  1 00:46:30: sccp_create_application: send keepalive msg, appl 6248FC10, appl_type 2,
 count 0
*Mar  1 00:46:30: sccp_keepalive: send keepalive id 0, len 4
*Mar  1 00:46:30: sccp_process_mtp_pdu: appl - 6248FC10, mbuf - 6248FC84
*Mar  1 00:46:30: sccp_process_mtp_pdu: msg_ptr 6248FC8C, len 4, offset 12, msg_id 256
*Mar  1 00:46:37: sccp_create_application: send keepalive msg, appl 6248F760, appl_type 1,
 count 0
*Mar  1 00:46:37: sccp_keepalive: send keepalive id 0, len 4
*Mar  1 00:46:37: sccp_process_mtp_pdu: appl - 6248F760, mbuf - 6248F7D4
*Mar  1 00:46:37: sccp_process_mtp_pdu: msg_ptr 6248F7DC, len 4, offset 12, msg_id 256
*Mar  1 00:46:37: sccp_create_application: send keepalive msg, appl 6248FC10, appl_type 2,
 count 0
*Mar  1 00:46:37: sccp_keepalive: send keepalive id 0, len 4
*Mar  1 00:46:38: sccp_process_mtp_pdu: appl - 6248FC10, mbuf - 6248FC84
*Mar  1 00:46:38: sccp_process_mtp_pdu: msg_ptr 6248FC8C, len 4, offset 12, msg_id 256
*Mar  1 00:46:43: sccp_process_mtp_pdu: appl - 6248FC10, mbuf - 6248FC84
*Mar  1 00:46:43: sccp_process_mtp_pdu: msg_ptr 6248FC8C, len 28, offset 36, msg_id 261
*Mar  1 00:46:43: xapp_open_receive_chnl: SCCP orc_msg - 6248FC8C, appl - 6248FC10
*Mar  1 00:46:43: xapp_search_for_chnl_rec: sess_id 27, conn_id 2769
*Mar  1 00:46:43: xapp_add_chnl_rec: chnl 631142BC
*Mar  1 00:46:43: xapp_add_sess_rec: Add sess_rec (63114360) record
*Mar  1 00:46:43: xapp_open_receive_chnl: stat 0, eve 0, sid 27, cid 2769, codec 1, pkt-period
 20
*Mar  1 00:46:43: xapp_open_chnl_request: chnl_rec 631142BC
*Mar  1 00:46:43: xapp_open_chnl_request: chnl_rec 631142BC, sess_id 27, conn_id 2769,
cstate 0, nstate 1
*Mar  1 00:46:43: xapp_dequeue_and_process_dspf_events: chnl_rec 631142BC, state 1, eve_id
 1
*Mar  1 00:46:43: xapp_open_chnl_success: chnl_rec 631142BC
*Mar  1 00:46:43: xapp_open_chnl_success: chnl_rec 631142BC, sess_id 27, conn_id 2769,
cstate 1, nstate 2, lc_ipaddr 10.10.1.1, lport 21066
*Mar  1 00:46:43: sccp_process_mtp_pdu: appl - 6248FC10, mbuf - 6248FC84
*Mar  1 00:46:43: sccp_process_mtp_pdu: msg_ptr 6248FC8C, len 28, offset 36, msg_id 261
*Mar  1 00:46:43: xapp_open_receive_chnl: SCCP orc_msg - 6248FC8C, appl - 6248FC10
*Mar  1 00:46:43: xapp_search_for_chnl_rec: sess_id 27, conn_id 2785
*Mar  1 00:46:43: xapp_add_chnl_rec: chnl 631142E4
*Mar  1 00:46:43: xapp_open_receive_chnl: stat 0, eve 0, sid 27, cid 2785, codec 1, pkt-period
 20
*Mar  1 00:46:43: xapp_open_chnl_request: chnl_rec 631142E4
*Mar  1 00:46:43: xapp_open_chnl_request: chnl_rec 631142E4, sess_id 27, conn_id 2785,
cstate 0, nstate 1
*Mar  1 00:46:43: xapp_dequeue_and_process_dspf_events: chnl_rec 631142E4, state 1, eve_id
 1
*Mar  1 00:46:43: xapp_open_chnl_success: chnl_rec 631142E4
*Mar  1 00:46:43: xapp_open_chnl_success: chnl_rec 631142E4, sess_id 27, conn_id 2785,
cstate 1, nstate 2, lc_ipaddr 10.10.1.1, lport 25706
*Mar  1 00:46:43: sccp_process_mtp_pdu: appl - 6248FC10, mbuf - 6248FC84
*Mar  1 00:46:43: sccp_process_mtp_pdu: msg_ptr 6248FC8C, len 44, offset 52, msg_id 138
*Mar  1 00:46:43: xapp_start_media_transmission: SCCP stmt_msg - 6248FC8C, appl - 6248FC10
*Mar  1 00:46:43: xapp_search_for_chnl_rec: sess_id 27, conn_id 2769
*Mar  1 00:46:43: xapp_start_media_transmission: chnl_rec 631142BC, stat 2, sid 27, cid
2769, ripaddr 10.10.1.5, rport 32148, codec 1, pkt-period 20, pre 11, silen 16777500, mfpp
 1
*Mar  1 00:46:43: xapp_modify_chnl_request: chnl_rec 631142BC
*Mar  1 00:46:43: xapp_modify_chnl_request: chnl_rec 631142BC, sess_id 27, conn_id 2769,
cstate 2, nstate 2
*Mar  1 00:46:43: xapp_dequeue_and_process_dspf_events: chnl_rec 631142BC, state 2, eve_id
 4
*Mar  1 00:46:43: xapp_modify_chnl_success: chnl_rec 631142BC, sess_id 27, conn_id 2769,
cstate 2
*Mar  1 00:46:43: sccp_process_mtp_pdu: appl - 6248FC10, mbuf - 6248FC84
*Mar  1 00:46:43: sccp_process_mtp_pdu: msg_ptr 6248FC8C, len 44, offset 52, msg_id 138
*Mar  1 00:46:43: xapp_start_media_transmission: SCCP stmt_msg - 6248FC8C, appl - 6248FC10
*Mar  1 00:46:43: xapp_search_for_chnl_rec: sess_id 27, conn_id 2785
*Mar  1 00:46:43: xapp_start_media_transmission: chnl_rec 631142E4, stat 2, sid 27, cid
2785, ripaddr 10.10.1.7, rport 16422, codec 1, pkt-period 20, pre 11, silen 16777501, mfpp
 1
*Mar  1 00:46:43: xapp_modify_chnl_request: chnl_rec 631142E4
*Mar  1 00:46:43: xapp_modify_chnl_request: chnl_rec 631142E4, sess_id 27, conn_id 2785,
cstate 2, nstate 2
*Mar  1 00:46:43: xapp_dequeue_and_process_dspf_events: chnl_rec 631142E4, state 2, eve_id
```

```
 4
*Mar  1 00:46:43: xapp_modify_chnl_success: chnl_rec 631142E4, sess_id 27, conn_id 2785,
cstate 2
*Mar  1 00:46:44: sccp_create_application: send keepalive msg, appl 6248F760, appl_type 1,
 count 0
*Mar  1 00:46:44: sccp_keepalive: send keepalive id 0, len 4
*Mar  1 00:46:45: sccp_process_mtp_pdu: appl - 6248F760, mbuf - 6248F7D4
*Mar  1 00:46:45: sccp_process_mtp_pdu: msg_ptr 6248F7DC, len 4, offset 12, msg_id 256
*Mar  1 00:46:45: sccp_create_application: send keepalive msg, appl 6248FC10, appl_type 2,
 count 0
*Mar  1 00:46:45: sccp_keepalive: send keepalive id 0, len 4
*Mar  1 00:46:46: sccp_process_mtp_pdu: appl - 6248FC10, mbuf - 6248FC84
*Mar  1 00:46:46: sccp_process_mtp_pdu: msg_ptr 6248FC8C, len 4, offset 12, msg_id 256
*Mar  1 00:46:47: sccp_process_mtp_pdu: appl - 6248FC10, mbuf - 6248FC84
*Mar  1 00:46:47: sccp_process_mtp_pdu: msg_ptr 6248FC8C, len 28, offset 36, msg_id 261
*Mar  1 00:46:47: xapp_open_receive_chnl: SCCP orc_msg - 6248FC8C, appl - 6248FC10
*Mar  1 00:46:47: xapp_search_for_chnl_rec: sess_id 27, conn_id 2817
*Mar  1 00:46:47: xapp_add_chnl_rec: chnl 6311430C
*Mar  1 00:46:47: xapp_open_receive_chnl: stat 0, eve 0, sid 27, cid 2817, codec 1, pkt-period
 20
*Mar  1 00:46:47: xapp_open_chnl_request: chnl_rec 6311430C
*Mar  1 00:46:47: xapp_open_chnl_request: chnl_rec 6311430C, sess_id 27, conn_id 2817,
cstate 0, nstate 1
*Mar  1 00:46:47: xapp_dequeue_and_process_dspf_events: chnl_rec 6311430C, state 1, eve_id
 1
*Mar  1 00:46:47: xapp_open_chnl_success: chnl_rec 6311430C
*Mar  1 00:46:47: xapp_open_chnl_success: chnl_rec 6311430C, sess_id 27, conn_id 2817,
cstate 1, nstate 2, lc_ipaddr 10.10.1.1, lport 16730
*Mar  1 00:46:47: sccp_process_mtp_pdu: appl - 6248FC10, mbuf - 6248FC84
*Mar  1 00:46:47: sccp_process_mtp_pdu: msg_ptr 6248FC8C, len 44, offset 52, msg_id 138
*Mar  1 00:46:47: xapp_start_media_transmission: SCCP stmt_msg - 6248FC8C, appl - 6248FC10
*Mar  1 00:46:47: xapp_search_for_chnl_rec: sess_id 27, conn_id 2817
*Mar  1 00:46:47: xapp_start_media_transmission: chnl_rec 6311430C, stat 2, sid 27, cid
2817, ripaddr 10.10.1.6, rport 18160, codec 1, pkt-period 20, pre 11, silen 16777502, mfpp
 1
*Mar  1 00:46:47: xapp_modify_chnl_request: chnl_rec 6311430C
*Mar  1 00:46:47: xapp_modify_chnl_request: chnl_rec 6311430C, sess_id 27, conn_id 2817,
cstate 2, nstate 2
*Mar  1 00:46:47: xapp_dequeue_and_process_dspf_events: chnl_rec 6311430C, state 2, eve_id
 4
*Mar  1 00:46:47: xapp_modify_chnl_success: chnl_rec 6311430C, sess_id 27, conn_id 2817,
cstate 2
*Mar  1 00:46:52: sccp_create_application: send keepalive msg, appl 6248F760, appl_type 1,
 count 0
*Mar  1 00:46:52: sccp_keepalive: send keepalive id 0, len 4
*Mar  1 00:46:52: sccp_process_mtp_pdu: appl - 6248F760, mbuf - 6248F7D4
*Mar  1 00:46:52: sccp_process_mtp_pdu: msg_ptr 6248F7DC, len 4, offset 12, msg_id 256
*Mar  1 00:46:53: sccp_create_application: send keepalive msg, appl 6248FC10, appl_type 2,
 count 0
*Mar  1 00:46:53: sccp_keepalive: send keepalive id 0, len 4
*Mar  1 00:46:54: sccp_process_mtp_pdu: appl - 6248FC10, mbuf - 6248FC84
*Mar  1 00:46:54: sccp_process_mtp_pdu: msg_ptr 6248FC8C, len 4, offset 12, msg_id 256
*Mar  1 00:46:59: sccp_create_application: send keepalive msg, appl 6248F760, appl_type 1,
 count 0
*Mar  1 00:46:59: sccp_keepalive: send keepalive id 0, len 4
*Mar  1 00:47:00: sccp_process_mtp_pdu: appl - 6248F760, mbuf - 6248F7D4
*Mar  1 00:47:00: sccp_process_mtp_pdu: msg_ptr 6248F7DC, len 4, offset 12, msg_id 256
*Mar  1 00:47:01: sccp_create_application: send keepalive msg, appl 6248FC10, appl_type 2,
 count 0
*Mar  1 00:47:01: sccp_keepalive: send keepalive id 0, len 4
*Mar  1 00:47:01: sccp_process_mtp_pdu: appl - 6248FC10, mbuf - 6248FC84
*Mar  1 00:47:01: sccp_process_mtp_pdu: msg_ptr 6248FC8C, len 4, offset 12, msg_id 256
*Mar  1 00:47:07: sccp_create_application: send keepalive msg, appl 6248F760, appl_type 1,
 count 0
*Mar  1 00:47:07: sccp_keepalive: send keepalive id 0, len 4
*Mar  1 00:47:07: sccp_process_mtp_pdu: appl - 6248F760, mbuf - 6248F7D4
*Mar  1 00:47:07: sccp_process_mtp_pdu: msg_ptr 6248F7DC, len 4, offset 12, msg_id 256
*Mar  1 00:47:08: sccp_create_application: send keepalive msg, appl 6248FC10, appl_type 2,
 count 0
*Mar  1 00:47:08: sccp_keepalive: send keepalive id 0, len 4
*Mar  1 00:47:09: sccp_process_mtp_pdu: appl - 6248FC10, mbuf - 6248FC84
*Mar  1 00:47:09: sccp_process_mtp_pdu: msg_ptr 6248FC8C, len 4, offset 12, msg_id 256
*Mar  1 00:47:14: sccp_create_application: send keepalive msg, appl 6248F760, appl_type 1,
 count 0
*Mar  1 00:47:14: sccp_keepalive: send keepalive id 0, len 4
*Mar  1 00:47:15: sccp_process_mtp_pdu: appl - 6248F760, mbuf - 6248F7D4
*Mar  1 00:47:15: sccp_process_mtp_pdu: msg_ptr 6248F7DC, len 4, offset 12, msg_id 256
*Mar  1 00:47:16: sccp_create_application: send keepalive msg, appl 6248FC10, appl_type 2,
 count 0
*Mar  1 00:47:16: sccp_keepalive: send keepalive id 0, len 4
*Mar  1 00:47:16: sccp_process_mtp_pdu: appl - 6248FC10, mbuf - 6248FC84
*Mar  1 00:47:16: sccp_process_mtp_pdu: msg_ptr 6248FC8C, len 4, offset 12, msg_id 256
*Mar  1 00:47:22: sccp_create_application: send keepalive msg, appl 6248F760, appl_type 1,
 count 0
*Mar  1 00:47:22: sccp_keepalive: send keepalive id 0, len 4
*Mar  1 00:47:22: sccp_process_mtp_pdu: appl - 6248F760, mbuf - 6248F7D4
*Mar  1 00:47:22: sccp_process_mtp_pdu: msg_ptr 6248F7DC, len 4, offset 12, msg_id 256
*Mar  1 00:47:23: sccp_create_application: send keepalive msg, appl 6248FC10, appl_type 2,
 count 0
*Mar  1 00:47:23: sccp_keepalive: send keepalive id 0, len 4
*Mar  1 00:47:24: sccp_process_mtp_pdu: appl - 6248FC10, mbuf - 6248FC84
*Mar  1 00:47:24: sccp_process_mtp_pdu: msg_ptr 6248FC8C, len 4, offset 12, msg_id 256
*Mar  1 00:47:29: sccp_create_application: send keepalive msg, appl 6248F760, appl_type 1,
 count 0
*Mar  1 00:47:29: sccp_keepalive: send keepalive id 0, len 4
*Mar  1 00:47:30: sccp_process_mtp_pdu: appl - 6248F760, mbuf - 6248F7D4
*Mar  1 00:47:30: sccp_process_mtp_pdu: msg_ptr 6248F7DC, len 4, offset 12, msg_id 256
*Mar  1 00:47:31: sccp_create_application: send keepalive msg, appl 6248FC10, appl_type 2,
 count 0
*Mar  1 00:47:31: sccp_keepalive: send keepalive id 0, len 4
*Mar  1 00:47:31: sccp_process_mtp_pdu: appl - 6248FC10, mbuf - 6248FC84
*Mar  1 00:47:31: sccp_process_mtp_pdu: msg_ptr 6248FC8C, len 4, offset 12, msg_id 256
```

Related Commands

Command	Description
debug frame-relay vc-bundle	Sets debugging levels for the DSP-farm service.
dspfarm (DSP farm)	Enables DSP-farm service.
sccp	Enables SCCP and its associated transcoding and conferencing applications.
show sccp	Displays the SCCP configuration information and current status.

debug sccp config

To enable Skinny Client Control Protocol (SCCP) event debugging, use the **debug sccp config** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug sccp config {**all**| **errors**| **events**| **parser**}

no debug sccp config {**all**| **errors**| **events**| **parser**}

Syntax Description

all	Displays all SCCP auto-config debug trace.
errors	Displays SCCP auto-config errors.
events	Displays SCCP auto-config events.
parser	Displays SCCP auto-config parser.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(8)XY	This command was introduced on the Communication Media Module.
12.3(14)T	This command was integrated into Cisco IOS Release 12.3(14)T.
12.4(3)	This command was integrated into Cisco IOS Release 12.4(3).

Examples

The following example shows the **debug sccp config** command used to enable SCCP event debugging and to display SCCP auto-configuration events:

```
Router# debug sccp config events
...
Feb  8 02:17:31.119: mp_auto_cfg_request(req_id=2, prof=995, ccm_group_id=0)
Feb  8 02:17:31.123: mp_auto_cfg_is_up: SCCP auto-config is enabled & registered
...
```

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

Table 2: debug sccp config Field Descriptions

Field	Description
prof=995	Indicates the profile ID. If generated by media processor auto-configuration, profile IDs are preceded by 99.
SCCP auto-config is enabled & registered	Indicates the registration of sccp when auto-config is complete.

Related Commands

Command	Description
auto-config	Enables auto-configuration or enters auto-config application configuration mode for the SCCP application.
debug auto-config	Enables debugging for auto-configuration applications.
show auto-config	Displays the current status of auto-configuration applications.

debug qbm

To display debugging output for quality of service (QoS) bandwidth manager (QBM) options, use the **debug qbm** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug qbm {**api**| **events**}

no debug qbm {**api**| **events**}

Syntax Description

api	Displays information about QBM client requests and notifications. See the "Usage Guidelines" section for additional information.
events	Displays information about QBM pool events.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(33)SRC	This command was introduced.
Cisco IOS XE Release 2.6	This command was integrated into Cisco IOS XE Release 2.6.

Usage Guidelines

Use the **debug qbm** command to troubleshoot QBM behavior.

Examples of client requests are when a client creates or destroys a bandwidth pool and when a client attempts to admit bandwidth into a pool. An example of a notification is when a client's previously admitted bandwidth gets preempted from a pool.

Examples

The following example shows how to enable the **debug qbm api**command:

```
Router# debug qbm api
QBM client requests and notifications debugging is on
```

The following example show how to enable the **debug qbm events**command:

```
Router# debug qbm events
QBM pool events debugging is on
```

The following example shows how to verify that QBM debugging is enabled:

```
Router# show debug
QoS Bandwidth Manager:
  QBM client requests and notifications debugging is on
  QBM pool events debugging is on
```

Related Commands

Command	Description
show qbm client	Displays registered QBM clients.
show qbm pool	Displays allocated QBM pools and associated objects.

debug sdlc

To display information on Synchronous Data Link Control (SDLC) frames received and sent by any router serial interface involved in supporting SDLC end station functions, use the **debug sdlc** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug sdlc

no debug sdlc

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

Note Because the **debug sdlc** command can generate many messages and alter timing in the network node, use it only when instructed by authorized support personnel.

Examples

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

```
Router# debug sdlc
SDLC: Sending RR at location 4
Serial3: SDLC O (12495952) C2 CONNECT (2) RR P/F 6
Serial3: SDLC I (12495964) [C2] CONNECT (2) RR P/F 0 (R) [VR: 6 VS: 0]
Serial3: SDLC T [C2] 12496064 CONNECT 12496064 0
SDLC: Sending RR at location 4
Serial3: SDLC O (12496064) C2 CONNECT (2) RR P/F 6
Serial3: SDLC I (12496076) [C2] CONNECT (2) RR P/F 0 (R) [VR: 6 VS: 0]
Serial3: SDLC T [C2] 12496176 CONNECT 12496176 0
```

The following line of output indicates that the router is sending a Receiver Ready packet at location 4 in the code:

```
SDLC: Sending RR at location 4
```

The following line of output describes a frame output event:

```
Serial1/0: SDLC O 04 CONNECT (285) IFRAME P/F 6
```

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

Table 3: debug sdlc Field Descriptions for a Frame Output Event

Field	Description
Serial1/0	Interface type and unit number reporting the frame event.
SDLC	Protocol providing the information.

Field	Description
O	Command mode of frame event. Possible values are as follows: • I--Frame input • O--Frame output • T--T1 timer expired
04	SDLC address of the SDLC connection.
CONNECT	State of the protocol when the frame event occurred. Possible values are as follows: • CONNECT • DISCONNECT • DISCSENT (disconnect sent) • ERROR (FRMR frame sent) • REJSENT (reject frame sent) • SNRMSENT (SNRM frame sent) • USBUSY • THEMBUSY • BOTHBUSY
(285)	Size of the frame (in bytes).
IFRAME	Frame type name. Possible values are as follows: • DISC--Disconnect • DM--Disconnect mode • FRMR--Frame reject • IFRAME--Information frame • REJ--Reject • RNR--Receiver not ready • RR--Receiver ready • SIM--Set Initialization mode command • SNRM--Set Normal Response Mode • TEST--Test frame • UA--Unnumbered acknowledgment • XID--EXchange ID

Field	Description
P/F	Poll/Final bit indicator. Possible values are as follows: • F--Final (printed for Response frames) • P--Poll (printed for Command frames) • P/F--Poll/Final (printed for RR, RNR, and REJ frames, which can be either Command or Response frames)
6	Receive count; range: 0 to 7.

The following line of output describes a frame input event:

```
Serial1/0: SDLC I 02 CONNECT (16) IFRAME P 7 0,[VR: 7 VS: 0]
```

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

Table 4: debug sdlc Field Descriptions for a Frame Input Event

Field	Description
02	SDLC address.
IFRAME	Traffic engineering type.
P	Poll bit P is on.
VR: 7	Receive count; range: 0 to 7.
VS: 0	Send count; range: 0 to 7.

The following line of output describes a frame timer event:

```
Serial1/0: SDLC T 02 CONNECT 0x9CB69E8 P 0
```

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

Table 5: debug sdlc Field Descriptions for a Timer Event

Field	Description
Serial1/0	Interface type and unit number reporting the frame event.
SDLC	Protocol providing the information.
T	Timer has expired.
02	SDLC address of this SDLC connection.

Field	Description
CONNECT	State of the protocol when the frame event occurred. Possible values are as follows: • BOTHBUSY • CONNECT • DISCONNECT • DISCSENT (disconnect sent) • ERROR (FRMR frame sent) • REJSENT (reject frame sent) • SNRMSENT (SNRM frame sent) • THEMBUSY • USBUSY
0x9CB69E8	Top timer.
0	Retry count; default: 0.

Related Commands

Command	Description
debug list	Filters debugging information on a per-interface or per-access list basis.

debug sdlc local-ack

To display information on the local acknowledgment feature, use the **debug sdlc local-ack** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug sdlc local-ack [*number*]

no debug sdlc local-ack [*number*]

Syntax Description

number	(Optional) Frame-type that you want to monitor. See the "Usage Guidelines" section.

Command Modes

Privileged EXEC

Usage Guidelines

You can select the frame types you want to monitor; the frame types correspond to bit flags. You can select 1, 2, 4, or 7, which is the decimal value of the bit flag settings. If you select 1, the octet is set to 00000001. If you select 2, the octet is set to 0000010. If you select 4, the octet is set to 00000100. If you want to select all frame types, select 7; the octet is 00000111. The default is 7 for all events. The table below defines these bit flags.

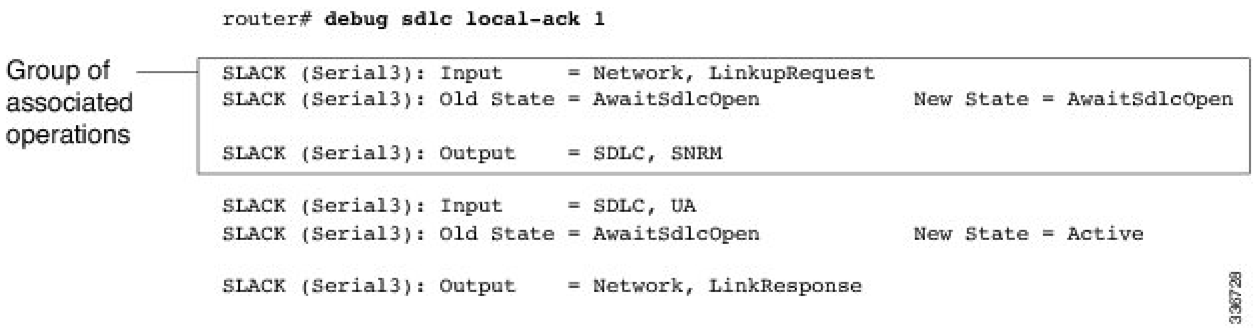
Table 6: debug sdlc local-ack Debugging Levels

Debug Command	Meaning
debug sdlc local-ack 1	Only U-Frame events
debug sdlc local-ack 2	Only I-Frame events
debug sdlc local-ack 4	Only S-Frame events
debug sdlc local-ack 7	All Synchronous Data Link Control (SDLC) Local-Ack events (default setting)

Caution Because using this command is processor intensive, it is best to use it after hours, rather than in a production environment. It is also best to use this command by itself, rather than in conjunction with other **debug**ging commands.

Examples

The following is sample output from the **debug sdlc local-ack** command:



```
router# debug sdlc local-ack 1

SLACK (Serial3): Input     = Network, LinkupRequest
SLACK (Serial3): Old State = AwaitSdlcOpen          New State = AwaitSdlcOpen

SLACK (Serial3): Output    = SDLC, SNRM

SLACK (Serial3): Input     = SDLC, UA
SLACK (Serial3): Old State = AwaitSdlcOpen          New State = Active

SLACK (Serial3): Output    = Network, LinkResponse
```

The first line shows the input to the SDLC local acknowledgment state machine:

```
SLACK (Serial3): Input     = Network, LinkupRequest
```

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

Table 7: debug sdlc local-ack Field Descriptions

Field	Description
SLACK	SDLC local acknowledgment feature is providing the information.
(Serial3):	Interface type and unit number reporting the event.
Input = Network	Source of the input.
LinkupRequest	Op code. A LinkupRequest is an example of possible values.

The second line shows the change in the SDLC local acknowledgment state machine. In this case the AwaitSdlcOpen state is an internal state that has not changed while this display was captured.

```
SLACK (Serial3): Old State = AwaitSdlcOpen           New State = AwaitSdlcOpen
```

The third line shows the output from the SDLC local acknowledgment state machine:

```
SLACK (Serial3): Output     = SDLC, SNRM
```

debug sdlc packet

To display packet information on Synchronous Data Link Control (SDLC) frames received and sent by any router serial interface involved in supporting SDLC end station functions, use the **debug sdlc packet** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug sdlc packet [*max-bytes*]

no debug sdlc packet [*max-bytes*]

Syntax Description

max-bytes	(Optional) Limits the number of bytes of data that are printed to the display.

Command Modes

Privileged EXEC

Usage Guidelines

This command requires intensive CPU processing; therefore, we recommend not using it when the router is expected to handle normal network loads, such as in a production environment. Instead, use this command when network response is noncritical. We also recommend that you use this command by itself, rather than in conjunction with other **debug** commands.

Examples

The following is sample output from the **debug sdlc packet** command with the packet display limited to 20 bytes of data:

```
Router# debug sdlc packet 20
 Serial3 SDLC Output
00000 C3842C00 02010010 019000C5 C5C5C5C5 Cd.........EEEEE
00010 C5C5C5C5                            EEEE
 Serial3 SDLC Output
00000 C3962C00 02010011 039020F2          Co.........2
 Serial3 SDLC Output
00000 C4962C00 0201000C 039020F2          Do.........2
 Serial3 SDLC Input
00000     C491                              Dj
```

debug serial interface

To display information on a serial connection failure, use the **debug serial interface** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug serial interface

no debug serial interface

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

If the **show interface serial** EXEC command shows that the line and protocol are down, you can use the **debug serial interface** command to isolate a timing problem as the cause of a connection failure. If the keepalive values in the mineseq, yourseen, and myseen fields are not incrementing in each subsequent line of output, there is a timing or line problem at one end of the connection.

Caution Although the **debug serial interface** command typically does not generate a substantial amount of output, nevertheless use it cautiously during production hours. When Switched Multimegabit Data Service (SMDS) is enabled, for example, it can generate considerable output.

The output of the **debug serial interface** command can vary, depending on the type of WAN configured for an interface: Frame Relay, High-Level Data Link Control (HDL) , High-Speed Serial Interface (HSSI), SMDS, or X.25. The output also can vary depending on the type of encapsulation configured for that interface. The hardware platform also can affect **debug serial interface** output.

Examples

The following sections show and describe sample **debug serial interface** output for various configurations.

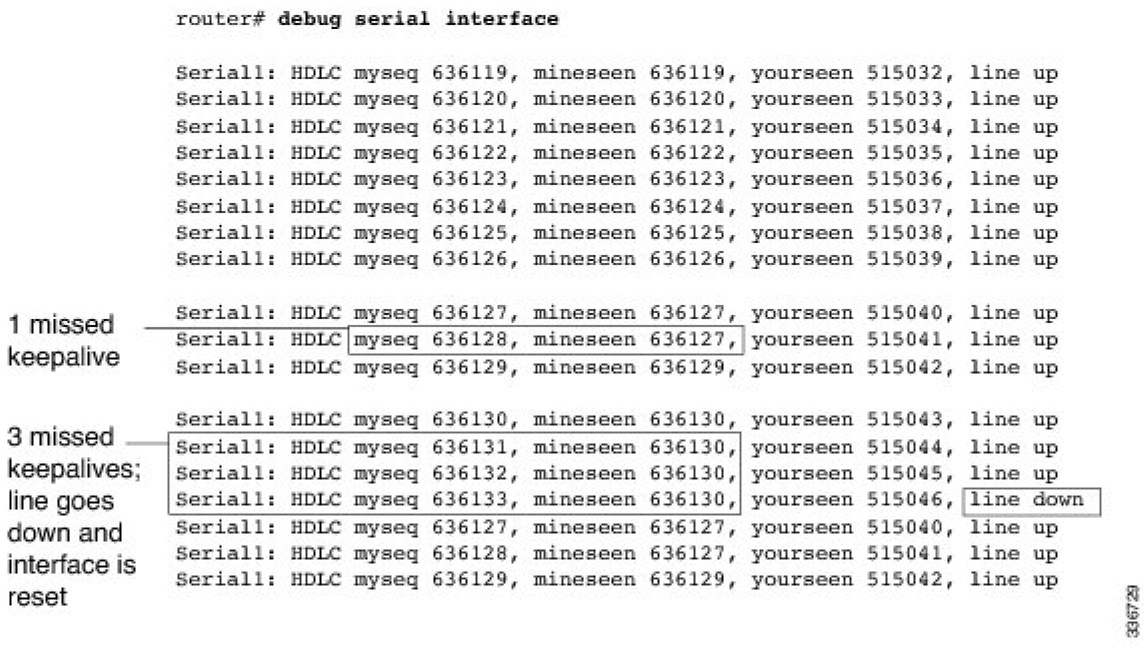
Examples

The following me ssage is displayed if the encapsulation for the interface is Frame Relay (or HDLC) and the router attempts to send a packet containing an unknown packet type:

```
Illegal serial link type code xxx
```

Examples

The following is sample output from the **debug serial interface** command for an HDLC connection when keepalives are enabled. This output shows that the remote router is not receiving all the keepalives the router is sending. When the difference in the values in the myseq and mineseen fields exceeds three, the line goes down and the interface is reset.



```
router# debug serial interface

Serial1: HDLC myseq 636119, mineseen 636119, yourseen 515032, line up
Serial1: HDLC myseq 636120, mineseen 636120, yourseen 515033, line up
Serial1: HDLC myseq 636121, mineseen 636121, yourseen 515034, line up
Serial1: HDLC myseq 636122, mineseen 636122, yourseen 515035, line up
Serial1: HDLC myseq 636123, mineseen 636123, yourseen 515036, line up
Serial1: HDLC myseq 636124, mineseen 636124, yourseen 515037, line up
Serial1: HDLC myseq 636125, mineseen 636125, yourseen 515038, line up
Serial1: HDLC myseq 636126, mineseen 636126, yourseen 515039, line up

Serial1: HDLC myseq 636127, mineseen 636127, yourseen 515040, line up
Serial1: HDLC myseq 636128, mineseen 636127, yourseen 515041, line up
Serial1: HDLC myseq 636129, mineseen 636129, yourseen 515042, line up

Serial1: HDLC myseq 636130, mineseen 636130, yourseen 515043, line up
Serial1: HDLC myseq 636131, mineseen 636130, yourseen 515044, line up
Serial1: HDLC myseq 636132, mineseen 636130, yourseen 515045, line up
Serial1: HDLC myseq 636133, mineseen 636130, yourseen 515046, line down
Serial1: HDLC myseq 636127, mineseen 636127, yourseen 515040, line up
Serial1: HDLC myseq 636128, mineseen 636127, yourseen 515041, line up
Serial1: HDLC myseq 636129, mineseen 636129, yourseen 515042, line up
```

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

Table 8: debug serial interface Field Descriptions for HDLC

Field	Description
Serial 1	Interface through which the serial connection is taking place.
HDLC	Serial connection is an HDLC connection.
myseq 636119	Myseq counter increases by one each time the router sends a keepalive packet to the remote router.
mineseen 636119	Value of the mineseen counter reflects the last myseq sequence number the remote router has acknowledged receiving from the router. The remote router stores this value in its yourseen counter and sends that value in a keepalive packet to the router.
yourseen 515032	Yourseen counter reflects the value of the myseq sequence number the router has received in a keepalive packet from the remote router.
line up	Connection between the routers is maintained. Value changes to "line down" if the values of the myseq and myseen fields in a keepalive packet differ by more than three. Value returns to "line up" when the interface is reset. If the line is in loopback mode, ("looped") appears after this field.

The table below describes additional error messages that the **debug serial interface** command can generate for HDLC.

Table 9: debug serial interface Error Messages for HDLC

Field	Description
Illegal serial link type code <*xxx*>, PC = 0x*nnnnnn*	Router attempted to send a packet containing an unknown packet type.
Illegal HDLC serial type code <*xxx*>, PC = 0x*nnnnn*	Unknown packet type is received.
Serial 0: attempting to restart	Interface is down. The hardware is then reset to correct the problem, if possible.
Serial 0: Received bridge packet sent to <*nnnnnnnnn*>	Bridge packet is received over a serial interface configured for HDLC, and bridging is not configured on that interface.

Examples

On an HSSI interface, the **debug serial interface** command can generate the following additional error message:

```
HSSI0: Reset from 0x
nnnnnnn
```

This message indicates that the HSSI hardware has been reset. The 0x*nnnnnnn* variable is the address of the routine requesting that the hardware be reset; this value is useful only to development engineers.

Examples

The table below describes error mes sages that the **debug serial interface** command can generate for ISDN Basic Rate.

Table 10: debug serial interface Error Messages for ISDN Basic Rate

Message	Description
BRI: D-chan collision	Collision on the ISDN D channel has occurred; the software will retry transmission.
Received SID Loss of Frame Alignment int.	ISDN hardware has lost frame alignment. This usually indicates a problem with the ISDN network.
Unexpected IMP int: ipr = 0x*nn*	ISDN hardware received an unexpected interrupt. The 0x*nn*variable indicates the value returned by the interrupt register.

Message	Description
BRI(d): RX Frame Length Violation. Length=*n* BRI(d): RX Nonoctet Aligned Frame BRI(d): RX Abort Sequence BRI(d): RX CRC Error BRI(d): RX Overrun Error BRI(d): RX Carrier Detect Lost	Any of these messages can be displayed when a receive error occurs on one of the ISDN channels. The (d) indicates which channel it is on. These messages can indicate a problem with the ISDN network connection.
BRI0: Reset from 0x*nnnnnnn*	BRI hardware has been reset. The 0x*nnnnnnn* variable is the address of the routine that requested that the hardware be reset; it is useful only to development engineers.
BRI(d): Bad state in SCMs scm1=*x*scm2=*x*scm3=*x* BRI(d): Bad state in SCONs scon1=*x* scon2 =*x*scon3=*x* BRI(d): Bad state ub SCR; SCR=*x*	Any of these messages can be displayed if the ISDN hardware is not in the proper state. The hardware is then reset. If the message is displayed constantly, it usually indicates a hardware problem.
BRI(d): Illegal packet encapsulation=*n*	Packet is received, but the encapsulation used for the packet is not recognized. The interface might be misconfigured.

Examples

The table below describes the additional error messa ges that the **debug serial interface** command can generate for an MK5025 device.

Table 11: debug serial interface Error Messages for an MK5025 Device

Message	Description
MK5(d): Reset from 0x*nnnnnnnn*	Hardware has been reset. The 0x*nnnnnnn* variable is the address of the routine that requested that the hardware be reset; it is useful only to development engineers.
MK5(d): Illegal packet encapsulation=*n*	Packet is received, but the encapsulation used for the packet is not recognized. Interface might be misconfigured.
MK5(d): No packet available for packet realignment	Serial driver attempted to get a buffer (memory) and was unable to do so.
MK5(d): Bad state in CSR0=(*x*)	This message is displayed if the hardware is not in the proper state. The hardware is reset. If this message is displayed constantly, it usually indicates a hardware problem.

Message	Description
MK5(d): New serial state=*n*	Hardware has interrupted the software. It displays the state that the hardware is reporting.
MK5(d): DCD is down. MK5(d): DCD is up.	If the interrupt indicates that the state of carrier has changed, one of these messages is displayed to indicate the current state of DCD.

Examples

When encapsulation is set to SMDS, the **debug serial interface** command dis plays SMDS packets that are sent and received, and any error messages resulting from SMDS packet transmission.

The error messages that the **debug serial interface** command can generate for SMDS follow.

The following message indicates that a new protocol requested SMDS to encapsulate the data for transmission. SMDS is not yet able to encapsulate the protocol.

```
SMDS: Error on Serial 0, encapsulation bad protocol =
x
```

The following message indicates that SMDS was asked to encapsulate a packet, but no corresponding destination E.164 SMDS address was found in any of the static SMDS tables or in the ARP tables:

```
SMDS send: Error in encapsulation, no hardware address, type =
x
```

The following message indicates that a protocol such as Connectionless Network Service (CLNS) or IP has been enabled on an SMDS interface, but the corresponding multicast addresses have not been configured. The *n* variable displays the link type for which encapsulation was requested.

```
SMDS: Send, Error in encapsulation, type=
n
```

The following messages can occur when a corrupted packet is received on an SMDS interface. The router expected *x*, but received *y*.

```
SMDS: Invalid packet, Reserved NOT ZERO,
x y
SMDS: Invalid packet, TAG mismatch
x y
SMDS: Invalid packet, Bad TRAILER length
x y
```

The following messages can indicate an invalid length for an SMDS packet:

```
SMDS: Invalid packet, Bad BA length
x
SMDS: Invalid packet, Bad header extension length
x
SMDS: Invalid packet, Bad header extension type
x
SMDS: Invalid packet, Bad header extension value
x
```

The following messages are displayed when the **debug serial interface** command is enabled:

```
Interface Serial 0 Sending SMDS L3 packet:
SMDS: dgsize:
x
 type:0
xn
 src:
```

```
y
 dst:
z
```

If the **debug serial interface** command is enabled, the following message can be displayed when a packet is received on an SMDS interface, but the destination SMDS address does not match any on that interface:

```
SMDS: Packet
n
, not addressed to us
```

debug serial lead-transition

To activate the leads status transition debug capability for all capable ports, use the **debug serial lead-transition**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug serial lead-transition

no debug serial lead-transition

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging is not turned on.

Command Modes

Privileged EXEC

Command History

Release	Modification
Release 12.2(15)ZJ	This command was introduced on the following platforms: Cisco 2610XM, Cisco 2611XM, Cisco 2620XM, Cisco 2621XM, Cisco 2650XM, Cisco 2651XM, Cisco 2691, Cisco 3631, Cisco 3660, Cisco 3725, and Cisco 3745 routers.
Release 12.3(2)T	This command was integrated into Cisco IOS Release 12.3(2)T.

Usage Guidelines

To control which port is to be reported and therefore reduce the risk of flooding the console screen with debug information, enter the **debug condition interface serial** *slot/port*command after using the **debug serial lead-transition** command to set the condition.

Caution To avoid having the debug message flood the console screen with debug information, use these commands only when traffic on the IP network is low, so other activity on the system is not adversely affected.

Examples

The following example shows the serial control leads reported for slot 1, port 1:

```
Router# debug serial lead-transition

Router# debug condition interface serial 1/1
*Mar  1 00:17:15.040:slot(1) Port(1):DSR/DTR is Deasserted
*Mar  1 00:17:15.040:slot(1) Port(1):CTS/RTS is Deasserted
*Mar  1 00:17:47.955:slot(1) Port(1):DCD/Local Loop is Deasserted
*Mar  1 00:17:47.955:slot(1) Port(1):DSR/DTR is Deasserted
*Mar  1 00:17:47.955:slot(1) Port(1):CTS/RTS is Deasserted
Router# no shut down serial 1/1
```

```
*Mar  1 00:16:52.298:slot(1) Port(1):DSR/DTR is Asserted
*Mar  1 00:16:52.298:slot(1) Port(1):CTS/RTS is Asserted
*Mar  1 00:16:31.648:slot(1) Port(1):DCD/Local Loop is Asserted
*Mar  1 00:16:31.648:slot(1) Port(1):DSR/DTR is Asserted
*Mar  1 00:16:31.648:slot(1) Port(1):CTS/RTS is Asserted
```

The table below describes significant fields shown in the displays.

Table 12: debug serial lead-transition Field Descriptions

Field	Description
DSR/DTR is Asserted/Deasserted	The DSR or DTE signal is activated or inactivated.
CTS/RTS is Asserted/Deasserted	The CTS or RTS signal is activated or inactivated.
DCD/Local Loop is Asserted/Deasserted	The DCD or Local Loopback signal is activated or inactivated.

Related Commands

Command	Description
debug condition interface serial	Enables conditional debugging on a serial interface.

debug serial packet

To display more detailed serial interface debugging information than you can obtain using the **debug serial interface** command, use the **debug serial packet**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug serial packet

no debug serial packet

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

The **debug serial packet** command generates output that is dependent on the type of serial interface and the encapsulation running on that interface. The hardware platform also can impact **debug serial packet** output.

The **debug serial packet** command displays output for only Switched Multimegabit Data Service (SMDS) encapsulations.

Examples

The following is sample output from the **debug serial packet** command when SM DS is enabled on the interface:

```
Router# debug serial packet
Interface Serial2 Sending SMDS L3 packet:
SMDS Header: Id: 00 RSVD: 00 BEtag: EC Basize: 0044
Dest:E18009999999FFFF Src:C12015804721FFFF Xh:04030000030001000000000000000000
SMDS LLC: AA AA 03 00 00 00 80 38
SMDS Data: E1 19 01 00 00 80 00 00 0C 00 38 1F 00 0A 00 80 00 00 0C 01 2B 71
SMDS Data: 06 01 01 0F 1E 24 00 EC 00 44 00 02 00 00 83 6C 7D 00 00 00 00 00
SMDS Trailer: RSVD: 00 BEtag: EC Length: 0044
```

As the output shows, when encapsulation is set to SMDS, the **debug serial packet** command displays the entire SMDS header (in hexadecimal notation), and some payload data on transmit or receive. This information is useful only when you have an understanding of the SMDS protocol. The first line of the output indicates either Sending or Receiving.

debug service-group

To enable debugging of service-group events and errors, use the **debug service-group** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug service-group {**all**| **error**| **feature**| **group**| **interface**| **ipc**| **member**| **qos**| **stats**}

no debug service-group {**all**| **error**| **feature**| **group**| **interface**| **ipc**| **member**| **qos**| **stats**}

Syntax Description

all	All service-group debugging.
error	Service-group errors.
feature	Service-group features.
group	Service-group events.
interface	Service-group interface events.
ipc	Service-group Inter-Process Communication (IPC) messaging.
member	Service-group member events.
qos	Service-group Quality of Service (QoS).
stats	Service-group statistics.

Command Modes

Privileged EXEC (#)

Command History

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

Examples

In the following example, service-group debugging for service-group member events has been enabled:

```
Router> enable
Router# debug service-group member
%Service Group membership debugging is on
```

debug service-module

To display debugging information that monitors the detection and clearing of network alarms on the integrated channel service unit/data service unit (CSU/DSU) modules, use the **debug service-module** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug service-module

no debug service-module

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

Use this command to enable and disable debug logging for the serial 0 and serial 1 interfaces when an integrated CSU/DSU is present. This command enables debugging on all interfaces.

Network alarm status can also be viewed through the use of the **show service-module** command.

Note The debug output varies depending on the type of service module installed in the router.

Examples

The following is sample output from the **debug service-module** command:

```
Router# debug service-module
SERVICE_MODULE(1): loss of signal ended after duration 00:05:36
SERVICE_MODULE(1): oos/oof ended after duration 01:05:14
SERVICE_MODULE(0): Unit has no clock
SERVICE_MODULE(0): detects loss of signal
SERVICE_MODULE(0): loss of signal ended after duration 00:00:33
```

debug sgbp dial-bids

To display large-scale dial-out negotiations between the primary network access server (NAS) and alternate NASs, use the **debug sgbp dial-bids** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug sgbp dial-bids

no debug sgbp dial-bids

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

Use this command only when the **sgbp dial-bids** command has been configured.

Examples

The following is sample output from the **debug sgbp dial-bids**command:

```
Router# debug sgbp dial-bids
*Jan  1 00:25:03.643: SGBP-RES:  New bid add request: 4B0 8 2 1 DAC0 1 1
This indicates a new dialout bid has started
.
*Jan  1 00:25:03.643: SGBP-RES: Sent Discover message to ID 7B09B71E  49 bytes
The bid request has been sent
.
*Jan  1 00:25:03.647: SGBP-RES: Received Message of 49 length:

*Jan  1 00:25:03.647: SGBP-RES: header 5  30  0  31
2  0  0  2D  0  0  0  0  0  0  0  3  0  0  0  1  1E  AF  3A  41  7B  9  B7  1E  8  15  B
3  2  C  6  0  0  DA  C0  D  4  0  0  E  3  1  F  3  1
*Jan  1 00:25:03.647:
*Jan  1 00:25:03.647: SGBP RES: Scan: Message type: Offer
*Jan  1 00:25:03.647: SGBP RES: Scan: Len is 45
*Jan  1 00:25:03.647: SGBP RES: Scan: Transaction ID: 3
*Jan  1 00:25:03.647: SGBP RES: Scan: Message ID: 1
*Jan  1 00:25:03.647: SGBP RES: Scan: Client ID: 1EAF3A41
*Jan  1 00:25:03.651: SGBP RES: Scan: Server ID: 7B09B71E
*Jan  1 00:25:03.651: SGBP RES: Scan: Resource type 8  length 21
*Jan  1 00:25:03.651: SGBP RES: Scan: Phy-Port Media type: ISDN
*Jan  1 00:25:03.651: SGBP RES: Scan: Phy-Port Min BW: 56000
*Jan  1 00:25:03.651: SGBP RES: Scan: Phy-Port Num Links: 0
*Jan  1 00:25:03.651: SGBP RES: Scan: Phy-Port User class: 1
*Jan  1 00:25:03.651: SGBP RES: Scan: Phy-Port Priority: 1
*Jan  1 00:25:03.651: SGBP-RES: received 45 length Offer packet
*Jan  1 00:25:03.651: SGBP-RES: Offer from 7B09B71E for Transaction 3 accepted
*Jan  1 00:25:03.651: SGBP RES: Server is uncongested. Immediate win
An alternate network access server has responded and won the bid
.
*Jan  1 00:25:03.651: SGBP-RES: Bid Succeeded  handle 7B09B71E  Server-id 4B0
*Jan  1 00:25:03.651: SGBP-RES: Sent Dial-Req message to ID 7B09B71E  66 bytes
The primary network access server has asked the alternate server to dial.
*Jan  1 00:25:04.651: SGBP-RES: QScan: Purging entry
*Jan  1 00:25:04.651: SGBP-RES: deleting entry 6112E204 1EAF3A41 from list...
```

debug sgbp error

To display debugging messages about routing problems between members of a stack group, use the **debug sgbp error**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug sgbp error

no debug sgbp error

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
11.2(9)	This command was introduced.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

Enter the **debug sgbp error**command to enable the display of debugging messages about routing problems between members of a stack group.

Note In unusual cases you may see debugging messages that are not documented on this command reference page. These debugging messages are intended for expert diagnostic interpretation by the Cisco Technical Assistance Center (TAC).

Examples

One common configuration error is setting a source IP address for a stack member that does not match the locally defined IP address for the same stack member. The following debugging output shows the error message that results from this misconfiguration:

```
Systema# debug sgbp error

%SGBP-7-DIFFERENT - systemb's addr 10.1.1.2 is different from hello's addr 10.3.4.5
```

This error means that the source IP address of the Stack Group Bidding Protocol (SGBP) hello message received from systemb does not match the IP address configured locally for systemb (through the **sgbp member** command). Correct this configuration error by going to systemb and checking for multiple interfaces by which the SGBP hello can send the message.

Another common error message is:

```
Systema# debug sgbp error

%SGBP-7-MISCONF, Possible misconfigured member routerk (10.1.1.6)
```

This error message means that routerk is not defined locally, but is defined on another stack member. Correct this configuration error by defining routerk across all members of the stack group using the **sgbp member**command.

The following error message indicates that an SGBP peer is leaving the stack group:

```
Systema# debug sgbp error
%SGBP-7-LEAVING:Member systemc leaving group stack1
```

This error message indicates that the peer systemc is leaving the stack group. Systemc could be leaving the stack group intentionally, or a connectivity problem may exist.

The following error message indicates that an SGBP event was detected from an unknown peer:

```
Systema# debug sgbp error
%SGBP-7-UNKNOWPEER:Event 0x10 from peer at 172.21.54.3
```

An SGBP event came from a network host that was not recognizable as an SGBP peer. Check to see if a network media error could have corrupted the address, or if peer equipment is malfunctioning to generate corrupted packets. Depending on the network topology and firewall of your network, SGBP packets from a nonpeer host could indicate probing and attempts to breach security.

Note If there is a chance your network is under attack, obtain knowledgeable assistance from TAC.

Related Commands

Command	Description
debug sgbp hellos	Displays debugging messages for authentication between stack group members.
sgbp group	Defines a named stack group and makes this router a member of that stack group.
sgbp member	Specifies the hostname and IP address of a router or access server that is a peer member of a stack group.
show sgbp	Displays the status of the stack group members.
username	Establishes a username-based authentication system.

debug sgbp hellos

To display debugging messages for authentication between stack members, use the **debug sgbp hellos**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug sgbp hellos

no debug sgbp hellos

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
11.2(9)	This command was introduced.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

Use the **debug sgbp hellos**command to enable the display of debugging messages for authentication between routers configured as members of a stack group.

Note In unusual cases you may see debugging messages that are not documented on this command reference page. These debugging messages are intended for expert diagnostic interpretation by the Cisco Technical Assistance Center (TAC).

Examples

The following output from the **debug sgbp hellos** command shows systema sending a successful Challenge Handshake Authentication Protocol (CHAP) challenge to and receiving a response from systemb. Similarly, systemb sends out a challenge and receives a response from systema.

```
systema# debug sgbp hellos

%SGBP-7-CHALLENGE: Send Hello Challenge to systemb group stack1
%SGBP-7-CHALLENGED: Hello Challenge message from member systemb (10.1.1.2)
%SGBP-7-RESPONSE: Send Hello Response to systemb group stack1
%SGBP-7-CHALLENGE: Send Hello Challenge to systemb group stack1
%SGBP-7-RESPONDED: Hello Response message from member systemb (10.1.1.2)
%SGBP-7-AUTHOK: Send Hello Authentication OK to member systemb (10.1.1.2)
%SGBP-7-INFO: Addr = 10.1.1.2 Reference = 0xC347DF7
%SGBP-5-ARRIVING: New peer event for member systemb
```

This debug output is self-explanatory.

If authentication fails, you may see one of the following messages in your debug output:

```
%SGBP-7-AUTHFAILED - Member systemb failed authentication
```

This error message means that the remote systemb password for the stack group does not match the password defined on systema. To correct this error, make sure that both systema and systemb have the same password defined using the **username** command.

```
%SGBP-7-NORESP -Fail to respond to systemb group stack1, may not have password.
```

This error message means that systema does not have a username or password defined. To correct this error, define a common group password across all stack members using the **username**command.

Related Commands

Command	Description
debug sgbp error	Displays debugging messages about routing problems between members of a stack group.
sgbp group	Defines a named stack group and makes this router a member of that stack group.
sgbp member	Specifies the hostname and IP address of a router or access server that is a peer member of a stack group.
show sgbp	Displays the status of the stack group members.
username	Establishes a username-based authentication system.

debug sgcp

To debug the Simple Gateway Control Protocol (SGCP), use the **debug sgcp**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug sgcp {**errors**| **events**| **packet**}

no debug sgcp {**errors**| **events**| **packet**}

Syntax Description

errors	Displays debug information about SGCP errors.
events	Displays debug information about SGCP events.
packet	Displays debug information about SGCP packets.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(5)T	This command was introduced.
12.0(7)T	Support for this command was extended to the Cisco uBR924 cable access router.

Examples

See the following examples to enable and disable debugging at the specified level:

```
Router# debug sgcp errors
Simple Gateway Control Protocol errors debugging is on
Router# no debug sgcp errors
Simple Gateway Control Protocol errors debugging is off
Router#
Router# debug sgcp events
Simple Gateway Control Protocol events debugging is on
Router# no debug sgcp events
Simple Gateway Control Protocol events debugging is off
Router#
Router# debug sgcp packet
Simple Gateway Control Protocol packets debugging is on
Router# no debug sgcp packet
Simple Gateway Control Protocol packets debugging is off
Router#
```

Related Commands

Command	Description
sgcp	Starts and allocates resources for the SCGP daemon.

debug sgcp errors

To debug Simple Gateway Control Protocol (SGCP) errors, use the **debug sgcp errors** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug sgcp errors [**endpoint** *string*]

no debug sgcp errors

Syntax Description

endpoint *string*	(Optional) Specifies the endpoint string if you want to debug SGCP errors for a specific endpoint. On the Cisco MC3810 router, the endpoint string syntax takes the following forms: • DS1 endpoint: **DS1 -** *slot*/*port* • POTS endpoint: **aaln**/*slot*/*port* On the Cisco 3600 router, the endpoint string syntax takes the following forms: • DS1 endpoint: *slot*/*subunit*/**DS1 -** *ds1 number*/*ds0 number* • POTS endpoint: **aaln**/*slot*/*subunit*/*port*

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(5)T	This command was introduced on the Cisco AS5300 access server in a private release that was not generally available.
12.0(7)XK	Support for this command was extended to the Cisco MC3810 and the Cisco 3600 series routers (except for the Cisco 3620). Also, the **endpoint** keyword was added.

Examples

The following example shows the debugging of SGCP errors being enabled:

```
Router# debug sgcp errors
```

```
Simple Gateway Control Protocol errors debugging is on
no errors since call went through successfully.
```

The following example shows a debug trace for SGCP errors on a specific endpoint:

```
Router# debug sgcp errors endpoint DS1-0/1
End point name for error debug:DS1-0/1 (1)
00:08:41:DS1 = 0, DS0 = 1
00:08:41:Call record found
00:08:41:Enable error end point debug for (DS1-0/1)
```

Related Commands

Command	Description
debug rtpspi all	Debugs all RTP SPI errors, sessions, and in/out functions.
debug rtpspi errors	Debugs RTP SPI errors.
debug rtpspi inout	Debugs RTP SPI in/out functions.
debug rtpspi send-nse	Triggers the RTP SPI to send a triple redundant NSE.
debug sgcp events	Debugs SGCP events.
debug sgcp packet	Debugs SGCP packets.
debug vtsp send-nse	Sends and debugs a triple redundant NSE from the DSP to a remote gateway.

debug sgcp events

To debug Simple Gateway Control Protocol (SGCP) events, use the **debug sgcp events** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug sgcp events [**endpoint** *string*]

no debug sgcp events

Syntax Description

endpoint *string*	(Optional) Specifies the endpoint string if you want to debug SGCP errors for a specific endpoint. On the Cisco MC3810 router, the endpoint string syntax takes the following forms: • DS1 endpoint: **DS1 -** *slot*/*port* • POTS endpoint: **aaln**/*slot*/*port* On the Cisco 3600 router, the endpoint string syntax takes the following forms: • DS1 endpoint: *slot*/*subunit*/**DS1 -** *ds1 number*/*ds0 number* • POTS endpoint: **aaln**/*slot*/*subunit*/*port*

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(5)T	This command was introduced on the Cisco AS5300 access server in a private release that was not generally available.
12.0(7)XK	Support for this command was extended to the Cisco MC3810 and the Cisco 3600 series routers (except for the Cisco 3620 router). Also, the **endpoint** keyword was added.

Examples

The following example shows a debug trace for SGCP events on a specific endpoint:

```
Router# debug sgcp events endpoint DS1-0/1
End point name for event debug:DS1-0/1 (1)
```

```
00:08:54:DS1 = 0, DS0 = 1
00:08:54:Call record found
00:08:54:Enable event end point debug for (DS1-0/1)
```

The following example shows a debug trace for all SGCP events on a gateway:

```
Router# debug sgcp events
*Mar  1 01:13:31.035:callp :19196BC, state :0, call ID :-1, event :23
*Mar  1 01:13:31.035:voice_if->call_agent_ipaddr used as Notify entityNotify entity available
 for Tx SGCP msg
NTFY send to ipaddr=1092E01 port=2427
*Mar  1 01:13:31.039:Push msg into SGCP wait ack queue* (1)[25]
*Mar  1 01:13:31.039:Timed Out interval [1]:(2000)
*Mar  1 01:13:31.039:Timed Out interval [1]:(2000)(0):E[25]
*Mar  1 01:13:31.075:Removing msg :
NTFY 25 ds1-1/13@mc1 SGCP 1.1
X:358258758
O:hd
*Mar  1 01:13:31.075:Unqueue msg from SGCP wait ack q** (0)[25]DS1 = 1, DS0 = 13
*Mar  1 01:13:31.091:callp :19196BC, vdbptr :1964EEC, state :1
*Mar  1 01:13:31.091:Checking ack (trans ID 237740140) :
*Mar  1 01:13:31.091:is_capability_ok:caps.codec=5, caps.pkt=10, caps.nt=8
*Mar  1 01:13:31.091:is_capability_ok:supported signal=0x426C079C, signal2=0x80003,
                        event=0x6003421F, event2=0x3FD
requested signal=0x0, signal2=0x0,
                        event=0x20000004, event2=0xC
*Mar  1 01:13:31.091:Same digit map is download (ds1-1/13@mc1)
*Mar  1 01:13:31.091:R:requested trans_id (237740140)
*Mar  1 01:13:31.091:process_signal_ev:seizure possible=1, signal mask=0x4, mask2=0x0
*Mar  1 01:13:32.405:SGCP Session Appl:ignore CCAPI event 10
*Mar  1 01:13:32.489:callp :19196BC, state :1, call ID :16, event :9
*Mar  1 01:13:32.610:SGCP Session Appl:ignore CCAPI event 10
*Mar  1 01:13:32.670:callp :19196BC, state :1, call ID :16, event :9
*Mar  1 01:13:32.766:SGCP Session Appl:ignore CCAPI event 10
*Mar  1 01:13:32.810:callp :19196BC, state :1, call ID :16, event :9
*Mar  1 01:13:32.931:SGCP Session Appl:ignore CCAPI event 10
*Mar  1 01:13:32.967:callp :19196BC, state :1, call ID :16, event :9
*Mar  1 01:13:33.087:SGCP Session Appl:ignore CCAPI event 10
*Mar  1 01:13:33.132:callp :19196BC, state :1, call ID :16, event :9
*Mar  1 01:13:33.240:SGCP Session Appl:ignore CCAPI event 10
*Mar  1 01:13:33.280:callp :19196BC, state :1, call ID :16, event :9
*Mar  1 01:13:33.389:SGCP Session Appl:ignore CCAPI event 10
*Mar  1 01:13:33.433:callp :19196BC, state :1, call ID :16, event :9
*Mar  1 01:13:33.537:SGCP Session Appl:ignore CCAPI event 10
*Mar  1 01:13:33.581:callp :19196BC, state :1, call ID :16, event :9
*Mar  1 01:13:33.702:SGCP Session Appl:ignore CCAPI event 10
*Mar  1 01:13:33.742:callp :19196BC, state :1, call ID :16, event :9
*Mar  1 01:13:33.742:voice_if->call_agent_ipaddr used as Notify entityNotify entity available
 for Tx SGCP msg
NTFY send to ipaddr=1092E01 port=2427
*Mar  1 01:13:33.742:Push msg into SGCP wait ack queue* (1)[26]
*Mar  1 01:13:33.742:Timed Out interval [1]:(2000)
*Mar  1 01:13:33.742:Timed Out interval [1]:(2000)(0):E[26]
*Mar  1 01:13:33.786:Removing msg :
NTFY 26 ds1-1/13@mc1 SGCP 1.1
X:440842371
O:k0, 4081037, s0
*Mar  1 01:13:33.786:Unqueue msg from SGCP wait ack q** (0)[26]DS1 = 1, DS0 = 13
*Mar  1 01:13:33.802:callp :19196BC, vdbptr :1964EEC, state :1
*Mar  1 01:13:33.802:Checking ack (trans ID 698549528) :
*Mar  1 01:13:33.802:is_capability_ok:caps.codec=5, caps.pkt=10, caps.nt=8
*Mar  1 01:13:33.802:is_capability_ok:supported signal=0x426C079C, signal2=0x80003,
                        event=0x6003421F, event2=0x3FD
requested signal=0x0, signal2=0x0,
                        event=0x4, event2=0x0
*Mar  1 01:13:33.802:R:requested trans_id (698549528)
*Mar  1 01:13:33.802:set_up_voip_call_leg:peer_addr=0, peer_port=0.
*Mar  1 01:13:33.806:call_setting_crcx:Enter CallProceeding state rc = 0, call_id=16
*Mar  1 01:13:33.806:callp :19196BC, state :4, call ID :16, event :31
*Mar  1 01:13:33.810:callp :1AF5798, state :2, call ID :17, event :8
call_pre_bridge!
*Mar  1 01:13:33.810:send_oc_create_ack:seizure_possiblle=1, ack-lready-sent=0, ack_send=0
*Mar  1 01:13:33.814:callp :1AF5798, state :4, call ID :17, event :28
```

```
*Mar  1 01:13:33.814:Call Connect:Raw Msg ptr=0x1995360, no-offhook=0; call-id=17
*Mar  1 01:13:33.814:SGCP Session Appl:ignore CCAPI event 37
*Mar  1 01:13:33.947:callp :19196BC, state :5, call ID :16, event :32
process_nse_on_orig
DS1 = 1, DS0 = 13
*Mar  1 01:13:34.007:callp :19196BC, vdbptr :1964EEC, state :5
*Mar  1 01:13:34.007:Checking ack (trans ID 123764791) :
*Mar  1 01:13:34.007:is_capability_ok:caps.codec=5, caps.pkt=10, caps.nt=8
*Mar  1 01:13:34.007:is_capability_ok:supported signal=0x426C079C, signal2=0x80003,
                        event=0x6003421F, event2=0x3FD
requested signal=0x0, signal2=0x0,
                        event=0x4, event2=0x0
*Mar  1 01:13:34.007:R:requested trans_id (123764791)
*Mar  1 01:13:34.007:process_signal_ev:seizure possible=1, signal mask=0x0, mask2=0x0
*Mar  1 01:13:34.007:modify_connection:echo_cancel=1.
*Mar  1 01:13:34.007:modify_connection:vad=0.
*Mar  1 01:13:34.007:modify_connection:peer_addr=6000001, peer_port=0->16500.
*Mar  1 01:13:34.007:modify_connection:conn_mode=2.
*Mar  1 01:13:34.011:callp :19196BC, state :5, call ID :16, event :31
*Mar  1 01:13:34.011:callp :1AF5798, state :5, call ID :17, event :31
process_nse_event
*Mar  1 01:13:34.051:callp :19196BC, state :5, call ID :16, event :39
*Mar  1 01:13:34.051:call_id=16, ignore_ccapi_ev:ignore 19 for state 5
DS1 = 1, DS0 = 13
*Mar  1 01:13:39.497:callp :19196BC, vdbptr :1964EEC, state :5
*Mar  1 01:13:39.497:Checking ack (trans ID 553892443) :
*Mar  1 01:13:39.497:is_capability_ok:caps.codec=5, caps.pkt=10, caps.nt=8
*Mar  1 01:13:39.497:is_capability_ok:supported signal=0x426C079C, signal2=0x80003,
                        event=0x6003421F, event2=0x3FD
requested signal=0x8, signal2=0x0,
                        event=0x4, event2=0x0
*Mar  1 01:13:39.497:R:requested trans_id (553892443)
*Mar  1 01:13:39.497:process_signal_ev:seizure possible=1, signal mask=0x0, mask2=0x0
*Mar  1 01:13:39.497:modify_connection:echo_cancel=1.
*Mar  1 01:13:39.497:modify_connection:vad=0.
*Mar  1 01:13:39.497:modify_connection:peer_addr=6000001, peer_port=16500->16500.
*Mar  1 01:13:39.497:modify_connection:conn_mode=3.
*Mar  1 01:13:39.497:callp :19196BC, state :5, call ID :16, event :31
*Mar  1 01:13:39.501:callp :1AF5798, state :5, call ID :17, event :31
*Mar  1 01:14:01.168:Removing ack (trans ID 237740140) :
 200 237740140 OK
*Mar  1 01:14:03.883:Removing ack (trans ID 698549528) :
 200 698549528 OK
I:7
v=0
c=IN IP4 5.0.0.1
m=audio 16400 RTP/AVP 0
*Mar  1 01:14:04.087:Removing ack (trans ID 123764791) :
 200 123764791 OK
I:7
v=0
c=IN IP4 5.0.0.1
m=audio 16400 RTP/AVP 0
*Mar  1 01:14:09.573:Removing ack (trans ID 553892443) :
 200 553892443 OK
I:7
v=0
c=IN IP4 5.0.0.1
m=audio 16400 RTP/AVP 0
*Mar  1 01:14:48.091:callp :19196BC, state :5, call ID :16, event :12
*Mar  1 01:14:48.091:voice_if->call_agent_ipaddr used as Notify entityNotify entity available
 for Tx SGCP msg
NTFY send to ipaddr=1092E01 port=2427
*Mar  1 01:14:48.091:Push msg into SGCP wait ack queue* (1)[27]
*Mar  1 01:14:48.091:Timed Out interval [1]:(2000)
*Mar  1 01:14:48.091:Timed Out interval [1]:(2000)(0):E[27]
*Mar  1 01:14:48.128:Removing msg :
NTFY 27 ds1-1/13@mc1 SGCP 1.1
X:97849341
O:hu
*Mar  1 01:14:48.128:Unqueue msg from SGCP wait ack q** (0)[27]DS1 = 1, DS0 = 13
*Mar  1 01:14:48.212:callp :19196BC, vdbptr :1964EEC, state :5
*Mar  1 01:14:48.212:Checking ack (trans ID 79307869) :
```

```
*Mar  1 01:14:48.212:is_capability_ok:caps.codec=5, caps.pkt=10, caps.nt=8
*Mar  1 01:14:48.212:is_capability_ok:supported signal=0x426C079C, signal2=0x80003,
                        event=0x6003421F, event2=0x3FD
requested signal=0x4, signal2=0x0,
                        event=0x0, event2=0x0
*Mar  1 01:14:48.212:delete_call:callp:19196BC, call ID:16
*Mar  1 01:14:48.212:sgcp delete_call:Setting disconnect_by_dlcx to 1
*Mar  1 01:14:48.216:callp :1AF5798, state :6, call ID :17, event :29
*Mar  1 01:14:48.216:Call disconnect:Raw Msg ptr = 0x0, call-id=17
*Mar  1 01:14:48.216:disconnect_call_leg O.K. call_id=17
*Mar  1 01:14:48.216:SGCP:Call disconnect:No need to send onhook
*Mar  1 01:14:48.216:Call disconnect:Raw Msg ptr = 0x19953B0, call-id=16
*Mar  1 01:14:48.216:disconnect_call_leg O.K. call_id=16
*Mar  1 01:14:48.220:callp :1AF5798, state :7, call ID :17, event :13
*Mar  1 01:14:48.220:Processing DLCX signal request :4, 0, 0
*Mar  1 01:14:48.220:call_disconnected:call_id=17, peer 16 is not idle yet.DS1 = 1, DS0 =
13
*Mar  1 01:14:48.272:callp :19196BC, vdbptr :1964EEC, state :7
*Mar  1 01:14:48.272:Checking ack (trans ID 75540355) :
*Mar  1 01:14:48.272:is_capability_ok:caps.codec=5, caps.pkt=10, caps.nt=8
*Mar  1 01:14:48.272:is_capability_ok:supported signal=0x426C079C, signal2=0x80003,
                        event=0x6003421F, event2=0x3FD
requested signal=0x0, signal2=0x0,
                        event=0x8, event2=0x0
*Mar  1 01:14:48.272:R:requested trans_id (75540355)
*Mar  1 01:14:48.272:process_signal_ev:seizure possible=1, signal mask=0x4, mask2=0x0
*Mar  1 01:14:49.043:callp :19196BC, state :7, call ID :16, event :27
*Mar  1 01:14:49.043:process_call_feature:Onhook event
*Mar  1 01:14:49.043:callp :19196BC, state :7, call ID :16, event :13
*Mar  1 01:15:18.288:Removing ack (trans ID 79307869) :
 250 79307869 OK
*Mar  1 01:15:18.344:Removing ack (trans ID 75540355) :
 200 75540355 OK
```

Related Commands

Command	Description
debug rtpspi all	Debugs all RTP SPI errors, sessions, and in/out functions.
debug rtpspi errors	Debugs RTP SPI errors.
debug rtpspi inout	Debugs RTP SPI in/out functions.
debug rtpspi send-nse	Triggers the RTP SPI to send a triple redundant NSE.
debug sgcp errors	Debugs SGCP errors.
debug sgcp packet	Debugs SGCP packets.
debug vtsp send-nse	Sends and debugs a triple redundant NSE from the DSP to a remote gateway.

debug sgcp packet

To debug the Simple Gateway Control Protocol (SGCP), use the **debug sgcp packet** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug sgcp packet [**endpoint** *string*]

no debug sgcp packet

Syntax Description

endpoint *string*	(Optional) Specifies the endpoint string if you want to debug SGCP errors for a specific endpoint. On the Cisco MC3810, the endpoint string syntax takes the following forms: • DS1 endpoint: **DS1 -***slot* /*port* • POTS endpoint: **aaln**/*slot* /*port* On the Cisco 3600, the endpoint string syntax takes the following forms: • DS1 endpoint: *slot* /*subunit* /**DS1 -***ds1number* /*ds0number* • POTS endpoint: **aaln**/*slot* /*subunit* /*port*

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(5)T	This command was introduced on the Cisco AS5300 in a private release that was not generally available.
12.0(7)XK	Support for this command was extended to the Cisco MC3810 and the Cisco 3600 series routers (except for the Cisco 3620). Also, the **endpoint** keyword was added.

Examples

The following example shows a debug trace for SGCP packets on a specific endpoint:

```
Router# debug sgcp packet endpoint DS1-0/1 End point name for packet debug:DS1-0/1 (1)
```

```
00:08:14:DS1 = 0, DS0 = 1
00:08:14:Enable packet end point debug for (DS1-0/1)
```

The following example shows a debug trace for all SGCP packets on a gateway:

```
Router# debug sgcp packet
*Mar  1 01:07:45.204:SUCCESS:Request ID string building is OK
*Mar  1 01:07:45.204:SUCCESS:Building SGCP Parameter lines is OK
*Mar  1 01:07:45.204:SUCCESS:SGCP message building OK
*Mar  1 01:07:45.204:SUCCESS:END of building
*Mar  1 01:07:45.204:SGCP Packet sent --->
NTFY 22 ds1-1/13@mc1 SGCP 1.1
X:550092018
O:hd
<---
*Mar  1 01:07:45.204:NTFY Packet sent successfully.
*Mar  1 01:07:45.240:Packet received -
200 22
*Mar  1 01:07:45.244:SUCCESS:SGCP Header parsing was OK
*Mar  1 01:07:45.244:SUCCESS:END of Parsing
*Mar  1 01:07:45.256:Packet received -
RQNT 180932866 ds1-1/13@mc1 SGCP 1.1
X:362716780
R:hu,k0(A),s0(N),[0-9T](A) (D)
D:(9xx|xxxxxxx)
*Mar  1 01:07:45.256:SUCCESS:SGCP Header parsing was OK
*Mar  1 01:07:45.256:SUCCESS:Request ID string(362716780) parsing is OK
*Mar  1 01:07:45.260:SUCCESS:Requested Event parsing is OK
*Mar  1 01:07:45.260:SUCCESS:Digit Map parsing is OK
*Mar  1 01:07:45.260:SUCCESS:END of Parsing
*Mar  1 01:07:45.260:SUCCESS:SGCP message building OK
*Mar  1 01:07:45.260:SUCCESS:END of building
*Mar  1 01:07:45.260:SGCP Packet sent --->
200 180932866 OK
<---
*Mar  1 01:07:47.915:SUCCESS:Request ID string building is OK
*Mar  1 01:07:47.915:SUCCESS:Building SGCP Parameter lines is OK
*Mar  1 01:07:47.919:SUCCESS:SGCP message building OK
*Mar  1 01:07:47.919:SUCCESS:END of building
*Mar  1 01:07:47.919:SGCP Packet sent --->
NTFY 23 ds1-1/13@mc1 SGCP 1.1
X:362716780
O:k0, 4081037, s0
<---
*Mar  1 01:07:47.919:NTFY Packet sent successfully.
*Mar  1 01:07:47.955:Packet received -
200 23
*Mar  1 01:07:47.955:SUCCESS:SGCP Header parsing was OK
*Mar  1 01:07:47.955:SUCCESS:END of Parsing
*Mar  1 01:07:47.971:Packet received -
CRCX 938694984 ds1-1/13@mc1 SGCP 1.1
M:recvonly
L:p:10,e:on,s:off, a:G.711u
R:hu
C:6
*Mar  1 01:07:47.971:SUCCESS:SGCP Header parsing was OK
*Mar  1 01:07:47.971:SUCCESS:Connection Mode parsing is OK
*Mar  1 01:07:47.971:SUCCESS:Packet period parsing is OK
*Mar  1 01:07:47.971:SUCCESS:Echo Cancellation parsing is OK
*Mar  1 01:07:47.971:SUCCESS:Silence Supression parsing is OK
*Mar  1 01:07:47.971:SUCCESS:CODEC strings parsing is OK
*Mar  1 01:07:47.971:SUCCESS:Local Connection option parsing is OK
*Mar  1 01:07:47.971:SUCCESS:Requested Event parsing is OK
*Mar  1 01:07:47.975:SUCCESS:Call ID string(6) parsing is OK
*Mar  1 01:07:47.975:SUCCESS:END of Parsing
*Mar  1 01:07:47.979:SUCCESS:Conn ID string building is OK
*Mar  1 01:07:47.979:SUCCESS:Building SGCP Parameter lines is OK
*Mar  1 01:07:47.979:SUCCESS:SGCP message building OK
*Mar  1 01:07:47.979:SUCCESS:END of building
*Mar  1 01:07:47.979:SGCP Packet sent --->
200 938694984 OK
I:6
v=0
```

```
c=IN IP4 5.0.0.1
m=audio 16538 RTP/AVP 0
<---
*Mar  1 01:07:48.188:Packet received -
MDCX 779665338 ds1-1/13@mc1 SGCP 1.1
I:6
M:recvonly
L:p:10,e:on,s:off,a:G.711u
R:hu
C:6
v=0
c=IN IP4 6.0.0.1
m=audio 16392 RTP/AVP 0
*Mar  1 01:07:48.188:SUCCESS:SGCP Header parsing was OK
*Mar  1 01:07:48.188:SUCCESS:Conn ID string(6) parsing is OK
*Mar  1 01:07:48.192:SUCCESS:Connection Mode parsing is OK
*Mar  1 01:07:48.192:SUCCESS:Packet period parsing is OK
*Mar  1 01:07:48.192:SUCCESS:Echo Cancellation parsing is OK
*Mar  1 01:07:48.192:SUCCESS:Silence Supression parsing is OK
*Mar  1 01:07:48.192:SUCCESS:CODEC strings parsing is OK
*Mar  1 01:07:48.192:SUCCESS:Local Connection option parsing is OK
*Mar  1 01:07:48.192:SUCCESS:Requested Event parsing is OK
*Mar  1 01:07:48.192:SUCCESS:Call ID string(6) parsing is OK
*Mar  1 01:07:48.192:SUCCESS:SDP Protocol version parsing OK
*Mar  1 01:07:48.192:SUCCESS:SDP Conn Data OK
*Mar  1 01:07:48.192:SUCCESS:END of Parsing
*Mar  1 01:07:48.200:SUCCESS:Conn ID string building is OK
*Mar  1 01:07:48.200:SUCCESS:Building SGCP Parameter lines is OK
*Mar  1 01:07:48.200:SUCCESS:SGCP message building OK
*Mar  1 01:07:48.200:SUCCESS:END of building
*Mar  1 01:07:48.200:SGCP Packet sent --->
200 779665338 OK
I:6
v=0
c=IN IP4 5.0.0.1
m=audio 16538 RTP/AVP 0
<---
*Mar  1 01:07:53.674:Packet received -
MDCX 177780432 ds1-1/13@mc1 SGCP 1.1
I:6
M:sendrecv
X:519556004
L:p:10,e:on, s:off,a:G.711u
C:6
R:hu
S:hd
v=0
c=IN IP4 6.0.0.1
m=audio 16392 RTP/AVP 0
*Mar  1 01:07:53.674:SUCCESS:SGCP Header parsing was OK
*Mar  1 01:07:53.674:SUCCESS:Conn ID string(6) parsing is OK
*Mar  1 01:07:53.674:SUCCESS:Connection Mode parsing is OK
*Mar  1 01:07:53.674:SUCCESS:Request ID string(519556004) parsing is OK
*Mar  1 01:07:53.678:SUCCESS:Packet period parsing is OK
*Mar  1 01:07:53.678:SUCCESS:Echo Cancellation parsing is OK
*Mar  1 01:07:53.678:SUCCESS:Silence Supression parsing is OK
*Mar  1 01:07:53.678:SUCCESS:CODEC strings parsing is OK
*Mar  1 01:07:53.678:SUCCESS:Local Connection option parsing is OK
*Mar  1 01:07:53.678:SUCCESS:Call ID string(6) parsing is OK
*Mar  1 01:07:53.678:SUCCESS:Requested Event parsing is OK
*Mar  1 01:07:53.678:SUCCESS:Signal Requests parsing is OK
*Mar  1 01:07:53.678:SUCCESS:SDP Protocol version parsing OK
*Mar  1 01:07:53.678:SUCCESS:SDP Conn Data OK
*Mar  1 01:07:53.678:SUCCESS:END of Parsing
*Mar  1 01:07:53.682:SUCCESS:Conn ID string building is OK
*Mar  1 01:07:53.682:SUCCESS:Building SGCP Parameter lines is OK
*Mar  1 01:07:53.682:SUCCESS:SGCP message building OK
*Mar  1 01:07:53.682:SUCCESS:END of building
*Mar  1 01:07:53.682:SGCP Packet sent --->
200 177780432 OK
I:6
v=0
c=IN IP4 5.0.0.1
```

```
m=audio 16538 RTP/AVP 0
<---
*Mar  1 01:09:02.401:SUCCESS:Request ID string building is OK
*Mar  1 01:09:02.401:SUCCESS:Building SGCP Parameter lines is OK
*Mar  1 01:09:02.401:SUCCESS:SGCP message building OK
*Mar  1 01:09:02.401:SUCCESS:END of building
*Mar  1 01:09:02.401:SGCP Packet sent --->
NTFY 24 ds1-1/13@mc1 SGCP 1.1
X:519556004
O:hu
<---
*Mar  1 01:09:02.401:NTFY Packet sent successfully.
*Mar  1 01:09:02.437:Packet received -
200 24
*Mar  1 01:09:02.441:SUCCESS:SGCP Header parsing was OK
*Mar  1 01:09:02.441:SUCCESS:END of Parsing
*Mar  1 01:09:02.541:Packet received -
DLCX 865375036 ds1-1/13@mc1 SGCP 1.1
C:6
S:hu
*Mar  1 01:09:02.541:SUCCESS:SGCP Header parsing was OK
*Mar  1 01:09:02.541:SUCCESS:Call ID string(6) parsing is OK
*Mar  1 01:09:02.541:SUCCESS:Signal Requests parsing is OK
*Mar  1 01:09:02.541:SUCCESS:END of Parsing
*Mar  1 01:09:02.545:SUCCESS:SGCP message building OK
*Mar  1 01:09:02.545:SUCCESS:END of building
*Mar  1 01:09:02.545:SGCP Packet sent --->
250 865375036 OK
<---
*Mar  1 01:09:02.577:Packet received -
RQNT 254959796 ds1-1/13@mc1 SGCP 1.1
X:358258758
R:hd
*Mar  1 01:09:02.577:SUCCESS:SGCP Header parsing was OK
*Mar  1 01:09:02.577:SUCCESS:Request ID string(358258758) parsing is OK
*Mar  1 01:09:02.577:SUCCESS:Requested Event parsing is OK
*Mar  1 01:09:02.581:SUCCESS:END of Parsing
*Mar  1 01:09:02.581:SUCCESS:SGCP message building OK
*Mar  1 01:09:02.581:SUCCESS:END of building
*Mar  1 01:09:02.581:SGCP Packet sent --->
200 254959796 OK
```

Related Commands

Command	Description
debug rtpspi all	Debugs all RTP SPI errors, sessions, and in/out functions.
debug rtpspi errors	Debugs RTP SPI errors.
debug rtpspi inout	Debugs RTP SPI in/out functions.
debug rtpspi send-nse	Triggers the RTP SPI to send a triple redundant NSE.
debug sgcp errors	Debugs SGCP errors.
debug sgcp events	Debugs SGCP events.
debug vtsp send-nse	Sends and debugs a triple redundant NSE from the DSP to a remote gateway.

debug shared-line

To display debugging information about SIP shared lines, use the **debug shared-line**command in privileged EXEC mode. To disable debugging messages, use the **no** form of this command.

debug shared-line {**all**| **errors**| **events**| **info**}

no debug shared-line {**all**| **errors**| **events**| **info**}

Syntax Description

all	Displays all shared-line debugging messages.
errors	Displays shared-line error messages.
events	Displays shared-line event messages.
info	Displays general information about shared lines.

Command Modes

Privileged EXEC (#)

Command History

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

Examples

The following example shows output from the **debug shared-line all** command:

```
Router# debug shared-line all

Aug 21 21:56:56.949: //Shared-Line/EVENT/shrl_validate_newcall_outgoing:Outgoing call
validation request from AFW for  user = 20143, usrContainer = 4A7CFBDC
.Aug 21 21:56:56.949: //Shared-Line/INFO/shrl_find_ccb_by_dn:Searching Shared-Line table
for dn '20143'
.Aug 21 21:56:56.949: //Shared-Line/INFO/shrl_find_ccb_by_dn:Entry not found for dn '20143'
.Aug 21 21:56:56.949: //Shared-Line/INFO/shrl_find_ccb_by_demote_dn:Demoted dn: 20143
.Aug 21 21:56:56.949: //Shared-Line/INFO/shrl_validate_newcall_outgoing:User '20143' doesn't
 exist in Shared-Line table
.Aug 21 21:56:56.957: //Shared-Line/EVENT/shrl_validate_newcall_incoming:Incominging call
validation request from AFW for user = 20141
.Aug 21 21:56:56.957: //Shared-Line/INFO/shrl_find_ccb_by_dn:Searching Shared-Line table
for dn '20141'
.Aug 21 21:56:56.957: //Shared-Line/INFO/shrl_find_ccb_by_dn:Entry found [ccb = 4742EAD4]
for dn '20141'
.Aug 21 21:56:56.957: //Shared-Line/INFO/shrl_validate_newcall_incoming:User '20141' found:
 ccb = 4742EAD4, mem_count = 2
.Aug 21 21:56:56.957: //Shared-Line/EVENT/shrl_validate_newcall_incoming:Obtained call
instance inst: 0 for incoming call, incoming leg (peer_callid): 5399)
.Aug 21 21:56:56.957: //Shared-Line/INFO/shrl_update_barge_calltype:Updating shared-line
call -1 with calltype = 1
```

```
.Aug 21 21:56:56.961: //Shared-Line/INFO/shrl_find_ccb_by_dn:Searching Shared-Line table
for dn '20141'
.Aug 21 21:56:56.961: //Shared-Line/INFO/shrl_find_ccb_by_dn:Entry found [ccb = 4742EAD4]
for dn '20141'
.Aug 21 21:56:56.961: //Shared-Line/INFO/shrl_find_ccb_by_dn:Searching Shared-Line table
for dn '20141'
.Aug 21 21:56:56.961: //Shared-Line/INFO/shrl_find_ccb_by_dn:Entry found [ccb = 4742EAD4]
for dn '20141'
.Aug 21 21:57:01.689: %IPPHONE-6-REG_ALARM: 24: Name=SEP00141C48E126 Load=8.0(5.0)
Last=Phone-Reg-Rej
.Aug 21 21:57:04.261: //Shared-Line/EVENT/shrl_app_event_notify_handler:Event notification
 received: event = 9, callID = 5401, dn = 20141
.Aug 21 21:57:04.261: //Shared-Line/INFO/shrl_find_ccb_by_dn:Searching Shared-Line table
for dn '20141'
.Aug 21 21:57:04.261: //Shared-Line/INFO/shrl_find_ccb_by_dn:Entry found [ccb = 4742EAD4]
for dn '20141'
.Aug 21 21:57:04.261: //Shared-Line/EVENT/shrl_process_connect:called with state = 3, callID
 = 5401, peer callID = 5399, dn = 20141, usrContainer = 4A7CACA4
.Aug 21 21:57:04.261: //Shared-Line/INFO/shrl_connect_upd_callinfo:Parsed To: 20141@15.6.0.2,
 to-tag: 2ed5b927-6ad6
.Aug 21 21:57:04.261: //Shared-Line/INFO/shrl_connect_upd_callinfo:Parsed Contact:
20141@15.6.0.2 for sipCallId: E8583537-6F0211DD-96A69BA1-1228BEFB@15.10.0.1
.Aug 21 21:57:04.261: //Shared-Line/EVENT/shrl_connect_upd_callinfo:Obtained call instance
 inst: 0
.Aug 21 21:57:04.261: //Shared-Line/INFO/shrl_connect_upd_callinfo:CONNECT from shared line
 for incoming shared-line call.
.Aug 21 21:57:04.261: //Shared-Line/INFO/shrl_find_peer_by_ipaddr:Trying to match peer for
 member 20141@15.6.0.2
.Aug 21 21:57:04.261: //Shared-Line/INFO/shrl_find_peer_by_ipaddr:Matching peer [40002]
session target parsed = 15.6.0.2
.Aug 21 21:57:04.261: //Shared-Line/INFO/shrl_connect_upd_callinfo:Matching member found:
20141@15.6.0.2
.Aug 21 21:57:04.261: //Shared-Line/INFO/shrl_update_remote_name:Updating shared-line call
 dialog info 5401

.Aug 21 21:57:04.261: //Shared-Line/INFO/shrl_process_connect:Updated callinfo for callid:
 5401, member: '20141@15.6.0.2', peer-tag: 40002
.Aug 21 21:57:04.261: //Shared-Line/INFO/shrl_process_connect:Notify remote users about
CALL-CONNECT.
.Aug 21 21:57:04.261: //Shared-Line/EVENT/shrl_send_dialog_notify:Sending NOTIFY to remote
 user:  20141@15.6.0.1
.Aug 21 21:57:04.261: //Shared-Line/INFO/shrl_send_dialog_notify:Sending NOTIFY to remote
user:  20141@15.6.0.1 about state 3 on incoming call from 20141@15.6.0.2 privacy OFF
.Aug 21 21:57:04.261: //Shared-Line/INFO/shrl_send_dialog_notify:Dialog msg: dir: 1, orient:
 2, local_tag: 2ed5b927-6ad6, remote_tag: 89DCF0-139B, local_uri: 20141@15.6.0.2, remote_uri:
 20143@15.10.0.1
.Aug 21 21:57:04.261: //Shared-Line/INFO/shrl_send_dialog_notify:Dialog notify sent
successfully
.Aug 21 21:57:04.261: //Shared-Line/INFO/shrl_process_connect:Shared-Line '20141':
Successfully sent notify for callid: 5401
.Aug 21 21:57:04.265: //Shared-Line/INFO/shrl_find_ccb_by_dn:Searching Shared-Line table
for dn '20141'
.Aug 21 21:57:04.265: //Shared-Line/INFO/shrl_find_ccb_by_dn:Entry found [ccb = 4742EAD4]
for dn '20141'
.Aug 21 21:57:04.265: //Shared-Line/INFO/shrl_find_ccb_by_dn:Searching Shared-Line table
for dn '20143'
.Aug 21 21:57:04.265: //Shared-Line/INFO/shrl_find_ccb_by_dn:Entry not found for dn '20143'
.Aug 21 21:57:04.269: //Shared-Line/INFO/shrl_find_ccb_by_demote_dn:Demoted dn: 20143
.Aug 21 21:57:04.269: //Shared-Line/INFO/shrl_update_totag:Shared-Line not enabled for
'20143'
.Aug 21 21:57:04.269: //Shared-Line/EVENT/shrl_app_event_notify_handler:Event notification
 received: event = 21, callID = 5401, dn = 20141
.Aug 21 21:57:04.269: //Shared-Line/INFO/shrl_find_ccb_by_dn:Searching Shared-Line table
for dn '20141'
.Aug 21 21:57:04.269: //Shared-Line/INFO/shrl_find_ccb_by_dn:Entry found [ccb = 4742EAD4]
for dn '20141'
.Aug 21 21:57:04.269: //Shared-Line/EVENT/shrl_process_callerid_update:called with state =
 7, callID = 5401, peer callID = 5399, dn = 20141
.Aug 21 21:57:04.269: //Shared-Line/INFO/shrl_process_callerid_update:Updated callinfo for
 callid: 5401, member: '20141@15.6.0.2', peer-tag: 40002
.Aug 21 21:57:04.269: //Shared-Line/EVENT/shrl_is_outbound:Check for shared line call type
 callid 5401for user = 20141
.Aug 21 21:57:04.269: //Shared-Line/INFO/shrl_find_ccb_by_dn:Searching Shared-Line table
```

```
for dn '20141'
.Aug 21 21:57:04.269: //Shared-Line/INFO/shrl_find_ccb_by_dn:Entry found [ccb = 4742EAD4]
for dn '20141'
.Aug 21 21:57:04.269: //Shared-Line/EVENT/shrl_barge_type:Check for shared line call type
callid 5401for user = 20141
.Aug 21 21:57:04.269: //Shared-Line/INFO/shrl_find_ccb_by_dn:Searching Shared-Line table
for dn '20141'
.Aug 21 21:57:04.269: //Shared-Line/INFO/shrl_find_ccb_by_dn:Entry found [ccb = 4742EAD4]
for dn '20141'
.Aug 21 21:57:04.273: //Shared-Line/INFO/shrl_find_ccb_by_dn:Searching Shared-Line table
for dn '20141'
.Aug 21 21:57:04.273: //Shared-Line/INFO/shrl_find_ccb_by_dn:Entry found [ccb = 4742EAD4]
for dn '20141'
.Aug 21 21:57:04.281: //Shared-Line/EVENT/shrl_notify_done_handler:NOTIFY_DONE received for
 subID: 5 respCode: 17
.Aug 21 21:57:04.281: //Shared-Line/INFO/shrl_find_ccb_by_subid:Search ccb for subid: 5
.Aug 21 21:57:04.281: //Shared-Line/INFO/shrl_find_ccb_by_subid:Found the entry ccb: 4742EAD4
 member: 20141@15.6.0.1
.Aug 21 21:57:04.281: //Shared-Line/INFO/shrl_free_spi_respinfo:Free ASNL resp info for
subID = 5
```

Related Commands

Command	Description
shared-line	Creates a directory number to be shared by multiple SIP phones.
show shared-line	Displays information about active calls using SIP shared lines.

debug smrp all

To display information about Simple Multicast Routing Protocol (SMRP) activity, use the **debug smrp all**privileged EXEC command. The **no** form of this command disables debugging output.

debug smrp all

no debug smrp all

Syntax Description

This command has no arguments or keywords.

Command History

10.0	This command was introduced.
12.2(13)T	This command is no longer supported in Cisco IOS Mainline releases or in Technology-based (T-train) releases. It might continue to appear in 12.2S-family releases.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.

Usage Guidelines

Because the **debug smrp all** command displays all SMRP debugging output, it is processor intensive and should not be enabled when memory is scarce or in very high traffic situations.

For general debugging, use the **debug smrp all** command and turn off excessive transactions with the **no debug smrp transaction** command. This combination of commands will display various state changes and events without displaying every transaction packet. For debugging a specific feature such as a routing problem, use the **debug smrp route** and **debug smrp transaction** commandsto learn if packets are sent and received and which specific routes are affected. The **show smrp traffic** EXEC command is highly recommended as a troubleshooting method because it displays the SMRP counters.

For examples of the type of output you may see, refer to each of the commands listed in the “Related Commands” section.

Related Commands

Command	Description
debug smrp group	Displays information about SMRP group activity.
debug smrp mcache	Displays information about SMRP multicast fast-switching cache entries.
debug smrp neighbor	Displays information about SMRP neighbor activity.
debug smrp port	Displays information about SMRP port activity.

Command	Description
debug smrp route	Displays information about SMRP routing activity.
debug smrp transaction	Displays information about SMRP transactions.

debug smrp group

To display information about SMRP group activity, use the **debug smrp group**privileged EXEC command. The **no** form of this command disables debugging output.

debug smrp group

no debug smrp group

Syntax Description

This command has no arguments or keywords.

Command History

10.0	This command was introduced.
12.2(13)T	This command is no longer supported in Cisco IOS Mainline releases or in Technology-based (T-train) releases. It might continue to appear in 12.2S-family releases.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.

Usage Guidelines

The **debug smrp group**command displays information when a group is created or deleted and when a forwarding entry for a group is created, changed, or deleted. For more information, refer to the **show smrp group** command described in the *Cisco IOS AppleTalk and Novell IPX Command Reference.*

Examples

The following is sample output from the **debug smrp group**command showing a port being created and deleted on group AT 20.34. (AT signifies that this is an AppleTalk network group.)

```
Router#
debug smrp group
SMRP: Group AT 20.34, created on port 20.1 by 20.2
SMRP: Group AT 20.34, deleted on port 20.1
```

The table below lists the messages that may be generated with the **debug smrp group** command concerning the forwarding table.

Table 13: debug smrp group Message Descriptions

Messages	Descriptions
Group <*address*>, deleted on port <*address*>	Group entry was deleted from the group table for the specified port.
Group <*address*>, forward state changed from *state* to *state*	State of the group changed. States are join, forward, and leave.

Messages	Descriptions
Group <*address*>, deleted forward entry	Group was deleted from the forwarding table.
Group <*address*>, created on port <*address*> by <*address*>	Group entry was created in the table for the specified port.
Group <*address*>, added by <*address*> to the group	Secondary router has added this group to its group table.
Group <*address*>, discard join request from <*address*>, not responsible	Discard Join Group request if the router is not the primary router on the local connected network or if it is not the port parent of the route.
Group <*address*>, join request from <*address*>	Request to join the group was received.
Group <*address*>, forward is found	Forward entry for the group was found in the forwarding table.
Group <*address*>, forward state is already joining, ignored	Request to join the group is in progress, so the second request was discarded.
Group <*address*>, no forward found	Forward entry for the group was not found in the forwarding table.
Group <*address*>, join request discarded, fw discarded, fwd parent port not operational	Request to join the group was discarded because the parent port is not available.
Group <*address*>, created forward entry - parent <*address*> child <*address*>	Forward entry was created in the forwarding table for the parent and child address.
Group <*address*>, creator no longer up on <*address*>	Group creator has not been heard from for a specified time and is deemed no longer available.
Group <*address*>, pruning duplicate path on <*address*>	Duplicate path was removed. If we are forwarding and we are a child port, and our port parent address is not pointing to our own port address, we are in a duplicate path.
Group <*address*>, member no longer up on <*address*>	Group member has not been heard from for a specified time and is deemed no longer available.
Group <*address*>, no more child ports in forward entry	Forward entry for group no longer has any child ports. As a result, the forward entry is no longer necessary.

Related Commands

Command	Description
debug sgbp dial-bids	Displays large-scale dial-out negotiations between the primary NAS and alternate NASs.

debug smrp mcache

To display information about SMRP multicast fast-switching cache entries, use the **debug smrp mcache**privileged EXEC command. The **no** form of this command disables debugging output.

debug smrp mcache

no debug smrp mcache

Syntax Description

This command has no arguments or keywords.

Command History

10.0	This command was introduced.
12.2(13)T	This command is no longer supported in Cisco IOS Mainline releases or in Technology-based (T-train) releases. It might continue to appear in 12.2S-family releases.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.

Usage Guidelines

Use the **show smrp mcache** EXEC command (described in the Cisco IOS AppleTalk and Novell IPX Command Reference to display the entries in the SMRP multicast cache, and use the **debug smrp mcache** command to learn whether the cache is being populated and invalidated.

Examples

The following is sample output from the **debug smrp mcache**command. In this example, the cache is created and populated for group AT 11.124. (AT signifies that this is an AppleTalk network group.)

```
Router#
debug smrp mcache
SMRP: Cache created
SMRP: Cache populated for group AT 11.124
        mac - 090007400b7c00000c1740d9
        net - 001fef7500000014ff020a0a0a
SMRP: Forward cache entry created for group AT 11.124
SMRP: Forward cache entry validated for group AT 11.124
SMRP: Forward cache entry invalidated for group AT 11.124
SMRP: Forward cache entry deleted for group AT 11.124
```

The table below lists all the messages that can be generated with the **debug smrp mcache** command concerning the multicast cache.

Table 14: debug smrp mcache Message Descriptions

Messages	Descriptions
Cache populated for group <*address*>	SMRP packet was received on a parent port that has fast switching enabled. As a result, the cache was created and the MAC and network headers were stored for all child ports that have fast switching enabled. Use the **show smrp port appletalk** EXEC command with the optional interface type and number to display the switching path.
Cache memory allocated	Memory was allocated for the multicast cache.
Forward cache entry created/deleted for group <*address*>	Forward cache entry for the group was added to or deleted from the cache.
Forward cache entry validated for group <*address*>	Forward cache entry is validated and is now ready for fast switching.
Forward cache entry invalidated for group <*address*>	Cache entry is invalidated because some change (such as port was shut down) occurred to one of the ports.

Related Commands

Command	Description
debug sgbp dial-bids	Displays large-scale dial-out negotiations between the primary NAS and alternate NASs.

debug smrp neighbor

To display information about SMRP neighbor activity, use the **debug smrp neighbor**privileged EXEC command. The **no** form of this command disables debugging output.

debug smrp neighbor

no debug smrp neighbor

Syntax Description

This command has no arguments or keywords.

Command History

10.0	This command was introduced.
12.2(13)T	This command is no longer supported in Cisco IOS Mainline releases or in Technology-based (T-train) releases. It might continue to appear in 12.2S-family releases.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.

Usage Guidelines

The **debug smrp neighbor**command displays information when a neighbor operating state changes. A neighbor is an adjacent router. For more information, refer to the **show smrp neighbor** EXEC command described in the *Cisco IOS AppleTalk and Novell IPX Command Reference.*

Examples

The following is sample output from the **debug smrp neighbor**command. In this example, the neighbor on port 30.02 has changed state from normal operation to secondary operation.

```
Router#
debug smrp neighbor
SMRP: Neighbor 30.2, state changed from “normal op” to “secondary op”
```

The table below lists all the messages that can be generated with the **debug smrp neighbor**command concerning the neighbor table.

Table 15: debug smrp neighbor Message Descriptions

Messages	Descriptions
Neighbor <*address*>, state changed from *state* to *state*	State of the neighbor changed. States are primary operation, secondary operation, normal operation, primary negotiation, secondary negotiation, and down.
Neighbor <*address*>, neighbor added/deleted	Neighbor was added to or removed from the neighbor table.

Messages	Descriptions
SMRP neighbor up/down	Neighbor is available for service or unavailable.
Neighbor <*address*>, no longer up	Neighbor is unavailable because it has not been heard from for a specified duration.

Related Commands

Command	Description
debug sgbp dial-bids	Displays large-scale dial-out negotiations between the primary NAS and alternate NASs.

debug smrp port

To display information about SMRP port activity, use the **debug smrp port**privileged EXEC command. The **no** form of this command disables debugging output.

debug smrp port

no debug smrp port

Syntax Description

This command has no arguments or keywords.

Command History

10.0	This command was introduced.
12.2(13)T	This command is no longer supported in Cisco IOS Mainline releases or in Technology-based (T-train) releases. It might continue to appear in 12.2S-family releases.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.

Usage Guidelines

The **debug smrp port**command displays information when a port operating state changes. For more information, refer to the **show smrp port** command described in the *Cisco IOS AppleTalk and Novell IPX Command Reference*.

Examples

The following is sample output from the **debug smrp port**command. In this example, port 30.1 has changed state from secondary negative to secondary operation to primary negative:

```
Router#
debug smrp port
SMRP: Port 30.1, state changed from "secondary neg" to "secondary op"
SMRP: Port 30.1, secondary router changed from 0.0 to 30.1
SMRP: Port 30.1, state changed from "secondary op" to "primary neg"
```

The table below lists all the messages that can be generated with the **debug smrp port**command concerning the port table.

Table 16: debug smrp port Message Descriptions

Messages	Descriptions
Port <*address*>, port created/deleted	Port entry was added to or removed from the port table.
Port <*address*>, line protocol changed to *state*	Line protocol for the port is up or down.

Messages	Descriptions
Port <*address*>, state changed from *state* to *state*	State of the port changed. States are primary operation, secondary operation, normal operation, primary negotiation, secondary negotiation, and down.
Port <*address*>, primary/secondary router changed from <*address*>to <*address*>	Primary or secondary port address of the router changed.

Related Commands

Command	Description
debug sgbp dial-bids	Displays large-scale dial-out negotiations between the primary NAS and alternate NASs.

debug smrp route

To display information about SMRP routing activity, use the **debug smrp route**privileged EXEC command. The **no** form of this command disables debugging output.

debug smrp route

no debug smrp route

Syntax Description

This command has no arguments or keywords.

Command History

10.0	This command was introduced.
12.2(13)T	This command is no longer supported in Cisco IOS Mainline releases or in Technology-based (T-train) releases. It might continue to appear in 12.2S-family releases.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.

Usage Guidelines

For more information, refer to the **show smrp route** EXEC command described in the *Cisco IOS AppleTalk and Novell IPX Command Reference.*

Examples

The following is sample output from the **debug smrp route**command. In this example, poison notification is received from port 30.2. Poison notification is the receipt of a poisoned route on a nonparent port.

```
Router#
debug smrp route
SMRP: Route AT 20-20, poison notification from 30.2
SMRP: Route AT 30-30, poison notification from 30.2
```

The table below lists all the messages that can be generated with the **debug smrp route**command concerning the routing table. In the table, the term *route* does not refer to an address but rather to a network range.

Table 17: debug smrp route Message Descriptions

Messages	Descriptions
Route address, deleted/created as local network	Route entry was removed from or added to the routing table.
Route address, from address has invalid distance value	Route entry from the specified address has an incorrect distance value and was ignored.

Messages	Descriptions
Route address, unknown route poisoned by address ignored	Route entry received from the specified address is bad and was ignored.
Route address, created via address - hop number tunnel number	New route entry added to the routing table with the specified number of hops and tunnels.
Route address, from address - overlaps existing route	Route entry received from the specified address overlaps an existing route and was ignored.
Route address, poisoned by address	Route entry has been poisoned by neighbor. Poisoned routes have distance of 255.
Route address, poison notification from address	Poisoned route is received from a nonparent port.
Route address, worsened by parent address	Distance to the route has worsened (become higher), received from the parent neighbor.
Route address, improved via address - number -> number hop, number-> number tunnel	Distance to the route has improved (become lower), received from a neighbor.
Route address, switched to address - higher address than address	Tie condition exists, and because this router had the highest network address, it was used to forward the packet.
Route address, parent port changed address -> address	Parent port address change occurred. The parent port address of a physical network segment determines which router should handle Join Group and Leave Group requests.
SMRP bad distance vector	Packet has an invalid distance vector and was ignored.
Route address, has been poisoned	Route has been poisoned. Poisoned routes are purged from the routing table after a specified time.

Related Commands

Command	Description
debug sgbp dial-bids	Displays large-scale dial-out negotiations between the primary NAS and alternate NASs.

debug smrp transaction

To display information about SMRP transactions, use the **debug smrp transaction**privileged EXEC command. The **no** form of this command disables debugging output.

debug smrp transaction

no debug smrp transaction

Syntax Description

This command has no arguments or keywords.

Examples

The following is sample output from the **debug smrp transaction**command. In this example, a secondary node request is sent out to all routers on port 30.1.

```
Router#
debug smrp transaction
SMRP: Transaction for port 30.1, secondary node request (seq 8435) sent to all routers
SMRP: Transaction for port 30.1, secondary node request (seq 8435) sent to all routers
SMRP: Transaction for port 30.1, secondary node request (seq 8435) sent to all routers
SMRP: Transaction for port 30.1, secondary node request (seq 8435) sent to all routers
```

The table below lists all the messages that can be generated with the **debug smrp route**command.

Table 18: debug smrp Transaction Message Descriptions

Messages	Descriptions
Transaction for port address, packet-type command-type (grp/sec number) sent to/received from address	Port message concerning a packet or command was sent to or received from the specified address.
Transaction for group address on port address, (seq number) sent to/received from address	Group message for a specified port was sent to or received from the specified address.
Unrecognized transaction for port address	Unrecognized message was received and ignored by the port.
Discarded incomplete request	Incomplete message was received and ignored.
Response in wrong state in HandleRequest	Message was received with the wrong state and was ignored.
SMRP bad packet type	SMRP packet was received with a bad packet type and was ignored.
Packet discarded, Bad Port ID	Packet was received with a bad port ID and was ignored.
Packet discarded, Check Packet failed	Packet was received with a failed check packet and was ignored.

Related Commands

Command	Description
debug sgbp dial-bids	Displays large-scale dial-out negotiations between the primary NAS and alternate NASs.

debug snasw dlc

To display frame information entering and leaving the Systems Network Architecture (SNA) switch in real time to the console, use the **debug snasw dlc** command in privileged EXEC mode.

debug snasw dlc detail

Syntax Description

detail	Indicates that in addition to a one-line description of the frame being displayed, an entire hexadecimal dump of the frame will follow.

Command Default

By default, a one-line description of the frame is displayed.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(6)T	This command was introduced.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelin

Caution The **debug snasw dlc** command displays the same trace information available via the **snasw dlctrace** command. The **snasw dlctrace** command is the preferred method for gathering this trace information because it is written to a capture buffer instead of directly to the console. The **debug snasw dlc** command should only be used when it is certain that the output will not cause excessive data to be output to the console.

Examples

The following shows sample output from the **debug snasw dlc** command:

```
Router# debug snasw dlc
Sequence
Number              Size of ISR/
       Link         SNA BTU HPR  Description of frame
343    MVSD     In  sz:134  ISR fmh5 DLUR Rq ActPU NETA.APPNRA29
344    MVSD     Out sz:12   ISR +Rsp IPM     slctd nws:0008
345    @I000002 Out sz:18   ISR Rq ActPU
346    MVSD     Out sz:273  ISR fmh5 TOPOLOGY UPDATE
347    @I000002 In  sz:9    ISR +Rsp Data
348    @I000002 In  sz:12   ISR +Rsp IPM     slctd nws:0002
349    @I000002 In  sz:29   ISR +Rsp ActPU
350    MVSD     Out sz:115  ISR fmh5 DLUR +Rsp ActPU
```

```
351    MVSD     In  sz:12   ISR +Rsp IPM     slctd nws:0007
352    MVSD     In  sz:88   ISR fmh5 DLUR Rq ActLU NETA.MARTLU1
353    MVSD     Out sz:108  ISR fmh5 REGISTER
354    @I000002 Out sz:27   ISR Rq ActLU NETA.MARTLU1
```

Related Commands

Command	Description
snasw dlcfilter	Filters frames traced by the **snasw dlctrace** or **debug snasw dlc** command.
snasw dlctrace	Captures trace frames entering and leaving the SNA Switching Services feature.

debug snasw ips

To display internal signal information between the Systems Network Architecture (SNA) switch and the console in real time, use the **debug snasw ips**command in privileged EXEC mode.

debug snasw dlc

Syntax Description

This command has no arguments or keywords.

Command Default

By default, a one-line description of the interprocess signal is displayed.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(6)T	This command was introduced.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelin

Caution The **debug snasw ips**command displays the same trace information available via the **snasw ipstrace** command. Output from this **debug** command can be large. The **snasw ipstrace**command is the preferred method for gathering this trace information because it is written to a capture buffer instead of directly to the console. The **debug snasw ips** command should only be used when it is certain that the output will not cause excessive data to be output to the console. The **debug snasw dlc** command displays the same trace information available via the **snasw dlctrace** command.

Examples

The following is an example of the **debug snasw ips** command output:

```
Router# debug snasw ips
Sequence
Number                 Sending     Receiving
        Signal Name    Process     Process    Queue
11257 : DEALLOCATE_RCB : --(0) -> RM(2130000) Q 4
11258 : RCB_DEALLOCATED : RM(2130000) -> PS(22E0000) Q 2
11259 : RCB_DEALLOCATED : --(0) -> PS(22E0000) Q 2
11260 : VERB_SIGNAL : PS(22E0000) -> DR(20F0000) Q 2
11261 : FREE_SESSION : --(0) -> RM(2130000) Q 2
11262 : BRACKET_FREED : RM(2130000) -> HS(22FB0001) Q 2
11263 : BRACKET_FREED : --(0) -> HS(22FB0001) Q 2
11264 : VERB_SIGNAL : --(0) -> DR(20F0000) Q 2
11265 : DLC_MU : DLC(2340000) -> PC(22DD0001) Q 2
11266 : DLC_MU : --(0) -> PC(22DD0001) Q 2
```

Related Commands

Command	Description
snasw ipstrace	Captures interprocess signal information between Switching Services components.

debug snmp bulkstat

To enable debugging messages for the Simple Network Management Protocol (SNMP) bulk statistics, use the **debug snmp bulkstat** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug snmp bulkstat

no debug snmp bulkstat

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.0(24)S	This command was introduced.
12.3(2)T	This command was integrated into Cisco IOS Release 12.3(2)T.
12.2(25)S	This command was integrated into Cisco IOS Release 12.2(25)S.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.
12.2(33)SB	This command was integrated into Cisco IOS Release 12.2(33)SB.
Cisco IOS XE Release 2.1	This command was integrated into Cisco IOS Release XE 2.1.

Usage Guidelines

This command is intended primarily for Cisco support personnel. Debugging output for the Periodic MIB Data Collection and Transfer Mechanism (Bulk Statistics feature) includes messages for data collection, local file generation, and transfer attempts.

Examples

In the following example, debugging command output is enabled for the Periodic MIB Data Collection and Transfer Mechanism (Bulk Statistics feature). Note that the references to a VFile indicate a local bulk statistics file, usually followed by the filename. The filename uses the format *specified-filename _device-name* _date_time-stamp.

```
Router# debug snmp
00:17:38:BULKSTAT-DC:Poll timer fired for ifmib
00:17:38:BULKSTAT-DC:In pollDataGroup
00:17:38:BULKSTAT-DC:creating new file
vfile:IfMIB_objects_ios108_030307_101119739
00:17:38:BULKSTAT-DC:Too small state buffer for ifmib
102
```

```
00:17:38:BULKSTAT-DC:Increased buffer state to 1024
00:17:38:BULKSTAT-DC:Interface type data group
00:17:38:BULKSTAT-DC:polling done
00:18:38:BULKSTAT-DC:Poll timer fired for ifmib
00:18:38:BULKSTAT-DC:In pollDataGroup
00:18:38:BULKSTAT-DC:Interface type data group
00:18:38:BULKSTAT-DC:polling done
00:19:26:
BULKSTAT-DC:Collection timer fired for IfMIB_objects
00:19:26:BULKSTAT-TP:Transfer request for
vfile:IfMIB_objects_ios108_030307_101119739
00:19:30:BULKSTAT-TP:written vfile
IfMIB_objects_ios108_030307_101119739
00:19:30:BULKSTAT-TP:retained vfile
vfile:IfMIB_objects_ios108_030307_101119739
00:19:38:BULKSTAT-DC:Poll timer fired for ifmib
00:19:38:BULKSTAT-DC:In pollDataGroup
00:19:38:BULKSTAT-DC:creating new file
vfile:IfMIB_objects_ios108_030307_101319739
00:19:38:BULKSTAT-DC:Interface type data group
00:19:38:BULKSTAT-DC:polling done
00:20:38:BULKSTAT-DC:Poll timer fired for ifmib
00:20:38:BULKSTAT-DC:In pollDataGroup
00:20:38:BULKSTAT-DC:Interface type data group
00:20:38:BULKSTAT-DC:polling done
00:21:38:BULKSTAT-DC:Poll timer fired for ifmib
00:21:38:BULKSTAT-DC:In pollDataGroup
00:21:38:BULKSTAT-DC:Interface type data group
00:21:38:BULKSTAT-DC:polling done
00:22:26:
BULKSTAT-DC:Collection timer fired for IfMIB_objects
00:22:26:BULKSTAT-TP:Transfer request for
vfile:IfMIB_objects_ios108_030307_101319739
00:22:26:BULKSTAT-TP:written vfile
IfMIB_objects_ios108_030307_101319739
00:22:26:BULKSTAT-TP:retained vfile
vfile:IfMIB_objects_ios108_030307_101319739
00:22:38:BULKSTAT-DC:Poll timer fired for ifmib
00:22:38:BULKSTAT-DC:In pollDataGroup
00:22:38:BULKSTAT-DC:creating new file
vfile:IfMIB_objects_ios108_030307_101619739
00:22:38:BULKSTAT-DC:Interface type data group
00:22:38:BULKSTAT-DC:polling done
00:23:38:BULKSTAT-DC:Poll timer fired for ifmib
00:23:38:BULKSTAT-DC:In pollDataGroup
00:23:38:BULKSTAT-DC:Interface type data group
00:23:38:BULKSTAT-DC:polling done
00:24:38:BULKSTAT-DC:Poll timer fired for ifmib
00:24:38:BULKSTAT-DC:In pollDataGroup
00:24:38:BULKSTAT-DC:Interface type data group
00:24:38:BULKSTAT-DC:polling done
00:25:26:
BULKSTAT-DC:Collection timer fired for IfMIB_objects
00:25:26:BULKSTAT-TP:Transfer request for
vfile:IfMIB_objects_ios108_030307_101619739
00:25:26:BULKSTAT-TP:written vfile
IfMIB_objects_ios108_030307_101619739
00:25:26:BULKSTAT-TP:retained vfile
vfile:IfMIB_objects_ios108_030307_101619739
00:25:38:BULKSTAT-DC:Poll timer fired for ifmib
00:25:38:BULKSTAT-DC:In pollDataGroup
00:25:38:BULKSTAT-DC:creating new file
vfile:IfMIB_objects_ios108_030307_101919739
00:25:38:BULKSTAT-DC:Interface type data group
00:25:38:BULKSTAT-DC:polling done
00:26:38:BULKSTAT-DC:Poll timer fired for ifmib
00:26:38:BULKSTAT-DC:In pollDataGroup
00:26:38:BULKSTAT-DC:Interface type data group
00:26:38:BULKSTAT-DC:polling done
```

Related Commands

Command	Description
show snmp mib bulkstat transfer	Displays the transfer status of files generated by the Periodic MIB Data Collection and Transfer Mechanism.
snmp mib bulkstat transfer	Names a bulk statistics transfer configuration and enters Bulk Statistics Transfer configuration mode.

debug snmp detail

To display the Simple Network Management Protocol (SNMP) debug messages, use the **debug snmp detail**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug snmp detail

no debug snmp detail

Syntax Description

This command has no arguments or keywords.

Command Default

SNMP debug messages are not displayed.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.4(20)T	This command was introduced.
12.2(33)SRE	This command was integrated into Cisco IOS Release 12.2(33)SRE.

Usage Guidelines

Before running the **debug snmp detail**command, connect the device to the Network Management System (NMS). The command output displays the debug messages for errors occurred during SNMP operations. The debug messages help in identifying and debugging errors.

Examples

The following is sample output from the **debug snmp detail**command:

```
Router# debug snmp detail

SNMP Detail Debugs debugging is on
process_mgmt_req_int: UDP packet being de-queued
findContextInfo: Authentication failure, bad community string
SrDoSnmp: Bad Community name.
process_mgmt_req_int: UDP packet being de-queued
SrParseV3SnmpMessage: No matching Engine ID.
SrParseV3SnmpMessage: Failed.
SrDoSnmp: authentication failure, Unknown Engine ID
process_mgmt_req_int: UDP packet being de-queued
ParseSequence, Unexpected type: 4
SrParseV3SnmpMessage: ParseSequence:
SrParseV3SnmpMessage: Failed.
SrDoSnmp: authentication failure, Unsupported security modelQ:
```

Related Commands

Command	Description
debug snmp packet	Displays information about every SNMP packet sent or received by the router.

debug snmp mib nhrp

To display messages about Simple Network Management Protocol (SNMP) Next Hop Resolution Protocol (NHRP) MIB, use the **debug snmp mib nhrp**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug snmp mib nhrp {**error**| **events**| **internal**| **notif [detail]**}

no debug snmp mib nhrp {**error**| **events**| **internal**| **notif [detail]**}

Syntax Description

error	Displays messages about SNMP NHRP MIB error events, including error information about packet processing or MIB special events.
events	Displays messages about SNMP NHRP MIB events, from the NHRP MIB tree data-structures and SNMP query-related events.
internal	Displays messages about SNMP NHRP MIB engineering events.
notif	Displays debug messages related to SNMP NHRP MIB notification events.
detail	(Optional) Displays detailed messages related to SNMP NHRP MIB notification events.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.4(20)T	This command was introduced.
15.0(1)M	This command was modified. The **notif** and **detail** keywords were added.

Usage Guidelines

The debug snmp mib nhrp internal command can generate many output messages. Due to the increased command processing and its effect on system usage, the use of this command is not advisable under normal circumstances.

Examples

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

```
*May 10 12:52:01.245: NHRP_SNMP-NOTIF[1488]: Retrieved values from instrumentation
*May 10 12:52:01.245: NHRP_SNMP-NOTIF[1646]: Varbind list created
*May 10 12:52:01.245: NHRP_SNMP-NOTIF[1665]: NHRP trap queued: cneNotifNextHopRegClientUp
```

The following is sample output from the **debug snmp mib nhrp notif detail**command:

```
*May 10 12:52:44.461: NHRP_SNMP-NOTIF[695]: Address parameters'
extraction for local and remote endpoints successful
*May 10 12:52:44.461: NHRP_SNMP-NOTIF[1488]: Retrieved values from instrumentation
*May 10 12:52:44.461: NHRP_SNMP-NOTIF[1589]: Instance OIDs populated
*May 10 12:52:44.461: NHRP_SNMP-NOTIF[1608]: Value types  and values populated
*May 10 12:52:44.461: NHRP_SNMP-NOTIF[1625]: Varbind created for
nhrpServerInternetworkAddrType
*May 10 12:52:44.461: NHRP_SNMP-NOTIF[1643]: Varbind created for nhrpServerInternetworkAddr
*May 10 12:52:44.461: NHRP_SNMP-NOTIF[1643]: Varbind created for nhrpServerNbmaAddrType
*May 10 12:52:44.461: NHRP_SNMP-NOTIF[1643]: Varbind created for nhrpServerNbmaAddr
*May 10 12:52:44.461: NHRP_SNMP-NOTIF[1643]: Varbind created for nhrpServerNbmaSubaddr
*May 10 12:52:44.461: NHRP_SNMP-NOTIF[1643]: Varbind created for
nhrpServerNhcInternetworkAddrType
*May 10 12:52:44.461: NHRP_SNMP-NOTIF[1643]: Varbind created for nhrpServerNhcInternetworkAddr
*May 10 12:52:44.461: NHRP_SNMP-NOTIF[1643]: Varbind created for nhrpServerNhcNbmaAddrType
*May 10 12:52:44.461: NHRP_SNMP-NOTIF[1643]: Varbind created for nhrpServerNhcNbmaAddr
*May 10 12:52:44.461: NHRP_SNMP-NOTIF[1643]: Varbind created for nhrpServerNhcNbmaSubaddr
*May 10 12:52:44.461: NHRP_SNMP-NOTIF[1643]: Varbind created for nhrpServerNhcPrefixLength
*May 10 12:52:44.461: NHRP_SNMP-NOTIF[1643]: Varbind created for nhrpServerNhcInUse
*May 10 12:52:44.461: NHRP_SNMP-NOTIF[1643]: Varbind created for nhrpServerCacheUniqueness
*May 10 12:52:44.461: NHRP_SNMP-NOTIF[1646]: Varbind list created
*May 10 12:52:44.461: NHRP_SNMP-NOTIF[1665]: NHRP trap queued: cneNotifNextHopRegClientUp
```

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

```
Router# debug snmp mib nhrp events
*Apr 10 13:34:46.175: NHRP_SNMP-EVE[2097]: In Get nhrpClientEntry for VRFID [0] ClientIndex
 [0]  NHS [0] Req [1]
*Apr 10 13:34:46.175: NHRP_SNMP-EVE[2148]: In here as expected.
*Apr 10 13:34:46.175: NHRP_SNMP-EVE[1050]: In Extract Client Entry Info
*Apr 10 13:34:46.223: NHRP_SNMP-EVE[2097]: In Get nhrpClientEntry for VRFID [0] ClientIndex
 [2]  NHS [0] Req [1]
*Apr 10 13:34:46.223: NHRP_SNMP-EVE[2140]: Could not find the Node
*Apr 10 13:34:46.223: NHRP_SNMP-EVE[2097]: In Get nhrpClientEntry for VRFID [0] ClientIndex
 [0]  NHS [0] Req [1]
*Apr 10 13:34:46.223: NHRP_SNMP-EVE[2148]: In here as expected.
*Apr 10 13:34:46.223: NHRP_SNMP-EVE[1050]: In Extract Client Entry Info
```

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

```
Router# debug snmp mib nhrp internal
*Apr 10 13:36:33.267: NHRP_SNMP-INTR[2089]: In nhrpClientEntry
*Apr 10 13:36:33.323: NHRP_SNMP-INTR[2089]: In nhrpClientEntry
*Apr 10 13:36:33.323: NHRP_SNMP-INTR[2089]: In nhrpClientEntry
```

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

Table 19: debug snmp mib nhrp Field Descriptions

Field	Description
NHRP_SNMP-ERR[ ]	Indicates output from the **debug snmp mib nhrp error command.**
NHRP_SNMP-EVE[2097]	Indicates output from the **debug snmp mib nhrp events**command.

Field	Description
NHRP_SNMP-INTR[2089]	Indicates output from the **debug snmp mib nhrp internal command.**
NHRP_SNMP-NOTIF[1488]	Indicates output from the **debug snmp mib nhrp notif command.**

Related Commands

Command	Description
show snmp mib nhrp status	Indicates the status of the NHRP MIB and whether the NHRP MIB is enabled or disabled.

debug snmp overhead

To display the list of Simple Network Management Protocol (SNMP) MIBs that take more than the threshold time to perform an SNMP get or get-next operation, use the **debug snmp overhead**command in privileged EXEC mode. To disable debugging, use the **no** form of this command.

debug snmp overhead

no debug snmp overhead

Syntax Description

This command has no arguments or keywords.

Command Default

SNMP debug messages are not displayed.

Command Modes

Privileged EXEC (#)

Command History

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

Examples

The following is sample output from the **debug snmp overhead** command:

```
Router# debug snmp overhead
SNMP overhead debugging is on
*Nov 11 16:35:02.579 PDT:  Process exceeds 1000ms threshold (200ms IOS quantum)
*Nov 11 16:35:02.579 PDT:  GETNEXT of ciscoFlashFileEntry.2.1.1.1--result
ciscoFlashFileEntry.2.1.1.2
```

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

Table 20: debug snmp overhead Field Descriptions

Field	Description
Process exceeds 1000ms threshold	Processing time for the SNMP get-next operation is more than 1000 milliseconds.
200ms IOS quantum	Threshold time in milliseconds.
GETNEXT of ciscoFlashFileEntry.2.1.1.1	The OID ciscoFlashFileEntry.2.1.1.1 is queried using the get-next operation.
result ciscoFlashFileEntry.2.1.1.2	The result of the get-next operation is ciscoFlashFileEntry.2.1.1.2, which is the next value of the OID being queried.

debug snmp packet

To display information about every Simple Network Management Protocol (SNMP) packet sent or received by the router, use the **debug snmp packet** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug snmp packet

no debug snmp packet

Syntax Description

This command has no arguments or keywords.

Command Default

The command is disabled by default.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.0(24)S	This command was introduced.
12.3(2)T	This command was integrated into Cisco IOS Release 12.3(2)T.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.
12.2(33)SB	This command was integrated into Cisco IOS Release 12.2(33)SB.
Cisco IOS XE Release 2.5	This command was implemented on Cisco ASR 1000 series routers.

Examples

The following is sample output from the **debug snmp packet**command. In this example, the router receives a get-next request from the host at 192.10.2.10 and responds with the requested information.

```
Router# debug snmp packet
SNMP: Packet received via UDP from 192.10.2.10 on Ethernet0
SNMP: Get-next request, reqid 23584, errstat 0, erridx 0
 sysUpTime = NULL TYPE/VALUE
 system.1 = NULL TYPE/VALUE
 system.6 = NULL TYPE/VALUE
SNMP: Response, reqid 23584, errstat 0, erridx 0
 sysUpTime.0 = 2217027
 system.1.0 = Cisco Internetwork Operating System Software
 system.6.0 =
SNMP: Packet sent via UDP to 192.10.2.10
```

Based on the kind of packet sent or received, the output may vary. For get-bulk requests, a line similar to the following is displayed:

```
SNMP: Get-bulk request, reqid 23584, nonrptr 10, maxreps 20
```

For traps, a line similar to the following is displayed:

```
SNMP: V1 Trap, ent 1.3.6.1.4.1.9.1.13, gentrap 3, spectrap 0
```

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

Table 21: debug snmp packet Field Descriptions

Field	Description
Get-next request	Indicates what type of SNMP protocol data unit (PDU) the packet is. Possible types are as follows: • Get request • Get-next request • Response • Set request • V1 Trap • Get-bulk request • Inform request • V2 Trap Depending on the type of PDU, the rest of this line displays different fields. The indented lines following this line list the MIB object names and corresponding values.
reqid	Request identification number. This number is used by the SNMP manager to match responses with requests.
errstat	Error status. All PDU types other than response will have an errstat of 0. If the agent encounters an error while processing the request, it will set errstat in the response PDU to indicate the type of error.
erridx	Error index. This value will always be 0 in all PDUs other than responses. If the agent encounters an error, the erridx will be set to indicate which varbind in the request caused the error. For example, if the agent had an error on the second varbind in the request PDU, the response PDU will have an erridx equal to 2.

Field	Description
nonrptr	Nonrepeater value. This value and the maximum repetition value are used to determine how many varbinds are returned. Refer to RFC 1905 for details.
maxreps	Maximum repetition value. This value and the nonrepeater value are used to determine how many varbinds are returned. Refer to RFC 1905 for details.
ent	Enterprise object identifier. Refer to RFC 1215 for details.
gentrap	Generic trap value. Refer to RFC 1215 for details.
spectrap	Specific trap value. Refer to RFC 1215 for details.

debug snmp requests

To display information about every Simple Network Management Protocol (SNMP) request made by the SNMP manager, use the **debug snmp requests** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug snmp requests

no debug snmp requests

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Examples

The following is sample output from the **debug snmp requests**command:

```
Router# debug snmp requests
SNMP Manager API: request
  dest: 171.69.58.33.161, community: public
  retries: 3, timeout: 30, mult: 2, use session rtt
  userdata: 0x0
```

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

Table 22: debug snmp requests Field Descriptions

Field	Description
SNMP Manager API	Indicates that the router sent an SNMP request.
dest	Destination of the request.
community	Community string sent with the request.
retries	Number of times the request has been re-sent.
timeout	Request timeout, or how long the router will wait before resending the request.
mult	Timeout multiplier. The timeout for a re-sent request will be equal to the previous timeout multiplied by the timeout multiplier.
use session rtt	Indicates that the average round-trip time of the session should be used in calculating the timeout value.
userdata	Internal Cisco IOS software data.

debug snmp sync

To debug Simple Network Management Protocol (SNMP) synchronization and faults in synchronization, use the **d ebug snmp sync** command in privileged EXEC mode. To disable the display of debugging output, use the **no** form of this command.

debug snmp sync

no debug snmp sync

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(22)S	This command was introduced.
12.2(18)S	This command was integrated into Cisco IOS Release 12.2(18)S.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(31)SB2	This command was integrated into Cisco IOS Release 12.2(31)SB2.

Usage Guidelines

The **debug snmp sync** command can be used to debug SNMP synchronization and faults in synchronization. The standby Route Processor (RP) may sometimes reset as a result of synchronization faults. If the fault occurs when SNMP activities such as SNMP sets are in progress, enter the **debug snmp sync** command to identify whether a synchronization fault caused the reset.

SNMP synchronizations (dynamic and bulk) are performed only if the router is configured to be in stateful switchover (SSO) mode.

Examples

The following example enables debugging of SNMP synchronization activity:

```
Router# debug snmp sync
```

Related Commands

Command	Description
debug snmp packets	Displays information about every SNMP packet sent or received by the networking device.

Command	Description
mode	Configures the redundancy mode of operation.

debug snmp tunnel-mib

To enable the debugging for configuring the IP Tunnel Management Information Base (MIB) through Simple Network Management Protocol (SNMP), use the **debug snmp tunnel-mib** command in privileged EXEC mode. To disable debugging, use the **no** form of this command.

debug snmp tunnel-mib

no debug snmp tunnel-mib

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(33)SRB	This command was introduced.
12.4(15)T	This command was integrated into Cisco IOS Release 12.4(15)T.
12.2(33)SB1	This command was integrated into Cisco IOS Release 12.2(33)SB1.
12.2(44)SG	This command was integrated into Cisco IOS Release 12.2(44)SG.
Cisco IOS Release XE 2.1	This command was integrated into Cisco IOS Release XE 2.1.

Usage Guidelines

Use the **debug snmp tunnel-mib** command to verify whether a tunnel is created or deleted.

Examples

The following is sample output from the **debug snmp tunnel-mib** command. The output shows that a tunnel is created through SNMP.

```
Router# debug snmp tunnel-mib
SNMP TUNNEL-MIB debugging is on
k_tunnelInetConfigEntry_get: Entering
k_tunnelInetConfigEntry_get: Exact search
tim_client_tunnel_endpoint_data_get: Entering
tim_client_tunnel_endpoint_data_get: Exact search
tim_client_tunnel_endpoint_data_get: No element found
k_tunnelInetConfigEntry_get: Client service failed
k_tunnelInetConfigEntry_test: Entering
k_tunnelInetConfigEntry_test: Completed
k_tunnelInetConfigEntry_set: Entering
tim_client_tunnel_endpoint_data_get: Entering
tim_client_tunnel_endpoint_data_get: Exact search
tim_client_tunnel_endpoint_data_get: No element found
k_tunnelInetConfigEntry_set: Calling tunnel create
tim_client_tunnel_create: Entering
tim_client_tunnel_create: Completed
```

debug sntp adjust through debug tag-switching xtagatm vc

- debug sntp adjust, page 106
- debug sntp packets, page 107
- debug sntp select, page 109
- debug software authenticity, page 110
- debug source bridge, page 114
- debug source error, page 116
- debug source event, page 118
- debug span, page 125
- debug spanning-tree, page 128
- debug ss7 mtp1, page 130
- debug ss7 mtp2, page 134
- debug ss7 sm, page 141
- debug sse, page 143
- debug ssg ctrl-errors, page 144
- debug ssg ctrl-events, page 146
- debug ssg ctrl-packets, page 147
- debug ssg data, page 149
- debug ssg data-nat, page 151
- debug ssg dhcp, page 153
- debug ssg errors, page 155
- debug ssg events, page 156
- debug ssg packets, page 158
- debug ssg port-map, page 160

- debug ssg tcp-redirect, page 162
- debug ssg transparent login, page 165
- debug ssl, page 167
- debug ssl openssl, page 169
- debug ssm, page 171
- debug sss aaa authorization event, page 175
- debug sss aaa authorization fsm, page 178
- debug sss error, page 180
- debug sss event, page 182
- debug sss fsm, page 184
- debug standby, page 186
- debug standby errors, page 189
- debug standby events, page 191
- debug standby events icmp, page 195
- debug standby events neighbor, page 197
- debug standby packets, page 199
- debug stun packet, page 201
- debug subscriber aaa authorization, page 204
- debug subscriber error, page 206
- debug subscriber event, page 207
- debug subscriber feature, page 208
- debug subscriber fsm, page 211
- debug subscriber packet, page 212
- debug subscriber policy, page 214
- debug subscriber service, page 217
- debug subscriber testing, page 219
- debug sw56, page 220
- debug syscon perfdata, page 221
- debug syscon sdp, page 222
- debug syslog-server, page 223
- debug tacacs, page 225
- debug tacacs events, page 227
- debug tag-switching atm-cos, page 230

- debug tag-switching atm-tdp api, page 231
- debug tag-switching atm-tdp routes, page 232
- debug tag-switching atm-tdp states, page 233
- debug tag-switching tdp advertisements, page 234
- debug tag-switching tdp bindings, page 235
- debug tag-switching tdp directed-neighbors, page 236
- debug tag-switching tdp peer state-machine, page 237
- debug tag-switching tdp pies received, page 238
- debug tag-switching tdp pies sent, page 239
- debug tag-switching tdp session io, page 240
- debug tag-switching tdp session state-machine, page 241
- debug tag-switching tdp transport connections, page 242
- debug tag-switching tdp transport events, page 243
- debug tag-switching tdp transport timers, page 244
- debug tag-switching xtagatm cross-connect, page 246
- debug tag-switching xtagatm errors, page 247
- debug tag-switching xtagatm events, page 248
- debug tag-switching xtagatm vc, page 249

debug sntp adjust

To display information about Simple Network Time Protocol (SNTP) clock adjustments, use the **debug sntp adjust** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug sntp adjust

no debug sntp adjust

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Examples

The following is sample output from the **debug sntp adjust** command when an offset to the time reported by the configured NTP server is calculated. The offset indicates the difference between the router time and the actual time (as kept by the server) and is displayed in milliseconds. The clock time is then successfully changed to the accurate time by adding the offset to the current router time.

```
Router# debug sntp adjust
Delay calculated, offset 3.48
Clock slewed.
```

The following is sample output from the **debug sntp adjust** command when an offset to the time reported by a broadcast server is calculated. Because the packet is a broadcast packet, no transmission delay can be calculated. However, in this case, the offset is too large, so the clock is reset to the correct time.

```
Router# debug sntp adjust
No delay calculated, offset 11.18
Clock stepped.
```

debug sntp packets

To display information about Simple Network Time Protocol (SNTP) packets sent and received, use the **debug sntp packets** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug sntp packets

no debug sntp packets

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Examples

The following is sample output from the **debug sntp packets** command when a message is received:

```
Router# debug sntp packets
Received SNTP packet from 172.16.186.66, length 48
 leap 0, mode 1, version 3, stratum 4, ppoll 1024
 rtdel 00002B00, rtdsp 00003F18, refid AC101801 (172.16.24.1)
 ref B7237786.ABF9CDE5 (23:28:06.671 UTC Tue May 13 1997)
 org 00000000.00000000 (00:00:00.000 UTC Mon Jan 1 1900)
 rec 00000000.00000000 (00:00:00.000 UTC Mon Jan 1 1900)
 xmt B7237B5C.A7DE94F2 (23:44:28.655 UTC Tue May 13 1997)
 inp AF3BD529.810B66BC (00:19:53.504 UTC Mon Mar 1 1993)
```

The following is sample output from the **debug sntp packets** command when a message is sent:

```
Router# debug sntp packets
Sending SNTP packet to 172.16.25.1
 xmt AF3BD455.FBBE3E64 (00:16:21.983 UTC Mon Mar 1 1993)
```

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

Table 23: debug sntp packets Field Descriptions

Field	Description
length	Length of the SNTP packet.
leap	Indicates if a leap second will be added or subtracted.
mode	Indicates the mode of the router relative to the server sending the packet.
version	SNTP version number of the packet.
stratum	Stratum of the server.
ppoll	Peer polling interval.
rtdel	Total delay along the path to the root clock.

Field	Description
rtdsp	Dispersion of the root path.
refid	Address of the server that the router is currently using for synchronization.
ref	Reference time stamp.
org	Originate time stamp. This value indicates the time the request was sent by the router.
rec	Receive time stamp. This value indicates the time the request was received by the SNTP server.
xmt	Transmit time stamp. This value indicates the time the reply was sent by the SNTP server.
inp	Destination time stamp. This value indicates the time the reply was received by the router.

debug sntp select

To display information about Simple Network Time Protocol (SNTP) server selection, use the **debug sntp select** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug sntp select

no debug sntp select

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Examples

The following is sample output from the **debug sntp select** command. In this example, the router will synchronize its time to the server at 172.16.186.66.

```
Router# debug sntp select
SNTP: Selected 172.16.186.66
```

debug software authenticity

To debug software authenticity events, use the **debug software authenticity** command in priveleged EXEC mode. To disable debugging, use the **no** form of this command.

debug software authenticity {**envelope**| **errors**| **key**| **revocation**| **show**| **verbose**}

no debug software authenticity {**envelope**| **errors**| **key**| **revocation**| **show**| **verbose**}

Syntax Description

envelope	Enables the display of all debugging output related to software authentication envelope events.
errors	Enables the display of all debugging output related to software authentication errors.
key	Enables the display of all debugging output related to software authentication key events.
revocation	Enables the display of all debugging output related to software authentication revocation events.
show	Enables the display of all debugging output related to the show software authenticity file, show software authenticity keys, and show software authenticity running commands.
verbose	Enables the display of all debugging output related to software authentication errors and events.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
15.0(1)M	This command was introduced for the Cisco 1941, 2900, and 3900 routers.
15.0(1)M2	This command was modified. The revocation keyword was added.
15.1(1)T	This command was integrated into Cisco IOS Release 15.1(1)T.

Usage Guidelines

Use the debug software authenticity command to enable debugging related to software authentication events.

Use the command in conjunction with the show software authenticity file, show software authenticity keys, show software authenticity running, and show software authenticity upgrade-status commands in order to display the debugging-related messages. For further information on these commands, see the Cisco IOS Master Command List, All Releases.

Examples

```
The following example enables the display of debugging output related to software
authentication errors:
Router# debug software authenticity errors
```

Software Authenticity Errors debugging is on

The following example enables the display of debugging output related to software authentication key errors, and the output from the show software authenticity keys command displays the key information related to software authentication debugging:

```
Router# debug software authenticity key
Software Authenticity Key debugging is on
Router# show software authenticity keys
Public Key #1 Information
-------------------------
Key Type             : Release  (Primary)
Public Key Algorithm : RSA
Modulus :
        CC:CA:40:55:8C:71:E2:4A:3A:B6:9D:5C:94:1D:02:BA:
        .....
        26:04:6B:33:EB:70:2B:18:24:C7:D9:31:3E:77:24:85
Exponent : xxx
Key
*May 14 23:23:13.988: code_sign_parse_key_record: START. list offset:(0), tlv tag: 0xAE,
tlv len: 281
*May 14 23:23:13.988: code_sign_parse_key_record: Tag (0xAE) found at offset: 0, list_offset:
 0
*May 14 23:23:13.988: code_sign_parse_key_record: key_rec_len: 281, pub key size: 288,
offset: 3
*May 14 23:23:13.988: code_sign_parse_key_record: Key Start magic: 0xxxxxxxD, at offset: 3
*May 14 23:23:13.988: code_sign_validate_key_end_magic: End Magic (0xBEEFCAFE) found at the
 end of the key record (292)
*May 14 23:23:13.988: code_sign_parse_key_record: Tlv start offset: 7, pub key size: 288
*May 14 23:23:13.988: code_sign_parse_key_record: Tag (Key Type:(0x1) found at offset: 7
*May 14 23:23:13.988: code_sign_parse_key_record: We increment offset by sizeof tlv: 3,
size of len: 2
*May 14 23:23:13.988: code_sign_parse_key_record: Key Type: 0x1, offset: 11
*May 14 23:23:13.988: code_sign_parse_key_record: Tag (Signature Algorithm:(0x2) found at
offset: 11
*May 14 23:23:13.988: code_sign_parse_key_record: We increment offset by sizeof tlv: 3,
size of len: 2
*May 14 23:23:13.988: code_sign_parse_key_record: Signature Algo: 0x1, offset: 15
*May 14 23:23:13.988: code_sign_parse_key_record: Tag (Key Info Length:(0x3) found at offset:
 15
*May 14 23:23:13.988: code_sign_parse_key_record: We increment offset by sizeof tlv: 3,
size of len: 2
*May 14 23:23:13.988: code_sign_parse_key_record:Length (266) for type (Key Info Length),
offset: 18
*May 14 23:23:13.988: code_sign_parse_key_record: Key Info Len: 266, offset: 18
*May 14 23:23:13.988: code_sign_parse_key_record: Tag (Modulus:(xxx) found at offset: 18
*May 14 23:23:13.988: code_sign_parse_key_record: We increment offset by sizeof tlv: 3,
size of len: 2
*May 14 23:23:13.988: code_sign_parse_key_record: offset: 277,  Modulus size: (xxx)
CCCA40558C71E24A3AB69D5C941D02BA63CDF0202FC6CBC1D73E8F27E3DA6DC615EB2FD0A66643D82BE17F3CE8.....
47AE5135955C58B164320B925608DA4002B75FB01EFEC2691B188D6FB2E3AFE8F453888FE063B4304DDC2EB25B
*May 14 23:23:13.988: code_sign_parse_key_record: Tag (Public Exponent:(xxx) found at offset:
 277
*May 14 23:23:13.988: code_sign_parse_key_record: We increment offset by sizeof tlv: 3,
size of len: 2
*May 14 23:23:13.988: code_sign_parse_key_record: offset: 284,  Public Exponent size: (xxx),
 public exponent: xxx
*May 14 23:23:13.988: code_sign_parse_key_record: Tag (Key Version:(0x6) found at offset:
284
*May 14 23:23:13.988: code_sign_parse_key_record: We increment offset by sizeof tlv: 3,
size of len: 2
*May 14 23:23:13.988: code_sign_parse_key_record: Key Version: 0x41, offset: 288
*May 14 23:23:13.988: code_sign_parse_key_record: END. offset (292), bitlist: (0x3F)Version
          : A
```

The following example enables the display of debugging output related to software authentication errors and events (the full range of messages), and the output from the show software authenticity file command displays the file information related to software authentication debugging:

```
Router# debug software authenticity verbose
Software Authenticity Verbose debugging is on
Router# show software authenticity file flash0:c3900-universalk9-mz.SSA
                                ##################
                                Signature Envelope
Version 1.xxx
hdr_length xxx
signer_id_len xxx
signer_name CN=CiscoSystems;OU=C3900;O=CiscoSystems
ca serial num len xxx
ca_serial_num xxx
ca_name CN=CiscoSystems;OU=C3900;O=CiscoSystems
digest_algo xxx
sign_algo xxx
mod_size xxx
key_type xxx
key_version 0xx1
signature length xxx
signature TLV offset xxx
signature
4F94AC7EAA7B9B9EAE66EFA8BF426C3BFE622D7C651A35F686F7DD7FBF329317B269CAEADB5679834B93BF2C91.....
F160EF79B82AB41176975D024D1DA9EB75499BC139BFED9AF8D3F4DFAE35BFC0CDA1519F7CD9C8EB08D8D09D18
 --More--
*May 28 08:05:44.487: code_sign_get_image_type: filename:flash0:c3900-universalk9-mz.SSA
*May 28 08:05:44.487: cs_open: Opened file flash0:c3900-universalk9-mz.SSA with fd=13
*May 28 08:05:44.491: code_sign_get_image_type: image type found: image (elf) (3)
*May 28 08:05:44.491: code_sign_get_image_envelope Start, fd(13)
*May 28 08:05:44.491: code_sign_get_number_of_sections num_sections: 7
*May 28 08:05:44.547: code_sign_get_image_envelope:SHA2 Note Section found at iter: 6
*May 28 08:05:44.547: code_sign_get_image_envelope: Note name len(n_namesz): 13, Signature
 Env Len(n_descz): 388
*May 28 08:05:44.547: code_sign_get_image_envelope: sizeof elf_note_hdr: 12, size of
Elf32_Nhdr: 12
*May 28 08:05:44.547: code_sign_get_image_envelope: Note Name:(CISCO SYSTEMS) fo
                       ##################
File Name                     : flash0:c3900-universalk9-mz.SSA
Image type                    : Development
    Signer Information
        Common Name           : xxx
        Organization Unit     : xxx
        Organization Name     : xxx
    Certificate Serial Number : xxx
    Hash Algorithm            : SHA512
    Signature Algorithm       : 2048-bit RSA
    Key Version               : A
```

Related Commands

Command	Description
show software authenticity file	Displays information related to software authentication for the loaded image file.
show software authenticity keys	Displays the software public keys that are in the storage with the key types.

Command	Description
show software authenticity running	Displays software authenticity information for the current ROMmon and Cisco IOS image used for booting.
show software authenticity upgrade-status	Displays software authenticity information indicating if the digitally signed software has been signed with a new production key after a production key revocation.

debug source bridge

To display information about packets and frames transferred across a source-route bridge, use the **debug source bridge** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug source bridge

no debug source bridge

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Examples

The following is sample output from the **debug source bridge** command for peer bridges using TCP as a transport mechanism. The remote source-route bridging (RSRB) network configuration has ring 2 and ring 1 bridged together through remote peer bridges. The remote peer bridges are connected via a serial line and use TCP as the transport mechanism.

```
Router# debug source bridge
RSRB: remote explorer to 5/192.108.250.1/1996 srn 2 [C840.0021.0050.0000]
RSRB: Version/Ring XReq sent to peer 5/192.108.250.1/1996
RSRB: Received version reply from 5/192.108.250.1/1996 (version 2)
RSRB: DATA: 5/192.108.250.1/1996 Ring Xchg Rep, trn 2, vrn 5, off 18, len 10
RSRB: added bridge 1, ring 1 for 5/192.108.240.1/1996
RSRB: DATA: 5/192.108.250.1/1996 Explorer trn 2, vrn 5, off 18, len 69
RSRB: DATA: 5/192.108.250.1/1996 Forward trn 2, vrn 5, off 0, len 92
RSRB: DATA: forward Forward srn 2, br 1, vrn 5 to peer 5/192.108.250.1/1996
```

The following line indicates that a remote explorer frame has been sent to IP address 192.108.250.1 and, like all RSRB TCP connections, has been assigned port 1996. The bridge belongs to ring group 5. The explorer frame originated from ring 2. The routing information field (RIF) descriptor has been generated by the local station and indicates that the frame was sent out via bridge 1 onto virtual ring 5.

```
RSRB: remote explorer to 5/192.108.250.1/1996 srn 2 [C840.0021.0050.0000]
```

The following line indicates that a request for remote peer information has been sent to IP address 192.108.250.1, TCP port 1996. The bridge belongs to ring group 5.

```
RSRB: Version/Ring XReq sent to peer 5/192.108.250.1/1996
```

The following line is the response to the version request previously sent. The response is sent from IP address 192.108.250.1, TCP port 1996. The bridge belongs to ring group 5.

```
RSRB: Received version reply from 5/192.108.250.1/1996 (version 2)
```

The following line is the response to the ring request previously sent. The response is sent from IP address 192.108.250.1, TCP port 1996. The target ring number is 2, virtual ring number is 5, the offset is 18, and the length of the frame is 10 bytes.

```
RSRB: DATA: 5/192.108.250.1/1996 Ring Xchg Rep, trn 2, vrn 5, off 0, len 10
```

The following line indicates that bridge 1 and ring 1 were added to the source-bridge table for IP address 192.108.250.1, TCP port 1996:

```
RSRB: added bridge 1, ring 1 for 5/192.108.250.1/1996
```

The following line indicates that a packet containing an explorer frame came across virtual ring 5 from IP address 192.108.250.1, TCP port 1996. The packet is 69 bytes in length. This packet is received after the Ring Exchange information was received and updated on both sides.

```
RSRB: DATA: 5/192.108.250.1/1996 Explorer trn 2, vrn 5, off 18, len 69
```

The following line indicates that a packet containing data came across virtual ring 5 from IP address 192.108.250.1 over TCP port 1996. The packet is being placed on the local target ring 2. The packet is 92 bytes in length.

```
RSRB: DATA: 5/192.108.250.1/1996 Forward trn 2, vrn 5, off 0, len 92
```

The following line indicates that a packet containing data is being forwarded to the peer that has IP address 192.108.250.1 address belonging to local ring 2 and bridge 1. The packet is forwarded via virtual ring 5. This packet is sent after the Ring Exchange information was received and updated on both sides.

```
RSRB: DATA: forward Forward srn 2, br 1, vrn 5 to peer 5/192.108.250.1/1996
```

The following is sample output from the **debug source bridge** command for peer bridges using direct encapsulation as a transport mechanism. The RSRB network configuration has ring 1 and ring 2 bridged together through peer bridges. The peer bridges are connected via a serial line and use TCP as the transport mechanism.

```
Router# debug source bridge
RSRB: remote explorer to 5/Serial1 srn 1 [C840.0011.0050.0000]
RSRB: Version/Ring XReq sent to peer 5/Serial1
RSRB: Received version reply from 5/Serial1 (version 2)
RSRB: IFin: 5/Serial1 Ring Xchg, Rep trn 0, vrn 5, off 0, len 10
RSRB: added bridge 1, ring 1 for 5/Serial1
```

The following line indicates that a remote explorer frame was sent to remote peer Serial1, which belongs to ring group 5. The explorer frame originated from ring 1. The RIF descriptor 0011.0050 was generated by the local station and indicates that the frame was sent out via bridge 1 onto virtual ring 5.

```
RSRB: remote explorer to 5/Serial1 srn 1 [C840.0011.0050.0000]
```

The following line indicates that a request for remote peer information was sent to Serial1. The bridge belongs to ring group 5.

```
RSRB: Version/Ring XReq sent to peer 5/Serial1
```

The following line is the response to the version request previously sent. The response is sent from Serial 1. The bridge belongs to ring group 5 and the version is 2.

```
RSRB: Received version reply from 5/Serial1 (version 2)
```

The following line is the response to the ring request previously sent. The response is sent from Serial1. The target ring number is 2, virtual ring number is 5, the offset is 0, and the length of the frame is 39 bytes.

```
RSRB: IFin: 5/Serial1 Ring Xchg Rep, trn 2, vrn 5, off 0, len 39
```

The following line indicates that bridge 1 and ring 1 were added to the source-bridge table for Serial1:

```
RSRB: added bridge 1, ring 1 for 5/Serial1
```

debug source error

To display source-route bridging (SRB) errors, use the **debug source error** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug source error

no debug source error

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

The debug source error command displays some output also found in the **debug source bridge** output. See the **debug source bridge** command for other possible output.

Examples

In all of the following examples of **debug source error** command messages, the variable *number* is the Token Ring interface. For example, if the line of output starts with SRB1, the output relates to the Token Ring 1 interface. SRB indicates a source-route bridging message. RSRB indicates a remote source-route bridging message. SRTLB indicates a source-route translational bridging (SR/TLB) message.

In the following example, a packet of protocol *protocol-type* was dropped:

```
SRB
number
 drop: Routed protocol
protocol-type
```

In the following example, an Address Resolution Protocol (ARP) packet was dropped. ARP is defined in RFC 826.

```
SRB
number
 drop:TYPE_RFC826_ARP
```

In the following example, the current Cisco IOS version does not support Qualified Logical Link Control (QLLC). Reconfigure the router with an image that has the IBM feature set.

```
RSRB: QLLC not supported in version version
Please reconfigure.
```

In the following example, the packet was dropped because the outgoing interface of the router was down:

```
RSRB IF: outgoing interface not up, dropping packet
```

In the following example, the router received an out-of-sequence IP sequence number in a Fast Sequenced Transport (FST) packet. FST has no recovery for this problem like TCP encapsulation does.

```
RSRB FST: bad sequence number dropping.
```

In the following example, the router was unable to locate the virtual interface:

```
RSRB: couldn't find virtual interface
```

In the following example, the TCP queue of the peer router is full. TCPD indicates that this is a TCP debug.

```
RSRB TCPD: tcp queue full for peer
```

In the following example, the router was unable to send data to the *peer* router. A *result* of 1 indicates that the TCP queue is full. A *result* of --1 indicates that the RSRB peer is closed.

```
RSRB TCPD: tcp send failed for peer result
```

In the following example, the routing information identifier (RII) was not set in the explorer packet going forward. The packet will not support SRB, so it is dropped.

```
vrforward_explorer - RII not set
```

In the following example, a packet sent to a virtual bridge in the router did not include a routing information field (RIF) to tell the router which route to use:

```
RSRB: no RIF on packet sent to virtual bridge
```

The following example indicates that the RIF did not contain any information or the length field was set to zero:

```
RSRB: RIF length of zero sent to virtual bridge
```

The following message occurs when the local service access point (LSAP) is out of range. The variable *lsap-out*is the value, *type* is the type of RSRB peer, and *state* is the state of the RSRB peer.

```
VRP: rsrb_lsap_out = lsap-out, type = type, state = state
```

In the following message, the router is unable to find another router with which to exchange bridge protocol data units (BPDUs). BPDUs are exchanged to set up the spanning tree and determine the forwarding path.

```
RSRB(span): BPDU's peer not found
```

Related Commands

Command	Description
debug source bridge	Displays information about packets and frames transferred across a source-route bridge.
debug source event	Displays information on SRB activity.

debug source event

To display information on source-route bridging (SRB) activity, use the **debug source event** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug source event

no debug source event

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

Some of the output from the **debug source bridge** and **debug source error** commands is identical to the output of this command.

Note In order to use the **debug source event** command to display traffic source-routed through an interface, you first must disable fast switching of SRB frames with the **no source bridge route-cache** interface configuration command.

Examples

The following is sample output from the **debug source event** command:

```
Router# debug source event
RSRB0: forward (srn 5 bn 1 trn 10), src: 8110.2222.33c1 dst: 1000.5a59.04f9
[0800.3201.00A1.0050]
RSRB0: forward (srn 5 bn 1 trn 10), src: 8110.2222.33c1 dst: 1000.5a59.04f9
[0800.3201.00A1.0050]
RSRB0: forward (srn 5 bn 1 trn 10), src: 8110.2222.33c1 dst: 1000.5a59.04f9
[0800.3201.00A1.0050]
RSRB0: forward (srn 5 bn 1 trn 10), src: 8110.2222.33c1 dst: 1000.5a59.04f9
[0800.3201.00A1.0050]
RSRB0: forward (srn 5 bn 1 trn 10), src: 8110.2222.33c1 dst: 1000.5a59.04f9
[0800.3201.00A1.0050]
```

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

Table 24: debug source event Field Descriptions

Field	Description
RSRB0:	Indication that this routing information field (RIF) cache entry is for the Token Ring interface 0, which has been configured for remote source-route bridging (SRB). (SRB1, in contrast, would indicate that this RIF cache entry is for Token Ring 1, configured for SRB.)

Field	Description
forward	Forward (normal data) packet, in contrast to a control packet containing proprietary Cisco bridging information.
srn 5	Ring number of the source ring of the packet.
bn 1	Bridge number of the bridge this packet traverses.
trn 10	Ring number of the target ring of the packet.
src: 8110.2222.33c1	Source address of the route in this RIF cache entry.
dst: 1000.5a59.04f9	Destination address of the route in this RIF cache entry.
[0800.3201.00A1.0050]	RIF string in this RIF cache entry.

In the following example messages, SRB*number* or RSRB*number*denotes a message associated with interface Token Ring *number*. A *number*of 99 denotes the remote side of the network.

```
SRB
number
: no path, s:
source-MAC-addr
d:
dst-MAC-addr
rif:
rif
```

In the preceding example, a bridgeable packet came in on interface Token Ring *number*but there was nowhere to send it. This is most likely a configuration error. For example, an interface has source bridging turned on, but it is not connected to another source bridging interface or a ring group.

In the following example, a bridgeable packet has been forwarded from Token Ring *number* to the target ring. The two interfaces are directly linked.

```
SRB
number
: direct forward (srn
ring
 bn
bridge
 trn
ring
)
```

In the following examples, a proxy explorer reply was not generated because the address could not be reached from this interface. The packet came from the node with the first *address*.

```
SRB
number
: br dropped proxy XID,
address
 for
address
, wrong vring (rem)
SRB
number
```

```
: br dropped proxy TEST,
address
 for
address
, wrong vring (rem)
SRB
number
: br dropped proxy XID,
address
 for
address
, wrong vring (local)
SRB
number
: br dropped proxy TEST,
address
 for
address
, wrong vring (local)
SRB
number
: br dropped proxy XID,
address
 for
address
, no path
SRB
number
: br dropped proxy TEST,
address
 for
address
, no path
```

In the following example, an appropriate proxy explorer reply was generated on behalf of the second *address*. It is sent to the first *address*.

```
SRB
number
: br sent proxy XID,
address
 for
address
[
rif
]
SRB
number
: br sent proxy TEST,
address
 for
address
[
rif
]
```

The following example indicates that the broadcast bits were not set, or that the routing information indicator on the packet was not set:

```
SRB
number
: illegal explorer, s:
source-MAC-addr
 d:
dst-MAC-addr
 rif:
rif
```

The following example indicates that the direction bit in the RIF field was set, or that an odd packet length was encountered. Such packets are dropped.

```
SRB
```

```
number
: bad explorer control, D set or odd
```

The following example indicates that a spanning explorer was dropped because the spanning option was not configured on the interface:

```
SRB
number
: span dropped, input off, s:
source-MAC-addr
 d:
dst-MAC-addr
rif:
rif
```

The following example indicates that a spanning explorer was dropped because it had traversed the ring previously:

```
SRB
number
: span violation, s:
source-MAC-addr
 d:
dst-MAC-addr
 rif:
rif
```

The following example indicates that an explorer was dropped because the maximum hop count limit was reached on that interface:

```
SRB
number
: max hops reached -
hop-cnt
, s:
source-MAC-addr
 d:
dst-MAC-addr
rif:
rif
```

The following example indicates that the ring exchange request was sent to the indicated peer. This request tells the remote side which rings this node has and asks for a reply indicating which rings that side has.

```
RSRB: sent RingXreq to
ring-group
/
ip-addr
```

The following example indicates that a message was sent to the remote peer. The *label* variable can be AHDR (active header), PHDR (passive header), HDR (normal header), or DATA (data exchange), and *op* can be Forward, Explorer, Ring Xchg, Req, Ring Xchg, Rep, Unknown Ring Group, Unknown Peer, or Unknown Target Ring.

```
RSRB:
label
: sent
op
 to
ring-group
/
ip-addr
```

The following example indicates that the remote bridge and ring pair were removed from or added to the local ring group table because the remote peer changed:

```
RSRB: removing bn
bridge
 rn
```

```
ring
 from
ring-group
/
ip-addr
RSRB: added bridge
bridge
, ring
ring
 for
ring-group
/
ip-addr
```

The following example shows miscellaneous remote peer connection establishment messages:

```
RSRB: peer
ring-group
/
ip-addr
 closed [last state
n
]
RSRB: passive open
ip-addr
(remote port) ->
local port
RSRB: CONN: opening peer
ring-group
/
ip-addr
, attempt
n
RSRB: CONN: Remote closed
ring-group
/
ip-addr
 on open
RSRB: CONN: peer
ring-group
/
ip-addr
 open failed,
reason
[
code
]
```

The following example shows that an explorer packet was propagated onto the local ring from the remote ring group:

```
RSRBn: sent local explorer, bridge
bridge
 trn
ring
, [
rif
]
```

The following messages indicate that the RSRB code found that the packet was in error:

```
RSRBn: ring group
ring-group
 not found
RSRBn: explorer rif [
rif
] not long enough
```

The following example indicates that a buffer could not be obtained for a ring exchange packet (this is an internal error):

```
RSRB: couldn't get pak for ringXchg
```

The following example indicates that a ring exchange packet was received that had an incorrect length (this is an internal error):

```
RSRB: XCHG: req/reply badly formed, length
pak-length
, peer
peer-id
```

The following example indicates that a ring entry was removed for the peer; the ring was possibly disconnected from the network, causing the remote router to send an update to all its peers.

```
RSRB: removing bridge
bridge
 ring
ring
 from
peer-id

ring-type
```

The following example indicates that a ring entry was added for the specified peer; the ring was possibly added to the network, causing the other router to send an update to all its peers.

```
RSRB: added bridge
bridge
, ring
ring
 for
peer-id
```

The following example indicates that no memory was available to add a ring number to the ring group specified (this is an internal error):

```
RSRB: no memory for ring element
ring-group
```

The following example indicates that memory was corrupted for a connection block (this is an internal error):

```
RSRB: CONN: corrupt connection block
```

The following example indicates that a connector process started, but that there was no packet to process (this is an internal error):

```
RSRB: CONN: warning, no initial packet, peer:
ip-addr peer-pointer
```

The following example indicates that a packet was received with a version number different from the one pre-sent on the router:

```
RSRB: IF New version. local=
local-version
, remote=
remote-version
,
pak-op-code

peer-id
```

The following example indicates that a packet with a bad op code was received for a direct encapsulation peer (this is an internal error):

```
RSRB: IFin: bad op
op-code
 (op code
string
) from
peer-id
```

The following example indicates that the virtual ring header will not fit on the packet to be sent to the peer (this is an internal error):

```
RSRB: vrif_sender, hdr won't fit
```

The following example indicates that the specified peer is being opened. The retry count specifies the number of times the opening operation is attempted.

```
RSRB: CONN: opening peer
peer-id

retry-count
```

The following example indicates that the router, configured for FST encapsulation, received a version reply to the version request packet it had sent previously:

```
RSRB: FST Rcvd version reply from
peer-id
 (version
version-number
)
```

The following example indicates that the router, configured for FST encapsulation, sent a version request packet to the specified peer:

```
RSRB: FST Version Request. op =
opcode
,
peer-id
```

The following example indicates that the router received a packet with a bad op code from the specified peer (this is an internal error):

```
RSRB: FSTin: bad op
opcode
 (op code
string
) from
peer-id
```

The following example indicates that the TCP connection between the router and the specified peer is being aborted:

```
RSRB: aborting
ring-group
/
peer-id
 (vrtcpd_abort called)
```

The following example indicates that an attempt to establish a TCP connection to a remote peer timed out:

```
RSRB: CONN: attempt timed out
```

The following example indicates that a packet was dropped because the ring group number in the packet did not correlate with the ring groups configured on the router:

```
RSRB
number
: ring group
ring-group
 not found
```

debug span

To display information on changes in the spanning-tree topology when debugging a transparent bridge, use the **debug span** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug span

no debug span

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

This command is useful for tracking and verifying that the spanning-tree protocol is operating correctly.

Examples

The following is sample output from the **debug span** command for an IEEE bridge protocol data unit (BPDU) packet:

```
Router# debug span
ST: Ether4 0000000000000A080002A02D6700000000000A080002A02D6780010000140002000F00
```

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

```
ST: Ether4 0000000000000A080002A02D6700000000000A080002A02D6780010000140002000F00
A   B C D E   F           G       H   I           J K L   M   N   O
```

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

Table 25: debug span Field Descriptions--IEEE BPDU Packet

Field	Description
ST:	Indication that this is a spanning tree packet.
Ether4	Interface receiving the packet.
(A) 0000	Indication that this is an IEEE BPDU packet.
(B) 00	Version.
(C) 00	Command mode: • 00 indicates config BPDU. • 80 indicates the Topology Change Notification (TCN) BPDU.

Field	Description
(D) 00	Topology change acknowledgment: • 00 indicates no change. • 80 indicates a change notification.
(E) 000A	Root priority.
(F) 080002A02D67	Root ID.
(G) 00000000	Root path cost (0 means the sender of this BPDU packet is the root bridge).
(H) 000A	Bridge priority.
(I) 080002A02D67	Bridge ID.
(J) 80	Port priority.
(K) 01	Port Number 1.
(L) 0000	Message age in 256ths of a second (0 seconds, in this case).
(M) 1400	Maximum age in 256ths of a second (20 seconds, in this case).
(N) 0200	Hello time in 256ths of a second (2 seconds, in this case).
(O) 0F00	Forward delay in 256ths of a second (15 seconds, in this case).

The following is sample output from the **debug span** command for a DEC BPDU packet:

```
Router# debug span
ST: Ethernet4 E1190100000200000C01A2C90064008000000C0106CE0A01050F1E6A
```

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

```
E1 19 01 00 0002 00000C01A2C9 0064 0080 00000C0106CE 0A 01 05 0F 1E 6A
A  B  C  D  E    F            G    H    I            J  K  L  M  N  O
```

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

Table 26: debug span Field Descriptions for a DEC BPDU Packet

Field	Description
ST:	Indication that this is a spanning tree packet.

Field	Description
Ethernet4	Interface receiving the packet.
(A) E1	Indication that this is a DEC BPDU packet.
(B) 19	Indication that this is a DEC hello packet. Possible values are as follows: • 0x19--DEC Hello • 0x02--TCN
(C) 01	DEC version.
(D) 00	Flag that is a bit field with the following mapping: • 1--TCN • 2--TCN acknowledgment • 8--Use short timers
(E) 0002	Root priority.
(F) 00000C01A2C9	Root ID (MAC address).
(G) 0064	Root path cost (translated as 100 in decimal notation).
(H) 0080	Bridge priority.
(I) 00000C0106CE	Bridge ID.
(J) 0A	Port ID (in contrast to interface number).
(K) 01	Message age (in seconds).
(L) 05	Hello time (in seconds).
(M) 0F	Maximum age (in seconds).
(N) 1E	Forward delay (in seconds).
(O) 6A	Not applicable.

debug spanning-tree

To debug spanning-tree activities, use the **debug spanning-tree** command in **privileged EXEC**mode. To disable debugging output, use the **no** form of this command.

debug spanning-tree {**all**| **backbonefast**| **bpdu**| **bpdu-opt**| **config**| **etherchannel**| **events**| **exceptions**| **general**| **pvst+**| **root**| **snmp**| **uplinkfast**}

no debug spanning-tree {**all**| **backbonefast**| **bpdu**| **bpdu-opt**| **config**| **etherchannel**| **events**| **exceptions**| **general**| **pvst+**| **root**| **snmp**| **uplinkfast**}

Syntax Description

all	Displays all spanning-tree debugging messages.
backbonefast	Displays debugging messages for BackboneFast events.
bpdu	Displays debugging messages for spanning-tree Bridge Protocol Data Units (BPDUs).
bpdu-opt	Displays debugging messages for optimized BPDU handling.
config	Displays debugging messages for spanning-tree configuration changes.
etherchannel	Displays debugging messages for EtherChannel support.
events	Displays debugging messages for spanning-tree topology events.
exceptions	Displays debugging messages for spanning-tree exceptions.
general	Displays debugging messages for general spanning-tree activity.
pvst+	Displays debugging messages for per-VLAN Spanning Tree Plus (PVST+) events.
root	Displays debugging messages for spanning-tree root events.
snmp	Displays debugging messages for spanning-tree Simple Network Management Protocol (SNMP) handling.
uplinkfast	Displays debugging messages for UplinkFast events.

Command Default

Debugging is disabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.1(6)EA2	This command was introduced.
12.2(15)ZJ	This command was implemented on the following platforms: Cisco 2600 series, Cisco 3600 series, and Cisco 3700 series routers.
12.3(4)T	This command was integrated into Cisco IOS Release 12.3(4)T on the following platforms: Cisco 2600 series, Cisco 3600 series, and Cisco 3700 series routers.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.

Usage Guidelines

This command is supported only by the Supervisor Engine and can be entered only from the switch console.

The **undebug spanning-tree** command is the same as the **no debug spanning-tree** command.

Related Commands

Command	Description
show debugging	Displays information about the types of debugging that are enabled.
show spanning-tree	Displays spanning-tree state information.

debug ss7 mtp1

Note Use this command only if told to do so by your Cisco representative.

To initiate Signaling System 7 (SS7) Message Transfer Part Level 1 (MTP1) debugging, enter the **debug ss7 mtp1** command in global configuration mode during a low-traffic period. To disable debugging output, use the **no** form of this command.

debug ss7 mtp1 [**mtp2**| **ipc**| **link-state**| **oir**| **rx**| **scc-regs**| **siram**| **tdm-info**| **tx**]

no debug ss7 mtp1

Syntax Description

mtp2	(Optional) Initiates SS7 MTP2 debugging.
ipc	(Optional) Initiates SS7 MTP1 debugging for HOST/FW IPC.
link-state	(Optional) Initiates SS7 MTP1 debugging for link-state transitions.
oir	(Optional) Initiates SS7 MTP1 trunk dial feature card (DFC) online insertion and removal (OIR) debugging.
rx	(Optional) Initiates SS7 MTP1 debugging for receive events. Not used in Release 12.2(11)T.
scc-regs	(Optional) Initiates SS7 MTP1 debugging for SCC registers. Not used in Release 12.2(11)T.
siram	(Optional) Initiates SS7 MTP1 debugging for siram values. Not used in Release 12.2(11)T.
tdm-info	(Optional) Initiates SS7 MTP1 debugging for time-division multiplexing (TDM) information.
tx	(Optional) Initiates SS7 MTP1 debugging for transmission events. Not used in Release 12.2(11)T.

Command Default

Debug is disabled.

Command Modes

Global configuration

Command History

Release	Modification
12.2(11)T	This command was introduced on the Cisco AS5350 and Cisco AS5400 Signaling Link Terminal (SLT).

Usage Guidelines

The following debug commands are not used in this release:

- **debug ss7 mtp1 rx**
- **debug ss7 mtp1 tx**
- **debug ss7 mtp1 scc-regs**
- **debug ss7 mtp1 siram**

Examples

To turn on message tracing between the host processor and the trunk firmware for each trunk card inserted, use the **debug ss7 mtp1 ipc** command.

For example, there is a digital link in slot 7, trunk 0, channel-group 0 (therefore, timeslot 1). When you enter **show ss7 mtp1 links**, the following output is displayed:

```
Router# show ss7 mtp1 links
SS7 MTP1 Links [num = 1, platform max = 4]:
                       session
interface   type    SCC     state    channel
---------   -----   ---     -------  -------
7/0:0       digital 7/3     STOPPED     0
```

Notice that the link is stopped in this example. Enter the following commands:

```
Router# debug ss7 mtp1 ipc
Router# configure terminal
Router(config)# interface serial 7/0:0
Router(config-if)# no shutdown
Router(config-if)# end
```

You would see trace output similar to the following:

```
00:01:27:from Trunk(7):TRUNK_SERIAL_STOP(3), link_type=2
00:01:27:from Trunk(7):TRUNK_SERIAL_START(3), link_type=2
```

In this case, the output means that for the SS7 link that is using SCC3 on the trunk card in slot 7 (link 7/0:0), the host processor has told the board firmware to STOP then START.

To show low-level (MTP1) state changes for the internal state-machine implemented for each SS7 link, use the **debug ss7 mtp1 link-state** command. The following output shows the different MTP1 states link Serial 7/0:0 goes through during shutdown, no shutdown, and debug.

For example, if you stopped the SS7 link 7/0:0 (shutdown), then restarted it (no shutdown), you could see MTP1 state changes by enabling debugging, as follows:

```
Router# debug ss7 mtp1 link-state
Router# configure terminal
Router(config)# interface serial 7/0:0
Router(config-if)# shutdown
01:02:20:%TRUNK_SERIAL-3-STATE_GENERIC:
At ../src-7k-as5400/as5400_ss7_link.c:511 [Serial7/0:0]:STOP:
```

```
STARTED -> STOP_PENDING
ss7_link_ll_stop 7/0:0:Tx shadow ring has
0 unsent buffers
01:02:20:%TRUNK_SERIAL-3-STATE_GENERIC:
At ../src-7k-as5400/as5400_ss7_link.c:1010 [Serial7/0:0]: FW_STOPPED:
STOP_PENDING -> STOPPED
```

Now restart the link:

```
Router(config-if)# no shutdown
01:02:26:ss7_link_start:slot=7/SCCport=3 current state is STOPPED
01:02:26:%TRUNK_SERIAL-3-STATE_GENERIC:
At ../src-7k-as5400/as5400_ss7_link.c:1417 [Serial7/0:0]: START:
STOPPED -> START_PENDING
01:02:26:%TRUNK_SERIAL-3-STATE_GENERIC:
At ../src-7k-as5400/as5400_ss7_link.c:1164 [Serial7/0:0]: STOP_START:
START_PENDING -> STOP_START_PENDING
ss7_link_ll_stop 7/0:0:Tx shadow ring has 0 unsent buffers
01:02:26:%TRUNK_SERIAL-3-STATE_GENERIC:
At ../src-7k-as5400/as5400_ss7_link.c:1010 [Serial7/0:0]: FW_STOPPED:
STOP_START_PENDING -> START_PENDING
01:02:26:%TRUNK_SERIAL-3-STATE_GENERIC:
At ../src-7k-as5400/as5400_ss7_link.c:1234 [Serial7/0:0]: FW_STARTED:
START_PENDING -> STARTED
```

To show detailed information about how TDM timeslots on the DFC trunk card on the host backplane are allocated and deallocated based on link configuration activity, use the **debug ss7 mtp1 tdm-info** command.

For example, if you wanted to create a digital SS7 link on timeslot 1 of trunk 0 for an 8PRI board in slot 7, and you would like to see traces of the TDM resources allocated, you would enable TDM debugging using the **debug ss7 mtp1 tdm-info** command then create the new SS7 link as described above, as in the following example:

```
Router# debug ss7 mtp1 tdm-info
Router# configure terminal
Router(config)# controller t1 7/0
Router(config-controller)# channel-group 0 timeslots 1
Router(config-controller)# exit
Router(config)# interface serial 7/0:0
Router(config-if)# encapsulation ss7
```

Due to the debug flag, the following information is displayed:

```
05:26:55: ss7_link_flink_tdm_setup:card type for slot 7 is T1 8PRI
05:26:55: ds0-side BEFORE call to tdm_allocate_bp_ts()
   slot    = 7
   unit    = 0     (trunk)
   channel = 4
   stream  = 0
   group   = 0
05:26:55: scc-side BEFORE call to tdm_allocate_bp_ts()
   slot    = 7
   unit    = 29
   channel = 3     (SCC-port)
   stream  = 3
   group   = 0
05:26:55:
05:26:55:TDM(PRI:0x28002000):Close PRI framer st0 ch4
05:26:55:<<<   tdm_allocate_bp_ts(ss7_ch) SUCCEEDED   >>>
05:26:55:scc-side AFTER call to tdm_allocate_bp_ts()
   bp_channel = 4
   bp_stream  = 0
   bp_ts->bp_stream    = 0
   bp_ts->bp_channel   = 4
   bp_ts->vdev_slot    = 7
   bp_ts->vdev_channel = 3
```

bp_ts->vdev_slot = 7 should be same as the CLI slot, and bp_ts->vdev_channel = 3should be *->channel.

When you later remove the SS7 link, other information is displayed showing how resources are cleaned up.

Related Commands

Command	Description
debug ss7 sm	Displays debugging messages for an SS7 Session Manager.

debug ss7 mtp2

To trace backhaul Signaling System 7 (SS7) Message Transfer Part Level 2 (MTP2) message signaling units (MSUs), enter the **debug ss7 mtp2** command in global configuration mode during a low-traffic period. To disable debugging output, use the **no** form of this command.

debug ss7 mtp2 [**aerm| backhaul| cong| iac| lsc| lssu| msu| packet [all]| rcv| suerm| timer| txc**] [*channel*]

no debug ss7 mtp2

Syntax Description

aerm	(Optional) Initiates alignment Error Rate Monitor events.
backhaul	(Optional) Initiates trace backhaul control messages. The *channel*argument represents a logical channel number. Valid values are from 0 to 3.
cong	(Optional) Initiates congestion Control events.
iac	(Optional) Initiates initial Alignment Control events.
lsc	(Optional) Initiates Link State Control events.
lssu	(Optional) Initiates trace backhaul LSSU messages.
msu	(Optional) Initiates trace backhaul MSU messages (use during low traffic only).
packet [**all**]	(Optional) Initiates low-level MTP2 packet tracing. If you do not specify a channel number or enter the **all** keyword, the command displays information for channel 0.
rcv	(Optional) Displays information about SS7 MTP2 receiver state machine events and transitions.
suerm	(Optional) Displays information about SS7 MTP2 Signal Unit Error Rate Monitor (SUERM) state machine events and transitions.
timer	(Optional) Displays information about SS7 MTP2 timer starts and stops.
txc	(Optional) Displays information about SS7 MTP2 transmit state machine events and transitions.
channel	(Optional) The channelargument represents a logical channel number. Valid values are from 0 to 3.

Command Default

Debug is disabled.

Command Modes

Global configuration

Command History

Release	Modification
12.0(7)XR	This command was introduced.
12.1(1)T	This command was integrated into Cisco IOS Release 12.1(1)T.
12.2(11)T	This command was implemented on the Cisco AS5350 and Cisco AS5400 Cisco Signaling Link Terminal (SLT).

Usage Guidelines

If you do not specify a channel number with each keyword, the command displays information for channel 0.

Examples

The following is sample output from the **debug ss7 mtp2 aerm** command. See the MTP2 specification tables for details:

```
Router# debug ss7 mtp2 aerm 0
*Mar  8 08:59:30.991:itu2AERM_Start  chnl=0  MTP2AERM_IDLE
*Mar  8 08:59:35.070:itu2AERM_Stop  chnl=0  MTP2AERM_MONITORING
```

The following is an example of **debug ss7 mtp2 backhaul** command output for channel 0:

```
Router# debug ss7 mtp2 backhaul 0
*Mar  1 03:08:04.433: MTP2: send Disc Ind  ch=0  reason=0x14-T2 expired waiting for SIO
*Mar  1 03:08:04.433: MTP2: send LSC Ind  ch=0  event=0x8-lost link alignment cause=0x0
*Mar  1 03:08:08.721: MTP2: rcvd Conn Req - Normal  ch=0
*Mar  1 03:08:10.311: MTP2: rcvd Statistics Req-Send&Reset   ch=0
*Mar  1 03:08:10.311: MTP2: send Stats Cfm  ch=0
*Mar  1 03:08:20.440: MTP2: send Disc Ind  ch=0  reason=0x14-T2 expired waiting for SIO
*Mar  1 03:08:20.444: MTP2: send LSC Ind  ch=0  event=0x8-lost link alignment cause=0x0
*Mar  1 03:08:24.719: MTP2: rcvd Conn Req - Normal  ch=0
*Mar  1 03:08:36.438: MTP2: send Disc Ind  ch=0  reason=0x14-T2 expired waiting for SIO
*Mar  1 03:08:36.438: MTP2: send LSC Ind  ch=0  event=0x8-lost link alignment cause=0x0
*Mar  1 03:08:40.312: MTP2: rcvd Statistics Req-Send&Reset   ch=0
*Mar  1 03:08:40.312: MTP2: send Stats Cfm  ch=0
*Mar  1 03:08:40.721: MTP2: rcvd Conn Req - Normal  ch=0
*Mar  1 03:08:52.444: MTP2: send Disc Ind  ch=0  reason=0x14-T2 expired waiting for SIO
*Mar  1 03:08:52.444: MTP2: send LSC Ind  ch=0  event=0x8-lost link alignment cause=0x0
*Mar  1 03:08:56.719: MTP2: rcvd Conn Req - Normal  ch=0
*Mar  1 03:09:08.438: MTP2: send Disc Ind  ch=0  reason=0x14-T2 expired waiting for SIO
*Mar  1 03:09:08.438: MTP2: send LSC Ind  ch=0  event=0x8-lost link alignment cause=0x0
```

The following is an example of **debug ss7 mtp2 cong** command output. See the MTP2 specification tables for details:

```
Router# debug ss7 mtp2 cong 0
*Mar  8 09:10:56.219:itu2CongestionOnset  chnl=0  MTP2CONGESTION_IDLE
*Mar  8 09:10:59.332:itu2CongestionAbatement chnl=0
MTP2CONGESTION_ACTIVE
*Mar  8 09:11:01.143:itu2CongestionAbatement chnl=0  MTP2CONGESTION_IDLE
```

The following is an example of **debug ss7 mtp2 iac** command output. See the MTP2 specification tables for details:

```
Router# debug ss7 mtp2 iac 0
*Mar  8 09:17:58.367:itu2IAC_Start  chnl=0  MTP2IAC_IDLE
*Mar  8 09:17:58.739:itu2IAC_Rcvd_SIO  chnl=0  MTP2IAC_NOT_ALIGNED
*Mar  8 09:17:58.739:itu2IAC_Rcvd_SIN  chnl=0  MTP2IAC_ALIGNED
*Mar  8 09:17:58.739:itu2IAC_Rcvd_SIN  chnl=0  MTP2IAC_PROVING
*Mar  8 09:18:02.814:itu2IAC_T4_TMO   chnl=0  MTP2IAC_PROVING
```

The following is an example of **debug ss7 mtp2 lsc** command output. See the MTP2 specification tables for details:

```
Router# debug ss7 mtp2 lsc 0
*Mar  8 09:20:21.105:itu2LSC_Rcvd_SIOS  chnl=0  MTP2LSC_INSERVICE
*Mar  8 09:20:21.121:itu2LSC_Retrieve_BSNT  chnl=0  MTP2LSC_OOS
*Mar  8 09:20:22.058:itu2LSC_SetEmergency  chnl=0  MTP2LSC_OOS
*Mar  8 09:20:22.058:itu2LSC_Start  chnl=0  MTP2LSC_OOS
*Mar  8 09:20:33.785:itu2LSC_AlignmentNotPossible  chnl=0
MTP2LSC_INITIAL_ALIGNMENT
*Mar  8 09:20:38.758:itu2LSC_SetEmergency  chnl=0  MTP2LSC_OOS
*Mar  8 09:20:38.758:itu2LSC_Start  chnl=0  MTP2LSC_OOS
*Mar  8 09:20:44.315:itu2LSC_Rcvd_SIO  chnl=0  MTP2LSC_INITIAL_ALIGNMENT
*Mar  8 09:20:44.315:itu2LSC_Rcvd_SIO  chnl=0  MTP2LSC_INITIAL_ALIGNMENT
*Mar  8 09:20:44.319:itu2LSC_Rcvd_SIE  chnl=0  MTP2LSC_INITIAL_ALIGNMENT
*Mar  8 09:20:44.319:itu2LSC_Rcvd_SIE  chnl=0  MTP2LSC_INITIAL_ALIGNMENT
*Mar  8 09:20:48.397:itu2LSC_AlignmentComplete  chnl=0
MTP2LSC_INITIAL_ALIGNMENT
```

The following is an example of **debug ss7 mtp2 msu** command output for channel 2. The output for this command can slow traffic under busy conditions, so enter it when there is low traffic. See the MTP2 specification tables for details about the command output:

```
Router# debug ss7 mtp2 msu 2
*Mar  1 01:01:12.447: MTP2: send MSU Ind  ch=2  len=25
*Mar  1 01:01:12.455: MTP2: rcvd MSU Req  ch=2  len=252
```

Caution Use this command only for testing problems in a controlled environment. This command can generate significant amounts of output. If there is any significant amount of traffic flow when you issue the command, the processor may slow down so much that RUDP connections fail. This command is recommended for field support personnel only, and is not recommended for use without prior recommendation from Cisco.

The following is an example of **debug ss7 mtp2 packet** command output for channel 0:

```
Router# debug ss7 mtp2 packet 0
*Mar  1 00:53:00.052: MTP2 incoming trace enabled on channel 0.
*Mar  1 00:53:00.052: MTP2 outgoing trace enabled on channel 0.
*Mar  1 00:53:07.220: ---- Incoming Rudp msg (20 bytes) ----
SM_msg_type    0x00008000
protocol_type  0x0001
msg_ID         0x0001
msg_type       0x0044
channel_ID     0x0000
bearer_ID      0x0000
length         0x0004
data           0x00000001

*Mar  1 00:53:07.224: ---- Outgoing Rudp msg (132 bytes) ----
SM_msg_type    0x00008000
protocol_type  0x0001
msg_ID         0x0001
msg_type       0x0045
channel_ID     0x0000
bearer_ID      0x0000
length         0x0074
```

```
data           0x0000001E 0x00000000 0x00000000 0x00000000
               0x00000000 0x00000000 0x00000000 0x00000000
               0x00000000 0x00000000 0x00000000 0x00000000
               0x00000002 0x00000000 0x00008317 0x00000000
               0x00000002 0x00000000 0x00000008 0x009B5C97
               0x00000000 0x0032A2A7 0x0000061C 0x000000BF
               0x00000000 0x00000000 0x00000006 0x00000000
               0x000000ED

*Mar  1 00:53:11.343: ---- Outgoing Rudp msg (41 bytes) ----
SM_msg_type    0x00008000
protocol_type  0x0001
msg_ID         0x0000
msg_type       0x0011
channel_ID     0x0000
bearer_ID      0x0000
length         0x0019
data           0x8201190A 0x03190A00 0x11F01122 0x33445566
               0x778899AA 0xBBCCDDEE

*Mar  1 00:53:11.351: ---- Incoming Rudp msg (41 bytes) ----
SM_msg_type    0x00008000
protocol_type  0x0001
msg_ID         0x0001
msg_type       0x0010
channel_ID     0x0000
bearer_ID      0x0000
length         0x0019
data           0xB203190A 0x01190A00 0x21F01122 0x33445566
               0x778899AA 0xBBCCDDEE
*Mar  1 00:53:13.739: ---- Incoming Rudp msg (27 bytes) ----
SM_msg_type    0x00008000
protocol_type  0x0001
msg_ID         0x0001
msg_type       0x0010
channel_ID     0x0000
bearer_ID      0x0000
length         0x000B
data           0x9503190A 0x01190A00
```

The following is an example of **debug ss7 mtp2 rcv** command output. See the MTP2 specification tables for details:

```
Router# debug ss7 mtp2 rcv 0
*Mar  8 09:22:35.160:itu2RC_Stop  chnl=0  MTP2RC_INSERVICE
*Mar  8 09:22:35.164:itu2RC_Start  chnl=0  MTP2RC_IDLE
*Mar  8 09:22:52.565:BSNR not in window
        bsnr=2  bibr=0x80    fsnr=66  fibr=0x80  fsnf=0  fsnl=127  fsnx=0
        fsnt=127
*Mar  8 09:22:52.569:BSNR not in window
        bsnr=2  bibr=0x80    fsnr=66  fibr=0x80  fsnf=0  fsnl=127  fsnx=0
        fsnt=127
*Mar  8 09:22:52.569:AbnormalBSN_flag == TRUE
*Mar  8 09:22:52.569:itu2RC_Stop  chnl=0  MTP2RC_INSERVICE
*Mar  8 09:22:57.561:itu2RC_Start  chnl=0  MTP2RC_IDLE
```

The following is an example of **debug ss7 mtp2 suerm** command output. See the MTP2 specification tables for details:

```
Router# debug ss7 mtp2 suerm 0
*Mar  8 09:33:51.108:itu2SUERM_Stop  chnl=0  MTP2SUERM_MONITORING
*Mar  8 09:34:00.155:itu2SUERM_Start  chnl=0  MTP2SUERM_IDLE
```

Caution Use this command only for testing problems in a controlled environment. This command can generate significant amounts of output. If there is any significant amount of traffic flow when you issue the command, the processor may slow down so much that RUDP connections fail. This command is recommended for field support personnel only, and is not recommended for use without prior recommendation from Cisco.

The following is an example of **debug ss7 mtp2 timer** command output for channel 0:

```
Router# debug ss7 mtp2 timer 0
*Mar  1 01:08:13.738: Timer T7 (ex delay) Start    chnl=0
*Mar  1 01:08:13.762: Timer T7 (ex delay) Stop     chnl=0
*Mar  1 01:08:13.786: Timer T7 (ex delay) Start    chnl=0
*Mar  1 01:08:13.810: Timer T7 (ex delay) Stop     chnl=0
*Mar  1 01:08:43.819: Timer T7 (ex delay) Start    chnl=0
*Mar  1 01:08:43.843: Timer T7 (ex delay) Stop     chnl=0
*Mar  1 01:08:48.603: Timer T7 (ex delay) Start    chnl=0
*Mar  1 01:08:48.627: Timer T7 (ex delay) Stop     chnl=0
*Mar  1 01:09:13.784: Timer T7 (ex delay) Start    chnl=0
*Mar  1 01:09:13.808: Timer T7 (ex delay) Stop     chnl=0
*Mar  1 01:09:13.885: Timer T7 (ex delay) Start    chnl=0
*Mar  1 01:09:13.909: Timer T7 (ex delay) Stop     chnl=0
```

Caution Use this command only for testing problems in a controlled environment. This command can generate significant amounts of output. If there is any significant amount of traffic flow when you issue the command, the processor may slow down so much that RUDP connections fail. This command is recommended for field support personnel only, and is not recommended for use without prior recommendation from Cisco.

The following is an example of **debug ss7 mtp2 txc** command output for channel 2. The transmission control is functioning and updating backward sequence numbers (BSNs). See the MTP2 specification for details:

```
Router# debug ss7 mtp2 txc 2
*Mar  1 01:10:13.831: itu2TXC_bsn_update  chnl=2  MTP2TXC_INSERVICE
*Mar  1 01:10:13.831: itu2TXC_bsn_update  chnl=2  MTP2TXC_INSERVICE
*Mar  1 01:10:13.831: itu2TXC_bsn_update  chnl=2  MTP2TXC_INSERVICE
*Mar  1 01:10:13.839: itu2TXC_PDU2xmit  chnl=2  MTP2TXC_INSERVICE
*Mar  1 01:10:13.863: itu2TXC_bsn_update  chnl=2  MTP2TXC_INSERVICE
*Mar  1 01:10:13.863: itu2TXC_bsn_update  chnl=2  MTP2TXC_INSERVICE
*Mar  1 01:10:23.603: itu2TXC_PDU2xmit  chnl=2  MTP2TXC_INSERVICE
*Mar  1 01:10:23.627: itu2TXC_bsn_update  chnl=2  MTP2TXC_INSERVICE
*Mar  1 01:10:23.627: itu2TXC_bsn_update  chnl=2  MTP2TXC_INSERVICE
*Mar  1 01:10:23.631: itu2TXC_bsn_update  chnl=2  MTP2TXC_INSERVICE
*Mar  1 01:10:23.631: itu2TXC_bsn_update  chnl=2  MTP2TXC_INSERVICE
*Mar  1 01:10:23.635: itu2TXC_bsn_update  chnl=2  MTP2TXC_INSERVICE
*Mar  1 01:10:43.900: itu2TXC_bsn_update  chnl=2  MTP2TXC_INSERVICE
*Mar  1 01:10:43.900: itu2TXC_bsn_update  chnl=2  MTP2TXC_INSERVICE
*Mar  1 01:10:43.900: itu2TXC_bsn_update  chnl=2  MTP2TXC_INSERVICE
*Mar  1 01:10:43.908: itu2TXC_PDU2xmit  chnl=2  MTP2TXC_INSERVICE
*Mar  1 01:10:43.928: itu2TXC_bsn_update  chnl=2  MTP2TXC_INSERVICE
*Mar  1 01:10:43.932: itu2TXC_bsn_update  chnl=2  MTP2TXC_INSERVIC
```

The following MTP2 specification tables explain codes that appear in the command output.

Backhaul Debug Event Codes	Description
0x0	Local processor outage
0x1	Local processor outage recovered
0x2	Entered a congested state
0x3	Exited a congested state
0x4	Physical layer up
0x5	Physical layer down
0x7	Protocol error (see cause code)

Backhaul Debug Event Codes	Description
0x8	Link alignment lost
0x9	Retransmit buffer full
0xa	Retransmit buffer no longer full
0xc	Remote entered congestion
0xd	Remote exited congestion
0xe	Remote entered processor outage
0xf	Remote exited processor outage

Backhaul Debug Cause Codes	Description
0x0	Cause unknown--default
0x1	Management initiated
0x2	Abnormal BSN (backward sequence number)
0x3	Abnormal FIB (Forward Indicator Bit)
0x4	Congestion discard

Backhaul Debug Reason Codes	Description
0x0	Layer management request
0x1	SUERM (Signal Unit Error Monitor) failure
0x2	Excessively long alignment period
0x3	T7 timer expired
0x4	Physical interface failure
0x5	Two or three invalid BSNs
0x6	Two or three invalid FIBs
0x7	LSSU (Link Status Signal Unit) condition
0x13	SIOs (Service Information Octets) received in Link State Control (LSC)

Backhaul Debug Reason Codes	Description
0x14	Timer T2 expired waiting for SIO
0x15	Timer T3 expired waiting for SIE/SIN
0x16	SIO received in initial alignment control (IAC)
0x17	Proving period failure
0x18	Timer T1 expired waiting for FISU (Fill-In Signal Unit)
0x19	SIN received in the in-service state
0x20	CTS lost
0x25	No resources

Related Commands

Command	Description
debug ss7 sm	Displays debugging messages for an SS7 Session Manager.

debug ss7 sm

To display debugging messages for an Signaling System 7 (SS7) Session Manager, use the **debug ss7 sm**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ss7 sm [**session** *session-id*| **set**| **timer**]

no debug ss7 sm session

Syntax Description

session	(Optional) Sets Session Manager session debug.
session-id	(Optional) Specifies a session ID number from 0 to 3.
set	(Optional) Sets Session Manager debug.
timer	(Optional) Sets Session Manager timer debug.

Command Default

Debug is disabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(7)XR and 12.1(1)T	This command was introduced.
12.1(1)T	This command was integrated into Cisco IOS Release 12.1(1)T.
12.2(11)T	This command replaces the **debug ss7 sm session** command. This command was modified with the **session**, **set**, and **timer** keywords. This command was also modified to support up to four Session Manager sessions.

Usage Guidelines

Use this command to watch the Session Manager and Reliable User Data Protocol (RUDP) sessions. The Session Manager is responsible for establishing the RUDP connectivity to the Virtual Switch Controller (VSC).

Support for up to four Session Manager sessions was added. Session Manager sessions are now numbered 0 to 3. This feature changes the CLI syntax, and adds sessions 2 and 3.

Examples

The following is an example of **debug ss7 sm** command output using the **session** keyword. The Session Manager has established the connection (RUDP_CONN_OPEN_SIG) for session 3.

```
Router# debug ss7 sm session 3
*Mar  8 09:37:52.119:SM:rudp signal RUDP_SOFT_RESET_SIG, session = 3
*Mar  8 09:37:58.129:SM:rudp signal RUDP_CONN_RESET_SIG, session = 3
*Mar  8 09:37:58.129:SM:Opening session[0] to 10.5.0.4:8060
*Mar  8 09:37:58.137:SM:rudp signal RUDP_CONN_OPEN_SIG, session = 3
```

The following is an example of **debug ss7 sm session** command output for session 0. The Session Manager has established the connection (RUDP_CONN_OPEN_SIG):

```
Router# debug ss7 sm session 0
*Mar  8 09:37:52.119:SM:rudp signal RUDP_SOFT_RESET_SIG, session = 0
*Mar  8 09:37:58.129:SM:rudp signal RUDP_CONN_RESET_SIG, session = 0
*Mar  8 09:37:58.129:SM:Opening session[0] to 10.5.0.4:8060
*Mar  8 09:37:58.137:SM:rudp signal RUDP_CONN_OPEN_SIG, session = 0
```

Related Commands

Command	Description
encapsulation ss7	Assigns a channel group and selects the DS0 time slots desired for SS7 links.

debug sse

To display information for the silicon switching engine (SSE) processor, use the **debug sse** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug sse

no debug sse

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

Use the **debug sse** command to display statistics and counters maintained by the SSE.

Examples

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

```
Router# debug sse
SSE: IP number of cache entries changed 273 274
SSE: bridging enabled
SSE: interface Ethernet0/0 icb 0x30 addr 0x29 status 0x21A040 protos 0x11
SSE: interface Ethernet0/1 icb 0x33 addr 0x29 status 0x21A040 protos 0x11
SSE: interface Ethernet0/2 icb 0x36 addr 0x29 status 0x21A040 protos 0x10
SSE: interface Ethernet0/3 icb 0x39 addr 0x29 status 0x21A040 protos 0x11
SSE: interface Ethernet0/4 icb 0x3C addr 0x29 status 0x21A040 protos 0x10
SSE: interface Ethernet0/5 icb 0x3F addr 0x29 status 0x21A040 protos 0x11
SSE: interface Hssi1/0 icb 0x48 addr 0x122 status 0x421E080 protos 0x11
SSE: cache update took 316ms, elapsed 320ms
```

The following line indicates that the SSE cache is being updated due to a change in the IP fast-switching cache:

```
SSE: IP number of cache entries changed 273 274
```

The following line indicates that bridging functions were enabled on the SSE:

```
SSE: bridging enabled
```

The following lines indicate that the SSE is now loaded with information about the interfaces:

```
SSE: interface Ethernet0/0 icb 0x30 addr 0x29 status 0x21A040 protos 0x11
SSE: interface Ethernet0/1 icb 0x33 addr 0x29 status 0x21A040 protos 0x11
SSE: interface Ethernet0/2 icb 0x36 addr 0x29 status 0x21A040 protos 0x10
SSE: interface Ethernet0/3 icb 0x39 addr 0x29 status 0x21A040 protos 0x11
SSE: interface Ethernet0/4 icb 0x3C addr 0x29 status 0x21A040 protos 0x10
SSE: interface Ethernet0/5 icb 0x3F addr 0x29 status 0x21A040 protos 0x11
SSE: interface Hssi1/0 icb 0x48 addr 0x122 status 0x421E080 protos 0x11
```

The following line indicates that the SSE took 316 ms of processor time to update the SSE cache. The value of 320 ms represents the total time elapsed while the cache updates were performed.

```
SSE: cache update took 316ms, elapsed 320ms
```

debug ssg ctrl-errors

To display all error messages for control modules, use the **debug ssg ctrl-errors** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ssg ctrl-errors

no debug ssg ctrl-errors

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(3)DC	This command was introduced on the Cisco 6400 node route processor.
12.2(4)B	This command was integrated into Cisco IOS Release 12.2(4)B.
12.2(8)T	This command was integrated into Cisco IOS Release 12.2(8)T.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

Use this command to show error messages for the control modules. These modules include all those that manage the user authentication and service login and logout (RADIUS, PPP, Subblock, and Accounting). An error message is the result of an error detected during normal execution.

Examples

The following output is generated by using the **debug ssg ctrl-errors** command when a host logs in to and logs out of a service:

```
Router# debug ssg ctrl-errors
Mar 29 13:51:30 [192.168.5.1.15.21] 59:00:15:38:%VPDN-6-AUTHORERR:L2F NAS
LowSlot6 cannot locate a AAA server for Vi6 user User1
Mar 29 13:51:31 [192.168.5.1.15.21] 60:00:15:39:%LINEPROTO-5-UPDOWN:Line
protocol on Interface Virtual-Access6, changed state to down
```

Related Commands

Command	Description
debug ssg ctrl-events	Displays all event messages for control modules.
debug ssg ctrl-packets	Displays packet contents handled by control modules.

debug ssg ctrl-events

To display all event messages for control modules, use the **debug ssg ctrl-events** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ssg ctrl-events

no debug ssg ctrl-events

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(3)DC	This command was introduced on the Cisco 6400 node route processor.
12.2(4)B	This command was integrated into Cisco IOS Release 12.2(4)B.
12.2(8)T	This command was integrated into Cisco IOS Release 12.2(8)T.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

This command displays event messages for the control modules, which include all modules that manage the user authentication and service login and logout (RADIUS, PPP, Subblock, and Accounting). An event message is an informational message generated during normal execution.

Examples

The following output is generated by the **debug ssg ctrl-events** command when a host logs in to a service:

```
Router# debug ssg ctrl-events
Mar 16 16:20:30 [192.168.6.1.7.141] 799:02:26:51:SSG-CTL-EVN:Service logon is accepted.
Mar 16 16:20:30 [192.168.6.1.7.141] 800:02:26:51:SSG-CTL-EVN:Send cmd 11 to host 172.16.6.13.
 dst=192.168.100.24:36613
```

Related Commands

Command	Description
debug ssg ctrl-packets	Displays packet contents handled by control modules.
ssg local-forwarding	Displays all error messages for control modules.

debug ssg ctrl-packets

To display packet contents handled by control modules, use the **debug ssg ctrl-packets** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ssg ctrl-packets

no debug ssg ctrl-packets

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(3)DC	This command was introduced on the Cisco 6400 node route processor.
12.2(4)B	This command was integrated into Cisco IOS Release 12.2(4)B.
12.2(8)T	This command was integrated into Cisco IOS Release 12.2(8)T.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

Use this command to show packet messages for the control modules. These modules include all those that manage the user authentication and service login and logout (RADIUS, PPP, Subblock, and Accounting). A packet message displays the contents of a package.

Examples

The following output is generated by using the **debug ssg ctrl-packets** command when a host logs out of a service:

```
Router# debug ssg ctrl-packets
Mar 16 16:23:38 [192.168.6.1.7.141] 968:02:30:00:SSG-CTL-PAK:Received Packet:
Mar 16 16:23:38 [192.168.6.1.7.141] 980:02:30:00:SSG-CTL-PAK:Sent packet:
Mar 16 16:23:39 [192.168.6.1.7.141] 991:02:30:00:SSG-CTL-PAK:
Mar 16 16:23:39 [192.168.6.1.7.141] 992:Received Packet:
```

Related Commands

Command	Description
debug ssg ctrl-events	Displays all event messages for control modules.
ssg local-forwarding	Enables NRP-SSG to forward packets locally.

debug ssg data

To display all data-path packets, use the **debug ssg data** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ssg data

no debug ssg data

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(3)DC	This command was introduced on the Cisco 6400 node route processor.
12.2(4)B	This command was integrated into Cisco IOS Release 12.2(4)B.
12.2(8)T	This command was integrated into Cisco IOS Release 12.2(8)T.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

The **debug ssg data** command shows packets for the data modules. These modules include all those that forward data packets (Dynamic Host Configuration Protocol (DHCP), Domain Name System (DNS), tunneling, fast switching, IP stream, and multicast).

Examples

The following output is generated by using the **debug ssg data** command when a host logs in to and out of a service:

```
Router# debug ssg data
Mar 29 13:45:16 [192.168.5.1.15.21] 45:00:09:24:
SSG-DATA:PS-UP-SetPakOutput=1(Vi6:172.16.5.50->199.199.199.199)
Mar 29 13:45:16 [192.168.5.1.15.21] 46:00:09:24:
SSG-DATA:PS-DN-SetPakOutput=1(Fa0/0/0:171.69.2.132->172.16.5.50)
Mar 29 13:45:16 [192.168.5.1.15.21] 47:00:09:24:
SSG-DATA:FS-UP-SetPakOutput=1(Vi6:172.16.5.50->171.69.43.34)
Mar 29 13:45:16 [192.168.5.1.15.21] 48:00:09:24:
```

Related Commands

Command	Description
debug ssg data-nat	Displays all data-path packets for NAT processing.

debug ssg data-nat

To display all data-path packets for Network Address Translation (NAT) processing, use the **debug ssg data-nat** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ssg data-nat

no debug ssg data-nat

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(3)DC	This command was introduced on the Cisco 6400 node route processor.
12.2(4)B	This command was integrated into Cisco IOS Release 12.2(4)B.
12.2(8)T	This command was integrated into Cisco IOS Release 12.2(8)T.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

The **debug ssg data-nat** command displays packets for the data modules. These modules include all those that forward NAT data packets.

Examples

The following output is generated by using the **debug ssg data-nat** command when a host logs in to and out of a service:

```
Router# debug ssg data-nat
Mar 29 13:43:14 [192.168.5.1.15.21] 35:00:07:21:SSG-DATA:TranslateIP Dst
199.199.199.199->171.69.2.132
Mar 29 13:43:14 [192.168.5.1.15.21] 36:00:07:21:SSG-DATA:TranslateIP Src
171.69.2.132->199.199.199.199
Mar 29 13:43:30 [192.168.5.1.15.21] 39:00:07:38:SSG-DATA:TranslateIP Dst
199.199.199.199->171.69.2.132
Mar 29 13:43:30 [192.168.5.1.15.21] 40:00:07:38:SSG-DATA:TranslateIP Src
171.69.2.132->199.199.199.199
```

Related Commands

Command	Description
debug ssg data	Displays all data-path packets.

debug ssg dhcp

To enable the display of control errors and events related to Service Selection Gateway (SSG) Dynamic Host Configuration Protocol (DHCP), use the debug ssg dhcpcommand in **privileged EXEC**mode. To stop debugging, use the **no** form of this command.

debug ssg dhcp {**error**| **event**} **[ip-address]**

no debug ssg dhcp {**error**| **event**} **[ip-address]**

Syntax Description

error	Enables the display of SSG-DHCP control error information.
event	Enables the display of SSG-DHCP control events information.
ip-address	(Optional) Limits the display of information to the specified IP address.

Command Default

Displays SSG-DHCP information for all IP addresses.

Command Modes

Privileged EXEC

Command History

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

Examples

Examples

The following example shows user login events when DHCP intercept is enabled using the **ssg intercept dhcp** command.

```
debug ssg dhcp
01:01:03:   DHCPD: remote id 020a0000050101011000000000000
01:01:03:   DHCPD: circuit id 00000000
01:01:03: SSG-DHCP-EVN: DHCP-DISCOVER event received. SSG-dhcp awareness feature enabled
01:01:03: DHCPD: DHCPDISCOVER received from client
0063.6973.636f.2d30.3030.632e.3331.6561.2e61.3963.312d.4661.302f.31 on interface
FastEthernet1/0.
01:01:03: DHCPD: Seeing if there is an internally specified pool class:
01:01:03:   DHCPD: htype 1 chaddr 000c.31ea.a9c1
01:01:03:   DHCPD: remote id 020a0000050101011000000000000
01:01:03:   DHCPD: circuit id 00000000
01:01:03: SSG-DHCP-EVN: Get pool name called for 000c.31ea.a9c1. No hostobject
01:01:03: SSG-DHCP-EVN: Get pool class called, class name =
01:01:03: DHCPD: No internally specified class returned
```

```
01:01:03: DHCPD: Sending DHCPOFFER to client
0063.6973.636f.2d30.3030.632e.3331.6561.2e61.3963.312d.4661.302f.31 (5.1.1.2).
01:01:03: DHCPD: child  pool: 5.1.1.0 / 255.255.255.0 (Default-pool)
01:01:03: DHCPD: pool Default-pool has no parent.
01:01:03: DHCPD: child  pool: 5.1.1.0 / 255.255.255.0 (Default-pool)
01:01:03: DHCPD: pool Default-pool has no parent.
01:01:03: DHCPD: child  pool: 5.1.1.0 / 255.255.255.0 (Default-pool)
01:01:03: DHCPD: pool Default-pool has no parent.
01:01:03: DHCPD: broadcasting BOOTREPLY to client 000c.31ea.a9c1.
01:01:03: DHCPD: DHCPREQUEST received from client
0063.6973.636f.2d30.3030.632e.3331.6561.2e61.3963.312d.4661.302f.31.
01:01:03: DHCPD: Sending notification of ASSIGNMENT:
01:01:03:  DHCPD: address 5.1.1.2 mask 255.255.255.0
01:01:03:   DHCPD: htype 1 chaddr 000c.31ea.a9c1
01:01:03:   DHCPD: lease time remaining (secs) = 180
01:01:03: SSG-DHCP-EVN:5.1.1.2: IP address notification received.
01:01:03: SSG-DHCP-EVN:5.1.1.2: HostObject not present
01:01:03: DHCPD: No default domain to append - abort update
01:01:03: DHCPD: Sending DHCPACK to client
0063.6973.636f.2d30.3030.632e.3331.6561.2e61.3963.312d.4661.302f.31 (5.1.1.2).
01:01:03: DHCPD: child  pool: 5.1.1.0 / 255.255.255.0 (Default-pool)
01:01:03: DHCPD: pool Default-pool has no parent.
01:01:03: DHCPD: child  pool: 5.1.1.0 / 255.255.255.0 (Default-pool)
01:01:03: DHCPD: pool Default-pool has no parent.
01:01:03: DHCPD: child  pool: 5.1.1.0 / 255.255.255.0 (Default-pool)
01:01:03: DHCPD: pool Default-pool has no parent.
01:01:03: DHCPD: broadcasting BOOTREPLY to client 000c.31ea.a9c1.
```

Examples

The following example shows user login errors when a user tries to log into two different services that require IP addresses to be assigned from different pools.

```
debug ssg dhcp error
```

```
01:21:58: SSG-CTL-EVN: Checking maximum service count.
01:21:58: SSG-CTL-EVN: Service logon is accepted.
01:21:58: SSG-CTL-EVN: Activating the ConnectionObject.
01:21:58: SSG-DHCP-ERR:6.2.1.2: DHCP pool name of this service is different from,  users
already logged in service DHCP pool name
01:21:58: SSG-CTL-EVN: Connection Activation Failed for host 6.2.1.2
01:21:58: SSG-CTL-EVN: Send cmd 11 to host S6.2.1.2. dst=10.76.86.90:42412
01:21:58: SSG-CTL-PAK: Sent packet:
01:21:58: RADIUS: id= 0, code= Access-Reject, len= 79
```

Related Commands

Command	Description
ssg intercept dhcp	Configures SSG to assign IP addresses from a user's ISP.

debug ssg errors

To display all error messages for the system modules, use the **debug ssg errors** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ssg errors

no debug ssg errors

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(3)DC	This command was introduced on the Cisco 6400 node route processor.
12.2(4)B	This command was integrated into Cisco IOS Release 12.2(4)B.
12.2(8)T	This command was integrated into Cisco IOS Release 12.2(8)T.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

The **debug ssg errors** command displays error messages for the system modules, which include the basic Cisco IOS and other support modules (such as Object Model, Timeout, and Initialization). An error message is the result of an error detected during normal execution.

Examples

The following output is generated by using the **debug ssg errors** command when a PPP over Ethernet (PPPoE) client logs in with an incorrect password:

```
Router# debug ssg errors
Mar 16 08:46:20 [192.168.6.1.7.141] 225:00:16:06:SSG:SSGDoAccounting:
reg_invoke_do_acct returns FALSE
```

Related Commands

Command	Description
debug ssg events	Displays event messages for system modules.
debug ssg packets	Displays packet contents handled by system modules.

debug ssg events

To display event messages for system modules, use the **debug ssg events** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ssg events

no debug ssg events

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(3)DC	This command was introduced on the Cisco 6400 node route processor.
12.2(4)B	This command was integrated into Cisco IOS Release 12.2(4)B.
12.2(8)T	This command was integrated into Cisco IOS Release 12.2(8)T.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

The **debug ssg events**command displays event messages for the system modules, which include the basic Cisco IOS modules and other support modules (such as Object Model, Timeout, and Initialization). An event message is an informational message that appears during normal execution.

Examples

The following output is generated by using the **debug ssg events** command when a PPP over Ethernet (PPPoE) client logs in with the username “username” and the password “cisco”:

```
Router# debug ssg events
Mar 16 08:39:39 [192.168.6.1.7.141] 167:00:09:24:%LINK-3-UPDOWN:
Interface Virtual-Access3, changed state to up
Mar 16 08:39:39 [192.168.6.1.7.141] 168:00:09:25:%LINEPROTO-5-UPDOWN:
Line protocol on Interface Virtual-Access3, changed state to up
Mar 16 08:39:40 [192.168.6.1.7.141] 169:00:09:26:%VPDN-6-AUTHORERR:L2F
NAS LowSlot7 cannot locate a AAA server for Vi3 user username
Mar 16 08:39:40 [192.168.6.1.7.141] 170:HostObject::HostObject:size = 256
Mar 16 08:39:40 [192.168.6.1.7.141] 171:HostObject::Reset
Mar 16 08:39:40 [192.168.6.1.7.141] 172:Service List:
Mar 16 08:39:40 [192.168.6.1.7.141] 175:Service = isp-1
```

Related Commands

Command	Description
debug ssg error	Displays all error messages for the system modules.
debug ssg packets	Displays packet contents handled by system modules.

debug ssg packets

Note Effective with Release 12.2(13)T, the **debug ssg packets** command is replaced by the **debug ssg tcp-redirect** command. See the **debug ssg tcp-redirect** command for more information.

To display packet contents handled by system modules, use the **debug ssg packets** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ssg packets

no debug ssg packets

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(3)DC	This command was introduced on the Cisco 6400 node route processor.
12.2(4)B	This command was integrated into Cisco IOS Release 12.2(4)B.
12.2(8)T	This command was integrated into Cisco IOS Release 12.2(8)T.
12.2(13)T	This command was replaced by the **debug ssg tcp-redirect** command.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

The **debug ssg packets** command displays packet messages for the system modules, which include the basic Cisco IOS and other support modules (such as Object Model, Timeout, Initialization). A packet message displays the contents of a package.

Examples

The following output is generated by using the **debug ssg packets** command when a user is running a Telnet session to 192.168.250.12 and pinging 192.168.250.11:

```
Router# debug ssg packets
19:46:03:SSG-DATA:PS-UP-SetPakOutput=1(Vi2:172.16.17.71->192.168.250.12)
19:46:03:SSG-DATA:PS-UP-SetPakOutput=1(Vi2:172.16.17.71->192.168.250.12)
19:46:03:SSG-DATA:PS-UP-SetPakOutput=1(Vi3:172.16.17.72->192.168.250.12)
19:46:03:SSG-DATA:PS-UP-SetPakOutput=1(Vi2:172.16.17.71->192.168.250.12)
19:46:03:SSG-DATA:PS-UP-SetPakOutput=1(Vi2:172.16.17.71->192.168.250.12)
```

```
19:46:03:SSG-DATA:PS-UP-SetPakOutput=1(Vi2:172.16.17.71->192.168.250.12)
19:46:03:SSG-DATA:PS-UP-SetPakOutput=1(Vi3:172.16.17.72->192.168.250.11)
```

Related Commands

Command	Description
debug ssg errors	Displays all error messages for the system modules.
debug ssg events	Displays event messages for system modules.

debug ssg port-map

To display debugging messages for port-mapping, use the **debug ssg port-map** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ssg port-map {**events**| **packets**}

no debug ssg port-map {**events**| **packets**}

Syntax Description

events	Displays messages for port-map events: create and remove.
packets	Displays port-map packet contents and port address translations.

Command Default

This command is disabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(2)B	This command was introduced on the Cisco 6400 series.
12.2(2)XB	This command was integrated into Cisco IOS Release 12.2(2)XB.
12.2(13)T	This command was integrated into Cisco IOS Release 12.2(13)T.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

This command displays debugging messages for the creation of port maps.

Examples

Using the **debug ssg port-map** command generates the following output when a subscriber logs in to a service:

```
Router# debug ssg port-map events
SSG port-map events debugging is on
Router# show debug
SSG:
  SSG port-map events debugging is on
Router#
00:46:09:SSG-PMAP:Changing state of port-bundle 70.13.60.3:65 from FREE to RESERVED
00:46:09:SSG-PMAP:Changing state of port-bundle 70.13.60.3:65 from RESERVED to INUSE
00:46:10:%LINEPROTO-5-UPDOWN:Line protocol on Interface Virtual-Access2, changed state to
up
Router#
```

```
00:46:25:SSG-PMAP:Allocating new port-mapping:[4148<->1040] for port-bundle 70.13.60.3:65
00:46:29:SSG-PMAP:Allocating new port-mapping:[4149<->1041] for port-bundle 70.13.60.3:65
00:46:31:SSG-PMAP:Allocating new port-mapping:[4150<->1042] for port-bundle 70.13.60.3:65
00:46:31:SSG-PMAP:Allocating new port-mapping:[4151<->1043] for port-bundle 70.13.60.3:65
00:46:31:SSG-PMAP:Allocating new port-mapping:[4152<->1044] for port-bundle 70.13.60.3:65
Router# debug ssg port-map packets
SSG port-map packets debugging is on
Router#
00:51:55:SSG-PMAP:forwarding non-TCP packet
00:51:55:SSG-PMAP:forwarding packet
00:51:55:SSG-PMAP:forwarding non-TCP packet
00:51:55:SSG-PMAP:forwarding packet
00:51:55:SSG-PMAP:forwarding non-TCP packet
00:52:06:SSG-PMAP:srcip:70.13.6.100 srcport:8080  dstip:70.13.60.3 dstport:1044
00:52:06:SSG-PMAP:TCP flags:5011  Seq no:1162897784 Ack no:-1232234715
00:52:06:SSG-PMAP:received TCP-FIN packet
00:52:10:SSG-PMAP:cef:packet bound for default n/w
00:52:10:SSG-PMAP:Checking port-map ACLs
00:52:10:SSG-PMAP:Port-map ACL check passed
00:52:10:SSG-PMAP:cef:punting TCP-SYN packet to process
00:52:10:SSG-PMAP:packet bound for default n/w
00:52:10:SSG-PMAP:fast:punting TCP-SYN packet to process
00:52:10:SSG-PMAP:packet bound for default n/w
00:52:10:SSG-PMAP:translating source address from 10.3.6.1 to 70.13.60.3
00:52:10:SSG-PMAP:translating source port from 4158 to 1040
00:52:10:SSG-PMAP:srcip:70.13.6.100 srcport:8080  dstip:70.13.60.3 dstport:1040
00:52:10:SSG-PMAP:TCP flags:6012  Seq no:1186352744 Ack no:-1232047701
00:52:10:SSG-PMAP:translating destination address from 70.13.60.3 to 10.3.6.1
00:52:10:SSG-PMAP:translating destination port from 1040 to 4158
```

Related Commands

Command	Description
show ssg port-map ip	Displays information on a particular port bundle.
show ssg port-map status	Displays information on port bundles.

debug ssg tcp-redirect

To turn on debug information for the Service Selection Gateway (SSG) Transport Control Protocol (TCP) Redirect for Services feature, use the **debug ssg tcp-redirec**t command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ssg tcp-redirect {**packet**| **error**| **event**}

no debug ssg tcp-redirect {**packet**| **error**| **event**}

Syntax Description

packet	Displays redirection information and any changes made to a packet when it is due for redirection.
error	Displays any SSG TCP redirect errors.
event	Displays any major SSG TCP redirect events or state changes.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(4)B	This command was introduced.
12.2(2)XB	This command was integrated in Cisco IOS Release 12.2(2)XB.
12.2(13)T	This command was integrated into Cisco IOS Release 12.2(13)T. This command replaces the **debug ssg packets** command.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

Use this command to turn on debug information for the SSG TCP Redirect for Services feature. Use the **packet** keyword to display redirection information and any changes made to a packet when it is due for redirection. Use the **error** keyword to display any SSG TCP redirect errors. Use the **event** keyword to display any major SSG TCP redirect events or state changes.

Examples

The following example shows how to display redirection information and any changes made to a packet when it is due for redirection:

```
Router#
debug ssg tcp-redirect packet
```

Direction of the packet "-Up" indicates upstream packets from an SSG user, while "-Down" indicates downstream packets sent to a user:

```
07:13:15:SSG-REDIR-PKT:-Up:unauthorised user at 111.0.0.2 redirected to 9.2.36.253,8080
07:13:15:SSG-REDIR-PKT:-Down:TCP-RST Rxd for user at 111.0.0.2, port 11114
07:13:15:SSG-REDIR-PKT:-Down:return remap for user at 111.0.0.2 redirected from 9.2.36.25
```

The following example shows how to display any SSG TCP redirect errors:

```
Router#
debug ssg tcp-redirect error
07:15:20:SSG-REDIR-ERR:-Up:Packet from 172.0.0.2:11114 has different destination from stored
 connection
```

The following example shows how to display any major SSG TCP redirect events or state changes:

```
Router#
debug ssg tcp-redirect event
```

Upstream packets from users are redirected:

```
06:45:51:SSG-TCP-REDIR:-Up:created new remap entry for unauthorised user at 172.16.0.2
06:45:51:                 Redirect server set to  10.2.36.253,8080
06:45:51:                 Initial src/dest port mapping 11094<->23
06:45:51:SSG-REDIR-EVT: Freeing tcp-remap connections
06:46:21:SSG-REDIR-EVT:Host at 111.0.0.2, connection port  11094  timed out
06:46:21:SSG-REDIR-EVT: Unauthenticated user remapping for 172.16.0.2 removed
```

A host is being activated:

```
06:54:09:SSG-REDIR-EVT:- New Host at 172.16.0.2 set for default initial captivation
06:54:09:SSG-REDIR-EVT:- New Host at 172.16.0.2 set for default advertising captivation
```

Initial captivation begins:

```
06:59:32:SSG-REDIR-EVT:-Up:initial captivate got packet at start of connection (from
111.0.0.2)
06:59:32:SSG-REDIR-EVT:-Up:user at 111.0.0.2 starting initial captivation
06:59:32:SSG-REDIR-EVT:- Up:created new redirect connection and server for user at 111.0.0.2
06:59:32:         Redirect server set to  10.64.131.20,8000
06:59:32:         Initial src/dest port mapping 11109<->80
06:59:48:SSG-REDIR-EVT:-Up:initial captivate got packet at start of connection (from
111.0.0.2)
06:59:48:SSG-REDIR-EVT:-Up:initial captivate timed out for user at 172.16.0.2
06:59:48:SSG-REDIR-EVT:Removing server 10.64.131.20:8000 for host 172.16.0.2
```

Advertising captivation begins:

```
06:59:48:SSG-REDIR-EVT:Removing redirect map for host 172.16.0.2
06:59:48:SSG-REDIR-EVT:-Up:advert captivate got packet at start of connection (from 111.0.0.2)
06:59:48:SSG-REDIR-EVT:-Up:user at 111.0.0.2 starting advertisement captivation
06:59:48:SSG-REDIR-EVT:- Up:created new redirect connection and server for user at 111.0.0.2
06:59:48:         Redirect server set to  10.64.131.20,8000
06:59:48:         Initial src/dest port mapping 11110<->80
```

Related Commands

Command	Description
show ssg tcp-redirect group	Displays information about the captive portal groups and the networks associated with the captive portal groups.
show tcp-redirect mappings	Displays information about the TCP redirect mappings for hosts within your system.
ssg enable	Enables SSG.
ssg tcp-redirect	Enables SSG TCP redirect and enters SSG-redirect mode.

debug ssg transparent login

To display all the Service Selection Gateway (SSG) transparent login control events or errors, use the **debug ssg transparent login**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ssg transparent login {**errors**| **events**} [*ip-address*]

no debug ssg transparent login {**errors**| **events**} [*ip-address*]

Syntax Description

errors	Displays any SSG transparent login errors.
events	Displays significant SSG transparent login events or state changes.
ip-address	(Optional) Displays events or errors for a specified IP address.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.3(1a)BW	This command was introduced.
12.3(3)B	This command was integrated into Cisco IOS Release 12.3(3)B.
12.3(7)T	This command was integrated into Cisco IOS Release 12.3(7)T.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.

Usage Guidelines

Use this command when troubleshooting SSG for problems related to transparent autologon users.

Examples

The following examples show sample output from the **debug ssg transparent login**command. The output is self-explanatory.

Examples

```
*Jan 15 12:34:47.847:SSG-TAL-EVN:100.0.0.2 :Added entry successfully
*Jan 15 12:34:47.847:SSG-TAL-EVN:100.0.0.2 :Attempting authorization
*Jan 15 12:34:47.847:SSG-TAL-EVN:100.0.0.2 :Attempting to send authorization request
*Jan 15 12:35:09.711:SSG-TAL-EVN:100.0.0.2 :Authorization response received
*Jan 15 12:35:09.711:SSG-TAL-EVN:100.0.0.2 :Authorization timedout. User statechanged to
```

```
unidentified
*Jan 15 12:35:09.711:%SSG-5-SSG_TAL_NR:SSG TAL :No response from AAA server. AAA server
might be down or overloaded.
*Jan 15 12:35:09.711:SSG-TAL-EVN:100.0.0.2 :Start SP/NR entry timeout timer for 10 mins
```

Examples

```
*Jan 15 12:40:39.875:SSG-TAL-EVN:100.0.0.2 :Added entry successfully
*Jan 15 12:40:39.875:SSG-TAL-EVN:100.0.0.2 :Attempting authorization
*Jan 15 12:40:39.875:SSG-TAL-EVN:100.0.0.2 :Attempting to send authorization request
*Jan 15 12:40:39.879:SSG-TAL-EVN:100.0.0.2 :Authorization response received
*Jan 15 12:40:39.879:SSG-TAL-EVN:100.0.0.2 :Parsing profile for TP attribute
*Jan 15 12:40:39.879:SSG-TAL-EVN:100.0.0.2 :TP attribute found - Transparent user
*Jan 15 12:40:39.879:SSG-TAL-EVN:100.0.0.2 :Stop SP/NR timer
*Jan 15 12:40:39.879:SSG-TAL-EVN:100.0.0.2 :Idle timer started for 0 secs
*Jan 15 12:40:39.879:SSG-TAL-EVN:100.0.0.2 :Session timer started for 0 secs
```

Examples

```
*Jan 15 12:43:25.363:SSG-TAL-EVN:10.10.10.10 :Added entry successfully
*Jan 15 12:43:25.363:SSG-TAL-EVN:10.10.10.10 :Attempting authorization
*Jan 15 12:43:25.363:SSG-TAL-EVN:10.10.10.10 :Attempting to send authorization request
*Jan 15 12:43:25.939:SSG-TAL-EVN:10.10.10.10 :Authorization response received
*Jan 15 12:43:25.939:SSG-TAL-EVN:10.10.10.10 :Access reject from AAA server. Userstate
changed to suspect
*Jan 15 12:43:25.939:SSG-TAL-EVN:10.10.10.10 :Start SP/NR entry timeout timer for 60 mins
```

Examples

The following is sample output for the **debug ssg transparent login** command when used after all transparent autologon users have been cleared by using the **clear ssg user transparent all** command.

```
*Jan 15 12:47:08.943:SSG-TAL-EVN:10.10.10.10 :Entry removed
*Jan 15 12:47:08.943:SSG-TAL-EVN:10.10.10.10 :Stop SP/NR timer
*Jan 15 12:47:08.943:SSG-TAL-EVN:10.10.10.10 :Stop Idle timer
*Jan 15 12:47:08.943:SSG-TAL-EVN:10.10.10.10 :Stop session timer
*Jan 15 12:47:08.943:SSG-TAL-EVN:10.11.11.11 :Entry removed
*Jan 15 12:47:08.943:SSG-TAL-EVN:10.11.11.11 :Stop SP/NR timer
*Jan 15 12:47:08.943:SSG-TAL-EVN:10.11.11.11 :Stop Idle timer
*Jan 15 12:47:08.943:SSG-TAL-EVN:10.11.11.11 :Stop session timer
*Jan 15 12:47:08.943:SSG-TAL-EVN:10.0.0.2 :Entry removed
*Jan 15 12:47:08.943:SSG-TAL-EVN:10.0.0.2 :Stop SP/NR timer
*Jan 15 12:47:08.943:SSG-TAL-EVN:10.0.0.2 :Stop Idle timer
*Jan 15 12:47:08.943:SSG-TAL-EVN:10.0.0.2 :Stop session timer
```

Related Commands

Command	Description
ssg login transparent	Enables the SSG Transparent Autologon feature.

debug ssl

To display information about Secure Socket Layer (SSL) and Transport Layer Security (TLS) applications, use the **debug ssl**command in privileged EXEC mode. To turn off debugging, use the **no** form of this command.

debug ssl {**error**| **event**| **hdshake**| **traffic**| **openssl** {**errors**| **msg**| **states**}}

no debug ssl {**error**| **event**| **hdshake**| **traffic**| **openssl** {**errors**| **msg**| **states**}}

Syntax Description

error	Displays any errors during control (negotiation) and data phases.
event	Displays SSL negotiation events.
hdshake	Displays SSL HandShake protocol information.
traffic	Displays SSL traffic messages.
openssl	Displays TLS/SSL debugging of the OpenSSL toolkit.
errors	Displays protocol errors, such as a bad packet or authentication failure.
msg	Displays hex dumps of the protocol packets.
states	Displays protocol state transitions.

Command Default

Debugging is not turned on.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.0(5)T	This command was introduced.
12.4(6)T	The **openssl** keyword was added.
12.4(22)T	The **error**, **event**, **hdshake**, and **traffic** keywords were removed.

Usage Guidelines

To display information about SSL and TLS applications, you should first try the **debug ssl openssl errors** command because it will display any obvious failures that are reported by the protocol layer. Next, try the **debug ssl openssl states** command to display problems that are caused by system flow issues that do not produce an error message. If you need more information, you should try the **debug ssl openssl msg** command. This output will be verbose and is rarely useful, but in some circumstances, it can provide a binary dump of the protocol packets. If the problem requires debugging at the level of the packet dumps, it is usually better to use a protocol analyzer (for example, Wireshark).

Note The options available for the **debug ssl** command depend on the version of Cisco IOS software release. See the Command History table for the supported Cisco IOS software releases.

Note It is suggested that when setting debugging, you first enable the **debug ssl openssl errors**command, **debug ssl openssl states**command, and a subset of one of the **debug crypto pki** commands. If you still do not see the problem, you might use a protocol analyzer. The **debug ssl openssl msg** command should probably be used only if you cannot get a packet trace off the wire or if you suspect that the problem is between the wire and the protocol stack.

Examples

The following example shows that the **debug ssl openssl errors** command has been configured:

```
Router# debug ssl openssl errors
```

Related Commands

Command	Description
debug crypto pki messages	Displays debugging messages for the details of the interaction (message dump) between the CA and the router.
debug crypto pki server	Enables debugging for a crypto PKI certificate server.
debug crypto pki transactions	Displays debugging messages for the trace of interaction (message type) between the CA and the router.

debug ssl openssl

To display information about Secure Socket Layer (SSL) and Transport Layer Security (TLS) applications, use the **debug ssl openssl** command in privileged EXEC mode. To turn off debugging, use the **no** form of the command.

debug ssl openssl {**errors**| **msg**| **states**}

no debug ssl openssl {**errors**| **msg**| **states**}

Syntax Description

errors	Displays protocol errors, such as a bad packet or authentication failure.
msg	Displays hex dumps of the protocol packets.
states	Displays protocol state transitions.

Command Default

Debugging is not turned on.

Command Modes

Privileged EXEC (#)

Command History

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

Usage Guidelines

To display information about SSL and TLS applications, you must use the **debug ssl openssl errors** command, because it will display any obvious failures that are reported by the protocol layer. Next, you must use the **debug ssl openssl states** command to display problems that are caused by system flow issues that do not produce an error message. If you need more information, you must use the **debug ssl openssl msg** command. This output will be verbose and is rarely useful, but in some circumstances, it can provide a binary dump of the protocol packets. If the problem requires debugging at the level of the packet dumps, it is usually recommended to use a protocol analyzer (for example, Wireshark).

Examples

The following example shows how to enable the **debug ssl openssl errors** command :

```
Router# debug ssl openssl errors
TLS errors debugging is on
```

Related Commands

Command	Description
debug crypto pki messages	Displays debugging messages for the details of the interaction (message dump) between the CA and the router.
debug crypto pki server	Enables debugging for a crypto PKI certificate server.
debug crypto pki transactions	Displays debugging messages for the trace of interaction (message type) between the CA and the router.

debug ssm

To display diagnostic information about the Segment Switching Manager (SSM) for switched Layer 2 segments, use the **debug ssm** command in privileged EXEC mode. To disable debugging, use the **no** form of this command.

debug ssm {**cm errors**| **cm events**| **fhm errors**| **fhm events**| **sm errors**| **sm events**| **sm counters**| **xdr**}

no debug ssm {**cm errors**| **cm events**| **fhm errors**| **fhm events**| **sm errors**| **sm events**| **sm counters**| **xdr**}

Syntax Description

cm errors	Displays Connection Manager (CM) errors.
cm events	Displays CM events.
fhm errors	Displays Feature Handler Manager (FHM) errors.
fhm events	Displays FHM events.
sm errors	Displays Segment Handler Manager (SM) errors.
sm events	Displays SM events.
sm counters	Displays SM counters.
xdr	Displays external data representation (XDR) messages related to traffic sent across the backplane between Router Processors and line cards.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(26)S	This command was introduced.
12.2(25)S	This command was integrated to Cisco IOS Release 12.2(25)S.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.4(11)T	This command was integrated into Cisco IOS Release 12.4(11)T.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.

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

Usage Guidelines

The SSM manages the data-plane component of the Layer 2 Virtual Private Network (L2VPN) configuration. The CM tracks the connection-level errors and events that occur on an xconnect. The SM tracks the per-segment events and errors on the xconnect.

Use the **debug ssm** command to troubleshoot problems in bringing up the data plane.

This command is generally used only by Cisco engineers for internal debugging of SSM processes.

Examples

The following example shows sample output for the **debug ssm xdr** command:

```
Router# debug ssm xdr

SSM xdr debugging is on
2w5d: SSM XDR: [4096] deallocate segment, len 16
2w5d: SSM XDR: [8193] deallocate segment, len 16
2w5d: %LINK-3-UPDOWN: Interface FastEthernet2/1, changed state to down
2w5d: %LINK-3-UPDOWN: Interface FastEthernet2/1, changed state to up
2w5d: SSM XDR: [4102] provision segment, switch 4101, len 106
2w5d: SSM XDR: [4102] update segment status, len 17
2w5d: SSM XDR: [8199] provision segment, switch 4101, len 206
2w5d: SSM XDR: [4102] update segment status, len 17
2w5d: %SYS-5-CONFIG_I: Configured from console by console
2w5d: %LINK-3-UPDOWN: Interface FastEthernet2/1, changed state to down
2w5d: SSM XDR: [4102] update segment status, len 17
2w5d: %LINK-3-UPDOWN: Interface FastEthernet2/1, changed state to up
2w5d: SSM XDR: [4102] deallocate segment, len 16
2w5d: SSM XDR: [8199] deallocate segment, len 16
2w5d: SSM XDR: [4104] provision segment, switch 4102, len 106
2w5d: SSM XDR: [4104] update segment status, len 17
2w5d: SSM XDR: [8201] provision segment, switch 4102, len 206
2w5d: SSM XDR: [4104] update segment status, len 17
2w5d: SSM XDR: [4104] update segment status, len 17
2w5d: %SYS-5-CONFIG_I: Configured from console by console
```

The following example shows the events that occur on the segment manager when an Any Transport over MPLS (AToM) virtual circuit (VC) configured for Ethernet over MPLS is shut down and then enabled:

```
Router# debug ssm sm events

SSM Connection Manager events debugging is on
Router(config)# interface fastethernet 0/1/0.1

Router(config-subif)# shutdown

09:13:38.159: SSM SM: [SSS:AToM:36928] event Unprovison segment
09:13:38.159: SSM SM: [SSS:Ethernet Vlan:4146] event Unbind segment
09:13:38.159: SSM SM: [SSS:AToM:36928] free segment class
09:13:38.159: SSM SM: [SSS:AToM:36928] free segment
09:13:38.159: SSM SM: [SSS:AToM:36928] event Free segment
09:13:38.159: SSM SM: last segment class freed
09:13:38.159: SSM SM: [SSS:Ethernet Vlan:4146] segment ready
09:13:38.159: SSM SM: [SSS:Ethernet Vlan:4146] event Found segment data
Router(config-subif)# no shutdown

09:13:45.815: SSM SM: [SSS:AToM:36929] event Provison segment
09:13:45.815: label_oce_get_label_bundle: flags 14 label 16
```

```
09:13:45.815: SSM SM: [SSS:AToM:36929] segment ready
09:13:45.815: SSM SM: [SSS:AToM:36929] event Found segment data
09:13:45.815: SSM SM: [SSS:AToM:36929] event Bind segment
09:13:45.815: SSM SM: [SSS:Ethernet Vlan:4146] event Bind segment
```

The following example shows the events that occur on the CM when an AToM VC configured for Ethernet over MPLS is shut down and then enabled:

```
Router(config)# interface fastethernet 0/1/0.1

Router(config-subif)# shutdown

09:17:20.179: SSM CM: [AToM] unprovision segment, id 36929
09:17:20.179: SSM CM: CM FSM: state Open - event Free segment
09:17:20.179: SSM CM: [SSS:AToM:36929] unprovision segment 1
09:17:20.179: SSM CM: [SSS:AToM] shQ request send unprovision complete event
09:17:20.179: SSM CM: [SSS:Ethernet Vlan:4146] unbind segment 2
09:17:20.179: SSM CM: [SSS:Ethernet Vlan] shQ request send ready event
09:17:20.179: SSM CM: SM msg event send unprovision complete event
09:17:20.179: SSM CM: SM msg event send ready event
Router(config-subif)# no shutdown
09:17:35.879: SSM CM: Query AToM to Ethernet Vlan switching, enabled
09:17:35.879: SSM CM: [AToM] provision second segment, id 36930
09:17:35.879: SSM CM: CM FSM: state Down - event Provision segment
09:17:35.879: SSM CM: [SSS:AToM:36930] provision segment 2
09:17:35.879: SSM CM: [AToM] send client event 6, id 36930
09:17:35.879: SSM CM: [SSS:AToM] shQ request send ready event
09:17:35.883: SSM CM: SM msg event send ready event
09:17:35.883: SSM CM: [AToM] send client event 3, id 36930
```

The following example shows the events that occur on the CM and SM when an AToM VC is provisioned and then unprovisioned:

```
Router# debug ssm cm events

SSM Connection Manager events debugging is on
Router# debug ssm sm events
SSM Segment Manager events debugging is on
Router# configure terminal
Router(config)# interface ethernet1/0

Router(config-if)# xconnect 10.55.55.2 101 pw-class mpls
16:57:34: SSM CM: provision switch event, switch id 86040
16:57:34: SSM CM: [Ethernet] provision first segment, id 12313
16:57:34: SSM CM: CM FSM: state Idle - event Provision segment
16:57:34: SSM CM: [SSS:Ethernet:12313] provision segment 1
16:57:34: SSM SM: [SSS:Ethernet:12313] event Provison segment
16:57:34: SSM CM: [SSS:Ethernet] shQ request send ready event
16:57:34: SSM CM: SM msg event send ready event
16:57:34: SSM SM: [SSS:Ethernet:12313] segment ready
16:57:34: SSM SM: [SSS:Ethernet:12313] event Found segment data
16:57:34: SSM CM: Query AToM to Ethernet switching, enabled
16:57:34: SSM CM: [AToM] provision second segment, id 16410
16:57:34: SSM CM: CM FSM: state Down - event Provision segment
16:57:34: SSM CM: [SSS:AToM:16410] provision segment 2
16:57:34: SSM SM: [SSS:AToM:16410] event Provison segment
16:57:34: SSM CM: [AToM] send client event 6, id 16410
16:57:34: label_oce_get_label_bundle: flags 14 label 19
16:57:34: SSM CM: [SSS:AToM] shQ request send ready event
16:57:34: SSM CM: SM msg event send ready event
16:57:34: SSM SM: [SSS:AToM:16410] segment ready
16:57:34: SSM SM: [SSS:AToM:16410] event Found segment data
16:57:34: SSM SM: [SSS:AToM:16410] event Bind segment
16:57:34: SSM SM: [SSS:Ethernet:12313] event Bind segment
16:57:34: SSM CM: [AToM] send client event 3, id 16410
Router# configure terminal

Router(config)# interface e1/0
Router(config-if)# no xconnect

16:57:26: SSM CM: [Ethernet] unprovision segment, id 16387
16:57:26: SSM CM: CM FSM: state Open - event Free segment
```

```
16:57:26: SSM CM: [SSS:Ethernet:16387] unprovision segment 1
16:57:26: SSM SM: [SSS:Ethernet:16387] event Unprovison segment
16:57:26: SSM CM: [SSS:Ethernet] shQ request send unprovision complete event
16:57:26: SSM CM: [SSS:AToM:86036] unbind segment 2
16:57:26: SSM SM: [SSS:AToM:86036] event Unbind segment
16:57:26: SSM CM: SM msg event send unprovision complete event
16:57:26: SSM SM: [SSS:Ethernet:16387] free segment class
16:57:26: SSM SM: [SSS:Ethernet:16387] free segment
16:57:26: SSM SM: [SSS:Ethernet:16387] event Free segment
16:57:26: SSM SM: last segment class freed
16:57:26: SSM CM: unprovision switch event, switch id 12290
16:57:26: SSM CM: [SSS:AToM] shQ request send unready event
16:57:26: SSM CM: SM msg event send unready event
16:57:26: SSM SM: [SSS:AToM:86036] event Unbind segment
16:57:26: SSM CM: [AToM] unprovision segment, id 86036
16:57:26: SSM CM: CM FSM: state Down - event Free segment
16:57:26: SSM CM: [SSS:AToM:86036] unprovision segment 2
16:57:26: SSM SM: [SSS:AToM:86036] event Unprovison segment
16:57:26: SSM CM: [SSS:AToM] shQ request send unprovision complete event
16:57:26: SSM CM: SM msg event send unprovision complete event
16:57:26: SSM SM: [SSS:AToM:86036] free segment class
16:57:26: SSM SM: [SSS:AToM:86036] free segment
16:57:26: SSM SM: [SSS:AToM:86036] event Free segment
16:57:26: SSM SM: last segment class freed
```

Related Commands

Command	Description
show ssm	Displays SSM information for switched Layer 2 segments.

debug sss aaa authorization event

Note Effective with Cisco IOS Release 15.0(1)S, the **debug sss aaa authorization event** command is replaced by the **debug subscriber aaa authorization event** command. See the **debug subscriber aaa authorization event command** for more information.

To display messages about authentication, authorization, and accounting (AAA) authorization events that are part of normal call establishment, use the **debug sss aaa authorization event**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug sss aaa authorization event

no debug sss aaa authorization event

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(13)T	This command was introduced.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
15.0(1)S	This command was replaced by the **debug subscriber aaa authorization event** command.

Examples

The following is sample output of several Subscriber Service Switch (SSS) **debug** commands including the **debug sss aaa authorization event** command. The reports from these commands should be sent to technical personnel at Cisco Systems for evaluation.

```
Router# debug sss event
Router# debug sss error
Router# debug sss state
Router# debug sss aaa authorization event
Router# debug sss aaa authorization fsm
SSS:
  SSS events debugging is on
  SSS error debugging is on
  SSS fsm debugging is on
  SSS AAA authorization event debugging is on
  SSS AAA authorization FSM debugging is on
*Mar  4 21:33:18.248: SSS INFO: Element type is Access-Type, long value is 3
```

```
*Mar  4 21:33:18.248: SSS INFO: Element type is Switch-Id, long value is -1509949436
*Mar  4 21:33:18.248: SSS INFO: Element type is Nasport, ptr value is 6396882C
*Mar  4 21:33:18.248: SSS INFO: Element type is AAA-Id, long value is 7
*Mar  4 21:33:18.248: SSS INFO: Element type is AAA-ACCT_ENBL, long value is 1
*Mar  4 21:33:18.248: SSS INFO: Element type is AccIe-Hdl, ptr value is 78000006
*Mar  4 21:33:18.248: SSS MGR [uid:7]: Event service-request, state changed from wait-for-req
 to wait-for-auth
*Mar  4 21:33:18.248: SSS MGR [uid:7]: Handling Policy Authorize (1 pending sessions)
*Mar  4 21:33:18.248: SSS PM [uid:7]: Need the following key: Unauth-User
*Mar  4 21:33:18.248: SSS PM [uid:7]: Received Service Request
*Mar  4 21:33:18.248: SSS PM [uid:7]: Event <need keys>, State: initial-req to need-init-keys
*Mar  4 21:33:18.248: SSS PM [uid:7]: Policy reply - Need more keys
*Mar  4 21:33:18.248: SSS MGR [uid:7]: Got reply Need-More-Keys from PM
*Mar  4 21:33:18.248: SSS MGR [uid:7]: Event policy-or-mgr-more-keys, state changed from
wait-for-auth to wait-for-req
*Mar  4 21:33:18.248: SSS MGR [uid:7]: Handling More-Keys event
*Mar  4 21:33:20.256: SSS INFO: Element type is Unauth-User, string value is
nobody@example.com
*Mar  4 21:33:20.256: SSS INFO: Element type is AccIe-Hdl, ptr value is 78000006
*Mar  4 21:33:20.256: SSS INFO: Element type is AAA-Id, long value is 7
*Mar  4 21:33:20.256: SSS INFO: Element type is Access-Type, long value is 0
*Mar  4 21:33:20.256: SSS MGR [uid:7]: Event service-request, state changed from wait-for-req
 to wait-for-auth
*Mar  4 21:33:20.256: SSS MGR [uid:7]: Handling Policy Authorize (1 pending sessions)
*Mar  4 21:33:20.256: SSS PM [uid:7]: Received More Initial Keys
*Mar  4 21:33:20.256: SSS PM [uid:7]: Event <rcvd keys>, State: need-init-keys to
check-auth-needed
*Mar  4 21:33:20.256: SSS PM [uid:7]: Handling Authorization Check
*Mar  4 21:33:20.256: SSS PM [uid:7]: Event <send auth>, State: check-auth-needed to
authorizing
*Mar  4 21:33:20.256: SSS PM [uid:7]: Handling AAA service Authorization
*Mar  4 21:33:20.256: SSS PM [uid:7]: Sending authorization request for 'example.com'
*Mar  4 21:33:20.256: SSS AAA AUTHOR [uid:7]:Event <make request>, state changed from idle
 to authorizing
*Mar  4 21:33:20.256: SSS AAA AUTHOR [uid:7]:Authorizing key xyz.com
*Mar  4 21:33:20.260: SSS AAA AUTHOR [uid:7]:AAA request sent for key example.com
*Mar  4 21:33:20.260: SSS AAA AUTHOR [uid:7]:Received an AAA pass
*Mar  4 21:33:20.260: SSS AAA AUTHOR [uid:7]:Event <found service>, state changed from
authorizing to complete
*Mar  4 21:33:20.260: SSS AAA AUTHOR [uid:7]:Found service info for key example.com
*Mar  4 21:33:20.260: SSS AAA AUTHOR [uid:7]:Event <free request>, state changed from
complete to terminal
*Mar  4 21:33:20.260: SSS AAA AUTHOR [uid:7]:Free request
*Mar  4 21:33:20.264: SSS PM [uid:7]: Event <found>, State: authorizing to end
*Mar  4 21:33:20.264: SSS PM [uid:7]: Handling Service Direction
*Mar  4 21:33:20.264: SSS PM [uid:7]: Policy reply - Forwarding
*Mar  4 21:33:20.264: SSS MGR [uid:7]: Got reply Forwarding from PM
*Mar  4 21:33:20.264: SSS MGR [uid:7]: Event policy-start-service-fsp, state changed from
wait-for-auth to wait-for-service
*Mar  4 21:33:20.264: SSS MGR [uid:7]: Handling Connect-Forwarding-Service event
*Mar  4 21:33:20.272: SSS MGR [uid:7]: Event service-fsp-connected, state changed from
wait-for-service to connected
*Mar  4 21:33:20.272: SSS MGR [uid:7]: Handling Forwarding-Service-Connected event
```

Related Commands

Command	Description
debug sss aaa authorization fsm	Displays information about AAA authorization state changes.
debug sss error	Displays diagnostic information about errors that may occur during Subscriber Service Switch call setup.
debug sss event	Displays diagnostic information about Subscriber Service Switch call setup events.

Command	Description
debug sss fsm	Displays diagnostic information about the Subscriber Service Switch call setup state.

debug sss aaa authorization fsm

Note Effective with Cisco IOS Release 15.0(1)S, the **debug sss aaa authorization fsm** command is replaced by the **debug subscriber aaa authorization fsm** command. See the **debug subscriber aaa authorization fsm** command for more information.

To display information about authentication, authorization, and accounting (AAA) authorization state changes, use the **debug sss aaa authorization fsm**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug sss aaa authorization fsm

no debug sss aaa authorization fsm

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(13)T	This command was introduced.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
15.0(1)S	This command was replaced by the **debug subscriber aaa authorization fsm** command.

Examples

The following example shows how to enter this command. See the “Examples” section of the **debug sss aaa authorization event** command page for an example of output.

```
Router# debug sss aaa authorization fsm
```

Related Commands

Command	Description
debug sss aaa authorization event	Displays messages about AAA authorization events that are part of normal call establishment.
debug sss error	Displays diagnostic information about errors that may occur during Subscriber Service Switch call setup.

Command	Description
debug sss event	Displays diagnostic information about Subscriber Service Switch call setup events.
debug sss fsm	Displays diagnostic information about the Subscriber Service Switch call setup state.

debug sss error

Note Effective with Cisco IOS Release 15.0(1)S, the **debug sss error** command is replaced by the **debug subscriber error** command. See the **debug subscriber error** command for more information.

To display diagnostic information about errors that may occur during Subscriber Service Switch (SSS) call setup, use the **debug sss error**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug sss error

no debug sss error

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(13)T	This command was introduced.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
15.0(1)S	This command was replaced by the **debug subscriber error** command.

Examples

The following example shows how to enter this command. See the “Examples” section of the **debug sss aaa authorization event** command page for an example of output.

```
Router# debug sss error
```

Related Commands

Command	Description
debug sss aaa authorization event	Displays messages about AAA authorization events that are part of normal call establishment.
debug sss aaa authorization fsm	Displays information about AAA authorization state changes.

Command	Description
debug sss event	Displays diagnostic information about Subscriber Service Switch call setup events.
debug sss fsm	Displays diagnostic information about the Subscriber Service Switch call setup state.

debug sss event

Note Effective with Cisco IOS Release 15.0(1)S, the **debug sss event** command is replaced by the **debug subscriber event** command. See the **debug subscriber event** command for more information.

To display diagnostic information about Subscriber Service Switch (SSS) call setup events, use the **debug sss event**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug sss event

no debug sss event

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(13)T	This command was introduced.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
15.0(1)S	This command was replaced by the **debug subscriber event** command.

Examples

The following example shows how to enter this command. See the “Examples” section of the **debug sss aaa authorization event**command page for an example of output.

```
Router# debug sss event
```

Related Commands

Command	Description
debug sss aaa authorization event	Displays messages about AAA authorization events that are part of normal call establishment.
debug sss aaa authorization fsm	Displays information about AAA authorization state changes.
debug sss error	Displays diagnostic information about errors that may occur during Subscriber Service Switch call setup.

Command	Description
debug sss fsm	Displays diagnostic information about the Subscriber Service Switch call setup state.

debug sss fsm

Note Effective with Cisco IOS Release 15.0(1)S, the **debug sss fsm** command is replaced by the **debug subscriber fsm** command. See the **debug subscriber fsm** command for more information.

To display diagnostic information about the Subscriber Service Switch (SSS) call setup state, use the **debug sss fsm**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug sss fsm

no debug sss fsm

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(13)T	This command was introduced.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
15.0(1)S	This command was replaced by the **debug subscriber fsm** command.

Examples

The following example shows how to enter this command. See the “Examples” section of the **debug sss aaa authorization event** command page for an example of output.

```
Router# debug sss fsm
```

Related Commands

Command	Description
debug sss aaa authorization event	Displays messages about AAA authorization events that are part of normal call establishment.
debug sss aaa authorization fsm	Displays information about AAA authorization state changes.
debug sss error	Displays diagnostic information about errors that may occur during Subscriber Service Switch call setup.

Command	Description
debug sss event	Displays diagnostic information about the Subscriber Service Switch call setup events.

debug standby

To display Hot Standby Router Protocol (HSRP) state changes, use the **debug standby** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug standby [terse]

no debug standby [terse]

Syntax Description

terse	(Optional) Displays a limited range of HSRP errors, events, and packets.

Command Modes

Privileged EXEC

Command History

Release	Modification
10.0	This command was introduced.

Usage Guidelines

The **debug standby** command displays Hot Standby Protocol state changes and debugging information regarding transmission and receipt of Hot Standby Protocol packets. Use this command to determine whether hot standby routers recognize one another and take the proper actions.

Examples

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

```
Router# debug standby
SB: Ethernet0 state Virgin -> Listen
SB: Starting up hot standby process
SB:Ethernet0 Hello in 192.168.72.21 Active pri 90 hel 3 hol 10 ip 192.168.72.29
SB:Ethernet0 Hello in 192.168.72.21 Active pri 90 hel 3 hol 10 ip 192.168.72.29
SB:Ethernet0 Hello in 192.168.72.21 Active pri 90 hel 3 hol 10 ip 192.168.72.29
SB:Ethernet0 Hello in 192.168.72.21 Active pri 90 hel 3 hol 10 ip 192.168.72.29
SB: Ethernet0 state Listen -> Speak
SB:Ethernet0 Hello out 192.168.72.20 Speak pri 100 hel 3 hol 10 ip 192.168.72.29
SB:Ethernet0 Hello in 192.168.72.21 Active pri 90 hel 3 hol 10 ip 192.168.72.29
SB:Ethernet0 Hello out 192.168.72.20 Speak pri 100 hel 3 hol 10 ip 192.168.72.29
SB:Ethernet0 Hello in 192.168.72.21 Active pri 90 hel 3 hol 10 ip 192.168.72.29
SB:Ethernet0 Hello out 192.168.72.20 Speak pri 100 hel 3 hol 10 ip 192.168.72.29
SB:Ethernet0 Hello in 192.168.72.21 Active pri 90 hel 3 hol 10 ip 192.168.72.29
SB: Ethernet0 state Speak -> Standby
SB:Ethernet0 Hello out 192.168.72.20 Standby pri 100 hel 3 hol 10 ip 192.168.72.29
SB:Ethernet0 Hello in 192.168.72.21 Active pri 90 hel 3 hol 10 ip 192.168.72.29
SB:Ethernet0 Hello out 192.168.72.20 Standby pri 100 hel 3 hol 10 ip 192.168.72.29
SB:Ethernet0 Hello in 192.168.72.21 Active pri 90 hel 3 hol 10 ip 192.168.72.29
SB:Ethernet0 Hello out 192.168.72.20 Standby pri 100 hel 3 hol 10 ip 192.168.72.29
SB:Ethernet0 Hello in 192.168.72.21 Active pri 90 hel 3 hol 10 ip 192.168.72.29
SB: Ethernet0 Coup out 192.168.72.20 Standby pri 100 hel 3 hol 10 ip 192.168.72.29
SB: Ethernet0 state Standby -> Active
SB:Ethernet0 Hello out 192.168.72.20 Active pri 100 hel 3 hol 10 ip 192.168.72.29
SB:Ethernet0 Hello in 192.168.72.21 Speak pri 90 hel 3 hol 10 ip 192.168.72.29
```

```
SB:Ethernet0 Hello out 192.168.72.20 Active pri 100 hel 3 hol 10 ip 192.168.72.29
SB:Ethernet0 Hello in 192.168.72.21 Speak pri 90 hel 3 hol 10 ip 192.168.72.29
SB:Ethernet0 Hello out 192.168.72.20 Active pri 100 hel 3 hol 10 ip 192.168.72.29
```

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

Table 27: debug standby Field Descriptions

Field	Description
SB	Abbreviation for "standby."
Ethernet0	Interface on which a Hot Standby packet was sent or received.
Hello in	Hello packet received from the specified IP address.
Hello out	Hello packet sent from the specified IP address.
pri	Priority advertised in the hello packet.
hel	Hello interval advertised in the hello packet.
hol	Hold-down interval advertised in the hello packet.
ip *address*	Hot Standby group IP address advertised in the hello packet.
state	Transition from one state to another.
Coup out *address*	Coup packet sent by the router from the specified IP address.

The following line indicates that the router is initiating the Hot Standby Protocol. The **standby ip** interface configuration command enables Hot Standby.

```
SB: Starting up hot standby process
```

The following line indicates that a state transition occurred on the interface:

```
SB: Ethernet0 state Listen -> Speak
```

Related Commands

Command	Description
debug condition standby	Filters the output of the **debug standby**command on the basis of HSRP group number.
debug standby errors	Displays error messages related to HSRP.
debug standby events	Displays events related to HSRP.

Command	Description
debug standby events icmp	Displays debugging messages for the HSRP ICMP redirects filter.
debug standby packets	Displays debugging information for packets related to HSRP.

debug standby errors

To display error messages related to Host Standby Router Protocol (HSRP), use the **debug standby errors**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug standby errors

no debug standby errors

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging is not enabled.

Command Modes

Privileged EXEC

Command History

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

Usage Guidelines

You can filter the debug output using interface and HSRP group conditional debugging. To enable interface conditional debugging, use the debug condition interface command. To enable HSRP conditional debugging, use the debug condition standby command.

Examples

The following example enables the display of HSRP errors:

```
Router# debug standby errors
HSRP Errors debugging is on.
```

Related Commands

Command	Description
debug condition standby	Filters the output of the **debug standby**command on the basis of HSRP group number.
debug standby	Displays HSRP state changes.
debug standby events	Displays events related to HSRP.
debug standby events icmp	Displays debugging messages for the HSRP ICMP redirects filter.

Command	Description
debug standby packets	Displays debugging information for packets related to HSRP.

debug standby events

To display events related to Hot Standby Router Protocol (HSRP), use the debug standby events command in privileged EXEC mode. To disable debugging output, use the no form of this command.

debug standby events [**all**| **api**| **arp**| **ha**| **internal** {**data**| **init**| **state**| **timer**}| **protocol**| **redundancy**| **terse**| **track**] [**detail**]

no debug standby events [**all**| **arp**| **ha**| **internal** {**api**| **data**| **init**| **state**| **timer**}| **protocol**| **redundancy**| **terse**| **track**] [**detail**]

Syntax Description

all	(Optional) Displays all HSRP events.
api	(Optional) Displays HSRP application programming interface (API) events.
arp	(Optional) Displays HSRP Address Resolution Protocol (ARP) events.
ha	(Optional) Displays High availability (HA) events.
internal	(Optional) Displays Internal HSRP events.
data	(Optional) Displays HSRP data events.
init	(Optional) Displays HSRP startup and shutdown events.
state	(Optional) Displays HSRP state events.
timer	(Optional) Displays HSRP timer events.
protocol	(Optional) Displays HSRP protocol events.
redundancy	(Optional) Displays HSRP redundancy events.
terse	(Optional) Displays all HSRP packets, except hellos and advertisements.
track	(Optional) Displays HSRP tracking events.
detail	(Optional) Displays detailed debugging information.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.1	This command was introduced.
12.2(8)T	The **api** keyword was added.
12.4(4)T	The **ha** keyword was added.
12.2(25)S	This command was integrated into Cisco IOS Release 12.2(25)S.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.
12.2(33)SXI	The **arp** keyword was added.
12.4(24)T	This command was modified. The **init** keyword was added.
12.2(33)SXI1	This command was modified. The **init** keyword was added.
Cisco IOS XE Release 2.1	This command was integrated into Cisco IOS XE Release 2.1.

Usage Guidelines

You can filter the debug output using interface and HSRP group conditional debugging. To enable interface conditional debugging, use the debug condition interface command. To enable HSRP conditional debugging, use the debug condition standby command.

Examples

The following example shows how to enable the debugging of the active and standby Route Processors (RPs) on an active RP console. The HSRP group is configured on the active RP, and the HSRP state is active.

```
Router# debug standby events ha
!Active RP
*Apr 27 04:13:47.755: HSRP: Gi0/0/1 Grp 101 RF Encode state Listen into sync buffer
*Apr 27 04:13:47.855: HSRP: CF Sync send ok
*Apr 27 04:13:57.755: HSRP: Gi0/0/1 Grp 101 RF Encode state Speak into sync buffer
*Apr 27 04:13:57.855: HSRP: CF Sync send ok
*Apr 27 04:14:07.755: HSRP: Gi0/0/1 Grp 101 RF Encode state Standby into sync buffer
*Apr 27 04:14:07.755: HSRP: Gi0/0/1 Grp 101 RF Encode state Active into sync buffer
*Apr 27 04:14:07.863: HSRP: CF Sync send ok
*Apr 27 04:14:07.867: HSRP: CF Sync send ok
!Standby RP
*Apr 27 04:11:21.011: HSRP: RF CF client 32, entity 0 got msg len 24
*Apr 27 04:11:21.011: HSRP: Gi0/0/1 Grp 101 RF sync state Init -> Listen
*Apr 27 04:11:31.011: HSRP: RF CF client 32, entity 0 got msg len 24
*Apr 27 04:11:31.011: HSRP: Gi0/0/1 Grp 101 RF sync state Listen -> Speak
*Apr 27 04:11:41.071: HSRP: RF CF client 32, entity 0 got msg len 24
*Apr 27 04:11:41.071: HSRP: RF CF client 32, entity 0 got msg len 24
*Apr 27 04:11:41.071: HSRP: Gi0/0/1 Grp 101 RF sync state Speak -> Standby
*Apr 27 04:11:41.071: HSRP: Gi0/0/1 Grp 101 RF sync state Standby -> Active
```

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

Table 28: debug standby events Field Descriptions

Field	Description
RF	Redundancy facility--Internal mechanism that makes Stateful Switchover (SSO) work.
CF	Checkpoint facility--Internal mechanism that makes SSO work.

The following sample shows HSRP debug information when HSRP is configured to send gratuitous ARP packets every four seconds:

```
Router# debug standby event arp detail
HSRP Events debugging is on (arp)
*Jun 27 14:15:51.795: HSRP: Et0/0 Grp 1 Send grat ARP 10.0.0.1 mac 0000.0c07.ac01 (use vMAC)
*Jun 27 14:15:55.755: HSRP: Et0/0 Grp 1 Send grat ARP 10.0.0.1 mac 0000.0c07.ac01 (use vMAC)
*Jun 27 14:15:59.407: HSRP: Et0/0 Grp 1 Send grat ARP 10.0.0.1 mac 0000.0c07.ac01 (use vMAC)
```

Note Debug messages for gratuitous ARP packets are seen only if the **detail**keyword is entered.

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

Table 29: debug standby events detail Field Descriptions

Field	Description
Send grat ARP 10.0.0.1	IP address to which HSRP sends gratuitous ARP packets.
mac	MAC address of the host router to which HSRP sends gratuitous ARP packets.

The following examples show the output of the **debug standby event internal init**command when the IP address of an interface is changed and HSRP makes an internal evaluation to see if the added address permits the currently configured standby address to remain valid.

```
Router# debug standby events internal init
HSRP: Ethernet0/0 vIP intf primary subnet 172.24.1.0 added
.
.
.
HSRP: Ethernet0/0 vIP 172.24.1.254 matches intf primary subnet 172.24.1.0
Router# debug standby events internal init
HSRP: Ethernet0/0 vIP intf secondary subnet 172.24.1.0 added
.
.
.
HSRP: Ethernet0/0 vIP 172.24.1.254 matches intf secondary subnet 172.24.1.0

Router# debug standby events internal init
HSRP: Ethernet0/0 vIP intf secondary subnet 172.24.1.0 deleted
```

```
.
.
.
HSRP: Ethernet0/0 vIP 172.24.1.254 matches no intf subnets
```

Related Commands

Command	Description
debug condition interface	Limits output for some debug commands on the basis of the interface, VC, or VLAN.
debug condition standby	Filters the output of the **debug standby**command on the basis of HSRP group number.
debug standby	Displays HSRP state changes.
debug standby errors	Displays error messages related to HSRP.
debug standby events icmp	Displays debugging messages for the HSRP ICMP redirects filter.
debug standby packets	Displays debugging information for packets related to HSRP.

debug standby events icmp

To display debugging messages for the Hot Standby Router Protocol (HSRP) Internet Control Message Protocol (ICMP) redirects filter, use the **d ebug standby events icmp** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug standby events icmp

no debug standby events icmp

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.1(3)T	This command was introduced.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
Cisco IOS XE Release 2.1	This command was integrated into Cisco IOS XE Release 2.1.

Usage Guidelines

This command helps you determine whether HSRP is filtering an outgoing ICMP redirect message.

Examples

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

```
Router# debug standby events icmp
10:35:20: SB: changing ICMP redirect sent to 20.0.0.4 for dest 30.0.0.2
10:35:20: SB:   gw 20.0.0.2 -> 20.0.0.12, src 20.0.0.11
10:35:20: SB: Use HSRP virtual address 20.0.0.11 as ICMP src
```

If the router being redirected to is passive (HSRP enabled but no active groups), the following debugging message is displayed:

```
10:41:22: SB: ICMP redirect not sent to 20.0.0.4 for dest 40.0.0.3
10:41:22: SB:  20.0.0.3 does not contain an active HSRP group
```

If HSRP could not uniquely determine the gateway used by the host, then the following message is displayed:

```
10:43:08: SB: ICMP redirect not sent to 20.0.0.4 for dest 30.0.0.2
10:43:08: SB: could not uniquely determine IP address for mac 00d0.bbd3.bc22
```

The following messages are also displayed if the **debug ip icmp command**is enabled, in which case the message prefix is changed:

```
10:39:09: ICMP: HSRP changing redirect sent to 20.0.0.4 for dest 30.0.0.2
10:39:09: ICMP:   gw 20.0.0.2 -> 20.0.0.12, src 20.0.0.11
10:39:09: ICMP: Use HSRP virtual address 20.0.0.11 as ICMP src
10:39:09: ICMP: redirect sent to 20.0.0.4 for dest 30.0.0.2, use gw 20.0.0.12
```

Related Commands

Command	Description
debug ip icmp	Displays information on ICMP transactions.

debug standby events neighbor

To display Hot Standby Router Protocol (HSRP) Bidirectional Forwarding Detection (BFD) peering events, use the **debug standby events neighbor** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug standby events neighbor

no debug standby events neighbor

Syntax Description

This command has no arguments or keywords.

Command Default

HSRP neighbor debugging output is not displayed.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.4(11)T	This command was introduced.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.

Usage Guidelines

You can filter the debug output using interface and HSRP group conditional debugging. To enable interface conditional debugging, use the **debug condition** interface command. To enable HSRP conditional debugging, use the **debug condition standby**command.

Examples

In this example, two HSRP routers are configured as neighbors, supporting BFD peering with the **debug standby events neighbor**command configured. The following example shows the debug output that appears when an additional HSRP group is added to Router A:

Examples

```
RouterA# debug standby event neighbor
HSRP Events debugging is on
    (neighbor)
*Oct  3 02:57:48.587: HSRP: Fa2/0 Grp 2 Standby router is local
01:03:49: %HSRP-5-STATECHANGE: FastEthernet2/0 Grp 2 state Speak -> Standby
*Oct  3 02:57:49.087: HSRP: Fa2/0 Grp 2 Active router is local
*Oct  3 02:57:49.087: HSRP: Fa2/0 Grp 2 Standby router is unknown, was local
01:03:50: %HSRP-5-STATECHANGE: FastEthernet2/0 Grp 2 state Standby -> Active
```

Examples

```
RouterB# debug standby event neighbor
```

```
HSRP Events debugging is on
    (neighbor)
*Oct  3 10:00:28.503: HSRP: Fa2/0 Grp 2 Active router is 10.0.0.1 (no local config)
*Oct  3 10:00:28.503: HSRP: Fa2/0 Nbr 10.0.0.1 active for group 2
```

The following example shows the debug output when an additional HSRP group is added to Router B:

Examples

```
*Oct  3 10:02:28.067: HSRP: Fa2/0 Nbr 10.0.0.1 no longer active for group 2 (Disabled)
*Oct  3 10:02:28.503: HSRP: Fa2/0 Grp 2 Active router is 10.0.0.1
*Oct  3 10:02:28.503: HSRP: Fa2/0 Nbr 10.0.0.1 active for group 2
*Oct  3 10:02:48.071: HSRP: Fa2/0 Grp 2 Standby router is local
00:44:28: %HSRP-5-STATECHANGE: FastEthernet2/0 Grp 2 state Speak -> Standby
```

Examples

```
*Oct  3 03:00:08.655: HSRP: Fa2/0 Grp 2 Standby router is 10.0.0.2
*Oct  3 03:00:08.655: HSRP: Fa2/0 Nbr 10.0.0.2 standby for group 2
```

The following is sample debug output showing a possible network outage (the loss of signal between the ports of Router A and B):

Examples

```
*Oct  3 10:09:07.651: HSRP: Fa2/0 Grp 1 Active router is local, was 10.0.0.1
*Oct  3 10:09:07.651: HSRP: Fa2/0 Nbr 10.0.0.1 no longer active for group 1 (Standby)
*Oct  3 10:09:07.651: HSRP: Fa2/0 Grp 1 Standby router is unknown, was local
00:50:48: %HSRP-5-STATECHANGE: FastEthernet2/0 Grp 1 state Standby -> Active
*Oct  3 10:09:08.959: HSRP: Fa2/0 Grp 2 Active router is local, was 10.0.0.1
*Oct  3 10:09:08.959: HSRP: Fa2/0 Nbr 10.0.0.1 no longer active for group 2 (Standby)
*Oct  3 10:09:08.959: HSRP: Fa2/0 Nbr 10.0.0.1 Was active or standby - start passive holddown
*Oct  3 10:09:08.959: HSRP: Fa2/0 Grp 2 Standby router is unknown, was local
00:50:49: %HSRP-5-STATECHANGE: FastEthernet2/0 Grp 2 state Standby -> Active
```

Related Commands

Command	Description
debug bfd	Displays debugging messages about BFD.
debug condition	Limits the output for some debug commands based on specified conditions.
debug condition standby	Limits the debugging output of HSRP state changes.
show bfd neighbor	Displays a line-by-line listing of existing BFD adjacencies.
show standby	Displays HSRP information.
show standby neighbors	Displays information about HSRP neighbors.
standby bfd all-interfaces	Reenables HSRP BFD peering on all interfaces if it has been disabled.
standby ip	Activates HSRP.

debug standby packets

To display debugging information for packets related to Hot Standby Router Protocol (HSRP), use the **debug standby packets**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug standby packets [**advertise**| **all**| **terse**| **coup**| **hello**| **resign**] **[detail]**

no debug standby packet [**advertise**| **all**| **terse**| **coup**| **hello**| **resign**] **[detail]**

Syntax Description

advertise	(Optional) Specifies HSRP advertisement packets.
all	(Optional) Specifies all HSRP packets.
terse	(Optional) Specifies all HSRP packets, except hellos and advertisements.
coup	(Optional) Specifies HSRP coup packets.
hello	(Optional) Specifies HSRP hello packets.
resign	(Optional) Specifies HSRP resign packets.
detail	(Optional) Specifies HSRP packets in detail.

Command Default

Debugging is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.1	This command was introduced.
12.2	The **advertise** keyword was added.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

You can filter the debug output using interface and HSRP group conditional debugging. To enable interface conditional debugging, use the **debug condition interface** command. To enable HSRP conditional debugging, use the **debug condition standby** command.

Note HSRP advertisement packets are packets that are related to HSRP interfaces. Other packet types, including, hello, coup, and resign packets relate to an HSRP group.

Examples

The following example show how to enable the display of all HSRP packets:

```
Router# debug standby packets all
HSRP Packets debugging is on.
```

Related Commands

Command	Description
debug condition interface	Limits output for some debugging commands based on the interfaces.
debug condition standby	Filters the output of the **debug standby**command on the basis of HSRP group number.
debug standby	Displays HSRP state changes.
debug standby errors	Displays error messages related to HSRP.
debug standby events	Displays events related to HSRP.
debug standby events icmp	Displays debugging messages for the HSRP ICMP redirects filter.

debug stun packet

To display information on packets traveling through the serial tunnel (STUN) links, use the **debug stun packet** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug stun packet [*group*] [*address*]

no debug stun packet [*group*] [*address*]

Syntax Description

group	(Optional) A decimal integer assigned to a group. Using this option limits output to packets associated with the specified STUN group.
address	(Optional) The output is further limited to only those packets containing the specified STUN address. The *address* argument is in the appropriate format for the STUN protocol running for the specified group.

Command Modes

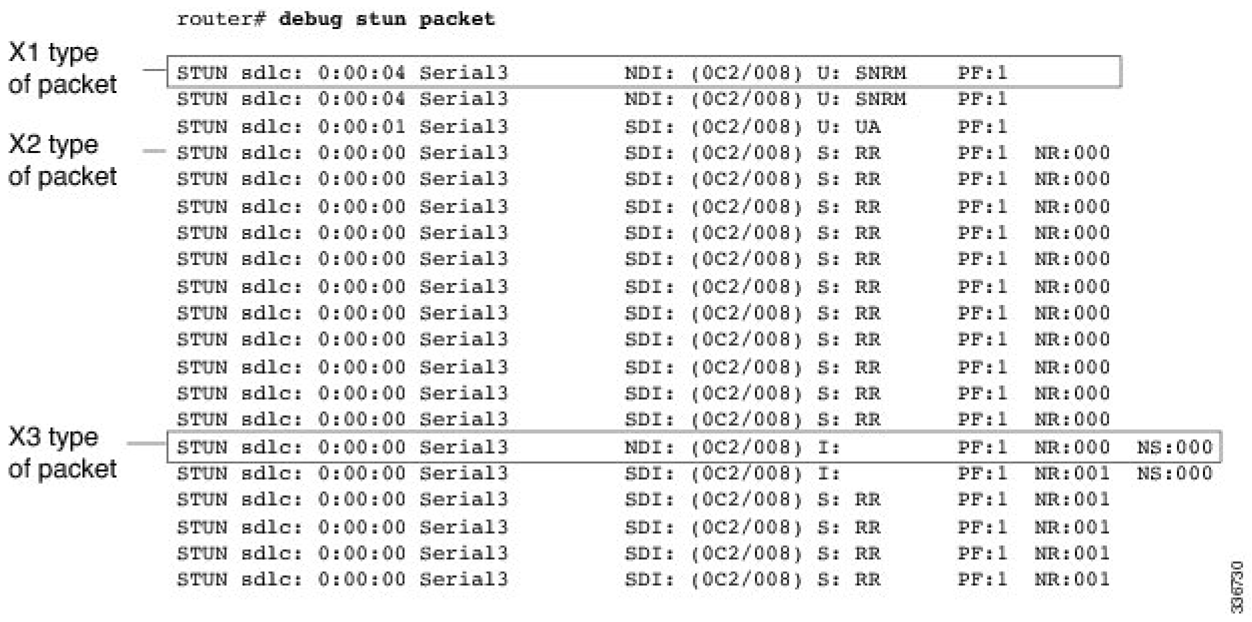
Privileged EXEC

Usage Guidelines

Because using this command is processor intensive, it is best to use it after regular business hours, rather than in a production environment. It is also best to turn this command on by itself, rather than use it in conjunction with other **debug** commands.

Examples

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



```
router# debug stun packet
STUN sdlc: 0:00:04 Serial3        NDI: (0C2/008) U: SNRM    PF:1
STUN sdlc: 0:00:04 Serial3        NDI: (0C2/008) U: SNRM    PF:1
STUN sdlc: 0:00:01 Serial3        SDI: (0C2/008) U: UA      PF:1
STUN sdlc: 0:00:00 Serial3        SDI: (0C2/008) S: RR      PF:1  NR:000
STUN sdlc: 0:00:00 Serial3        SDI: (0C2/008) S: RR      PF:1  NR:000
STUN sdlc: 0:00:00 Serial3        SDI: (0C2/008) S: RR      PF:1  NR:000
STUN sdlc: 0:00:00 Serial3        SDI: (0C2/008) S: RR      PF:1  NR:000
STUN sdlc: 0:00:00 Serial3        SDI: (0C2/008) S: RR      PF:1  NR:000
STUN sdlc: 0:00:00 Serial3        SDI: (0C2/008) S: RR      PF:1  NR:000
STUN sdlc: 0:00:00 Serial3        SDI: (0C2/008) S: RR      PF:1  NR:000
STUN sdlc: 0:00:00 Serial3        SDI: (0C2/008) S: RR      PF:1  NR:000
STUN sdlc: 0:00:00 Serial3        SDI: (0C2/008) S: RR      PF:1  NR:000
STUN sdlc: 0:00:00 Serial3        SDI: (0C2/008) S: RR      PF:1  NR:000
STUN sdlc: 0:00:00 Serial3        SDI: (0C2/008) S: RR      PF:1  NR:000
STUN sdlc: 0:00:00 Serial3        NDI: (0C2/008) I:         PF:1  NR:000  NS:000
STUN sdlc: 0:00:00 Serial3        SDI: (0C2/008) I:         PF:1  NR:001  NS:000
STUN sdlc: 0:00:00 Serial3        SDI: (0C2/008) S: RR      PF:1  NR:001
STUN sdlc: 0:00:00 Serial3        SDI: (0C2/008) S: RR      PF:1  NR:001
STUN sdlc: 0:00:00 Serial3        SDI: (0C2/008) S: RR      PF:1  NR:001
STUN sdlc: 0:00:00 Serial3        SDI: (0C2/008) S: RR      PF:1  NR:001
```

The following line describes an X1 type of packet:

```
STUN sdlc: 0:00:04 Serial3         NDI: (0C2/008) U: SNRM    PF:1
```

The table below describes the significant fields in this line of **debug stun packet** output.

Table 30: debug stun packet Field Descriptions

Field	Description
STUN sdlc:	Indication that the STUN feature is providing the information.
0:00:04	Time elapsed since receipt of the previous packet.
Serial3	Interface type and unit number reporting the event.
NDI:	Type of cloud separating the Synchronous Data Link Control (SDL) end nodes. Possible values are as follows: • NDI--Network input • SDI--Serial link
0C2	SDLC address of the SDLC connection.
008	Modulo value of 8.

Field	Description
U: SNRM	Frame type followed by the command or response type. In this case it is an Unnumbered frame that contains a Set Normal Response Mode (SNRM) command. The possible frame types are as follows: • I--Information frame • S--Supervisory frame. The possible commands and responses are: RR (Receive Ready), RNR (Receive Not Ready), and REJ (Reject). • U--Unnumbered frame. The possible commands are: UI (Unnumbered Information), SNRM, DISC/RD (Disconnect/Request Disconnect), SIM/RIM, XID Exchange Identification), TEST. The possible responses are UA (unnumbered acknowledgment), DM (Disconnected Mode), and FRMR (Frame Reject Mode)
PF:1	Poll/Final bit. Possible values are as follows: • 0--Off • 1--On

The following line of output describes an X2 type of packet:

```
STUN sdlc: 0:00:00 Serial3         SDI: (0C2/008) S: RR      PF:1 NR:000
```

All the fields in the previous line of output match those for an X1 type of packet, except the last field, which is additional. NR:000 indicates a receive count of 0; the range for the receive count is 0 to 7.

The following line of output describes an X3 type of packet:

```
STUN sdlc: 0:00:00 Serial3         SDI: (0C2/008) S:I PF:1 NR:000 NS:000
```

All fields in the previous line of output match those for an X2 type of packet, except the last field, which is additional. NS:000 indicates a send count of 0; the range for the send count is 0 to 7.

debug subscriber aaa authorization

To display diagnostic information about authentication, authorization, and accounting (AAA) authorization of Intelligent Services Gateway (ISG) subscriber sessions, use the **debug subscriber aaa authorization**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug subscriber aaa authorization {**event**| **fsm**}

no debug sss aaa authorization {**event**| **fsm**}

Syntax Description

event	Display information about AAA authorization events that occur during ISG session establishment.
fsm	Display information about AAA authorization state changes for ISG subscriber sessions.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(28)SB	This command was introduced.

Examples

The following is sample output of several **debug subscriber**commands, including the **debug subscriber aaa authorization**command. The reports from these commands should be sent to technical personnel at Cisco Systems for evaluation.

```
Router# debug subscriber event
Router# debug subscriber error
Router# debug subscriber state
Router# debug subscriber aaa authorization event
Router# debug subscriber aaa authorization fsm
SSS:
  SSS events debugging is on
  SSS error debugging is on
  SSS fsm debugging is on
  SSS AAA authorization event debugging is on
  SSS AAA authorization FSM debugging is on
*Mar  4 21:33:18.248: SSS INFO: Element type is Access-Type, long value is 3
*Mar  4 21:33:18.248: SSS INFO: Element type is Switch-Id, long value is -1509949436
*Mar  4 21:33:18.248: SSS INFO: Element type is Nasport, ptr value is 6396882C
*Mar  4 21:33:18.248: SSS INFO: Element type is AAA-Id, long value is 7
*Mar  4 21:33:18.248: SSS INFO: Element type is AAA-ACCT_ENBL, long value is 1
*Mar  4 21:33:18.248: SSS INFO: Element type is AccIe-Hdl, ptr value is 78000006
*Mar  4 21:33:18.248: SSS MGR [uid:7]: Event service-request, state changed from wait-for-req
 to wait-for-auth
*Mar  4 21:33:18.248: SSS MGR [uid:7]: Handling Policy Authorize (1 pending sessions)
*Mar  4 21:33:18.248: SSS PM [uid:7]: Need the following key: Unauth-User
*Mar  4 21:33:18.248: SSS PM [uid:7]: Received Service Request
```

```
*Mar  4 21:33:18.248: SSS PM [uid:7]: Event <need keys>, State: initial-req to need-init-keys
*Mar  4 21:33:18.248: SSS PM [uid:7]: Policy reply - Need more keys
*Mar  4 21:33:18.248: SSS MGR [uid:7]: Got reply Need-More-Keys from PM
*Mar  4 21:33:18.248: SSS MGR [uid:7]: Event policy-or-mgr-more-keys, state changed from
wait-for-auth to wait-for-req
*Mar  4 21:33:18.248: SSS MGR [uid:7]: Handling More-Keys event
*Mar  4 21:33:20.256: SSS INFO: Element type is Unauth-User, string value is nobody2@xyz.com
*Mar  4 21:33:20.256: SSS INFO: Element type is AccIe-Hdl, ptr value is 78000006
*Mar  4 21:33:20.256: SSS INFO: Element type is AAA-Id, long value is 7
*Mar  4 21:33:20.256: SSS INFO: Element type is Access-Type, long value is 0
*Mar  4 21:33:20.256: SSS MGR [uid:7]: Event service-request, state changed from wait-for-req
 to wait-for-auth
*Mar  4 21:33:20.256: SSS MGR [uid:7]: Handling Policy Authorize (1 pending sessions)
*Mar  4 21:33:20.256: SSS PM [uid:7]: Received More Initial Keys
*Mar  4 21:33:20.256: SSS PM [uid:7]: Event <rcvd keys>, State: need-init-keys to
check-auth-needed
*Mar  4 21:33:20.256: SSS PM [uid:7]: Handling Authorization Check
*Mar  4 21:33:20.256: SSS PM [uid:7]: Event <send auth>, State: check-auth-needed to
authorizing
*Mar  4 21:33:20.256: SSS PM [uid:7]: Handling AAA service Authorization
*Mar  4 21:33:20.256: SSS PM [uid:7]: Sending authorization request for 'xyz.com'
*Mar  4 21:33:20.256: SSS AAA AUTHOR [uid:7]:Event <make request>, state changed from idle
 to authorizing
*Mar  4 21:33:20.256: SSS AAA AUTHOR [uid:7]:Authorizing key xyz.com
*Mar  4 21:33:20.260: SSS AAA AUTHOR [uid:7]:AAA request sent for key xyz.com
*Mar  4 21:33:20.260: SSS AAA AUTHOR [uid:7]:Received an AAA pass
*Mar  4 21:33:20.260: SSS AAA AUTHOR [uid:7]:Event <found service>, state changed from
authorizing to complete
*Mar  4 21:33:20.260: SSS AAA AUTHOR [uid:7]:Found service info for key xyz.com
*Mar  4 21:33:20.260: SSS AAA AUTHOR [uid:7]:Event <free request>, state changed from
complete to terminal
*Mar  4 21:33:20.260: SSS AAA AUTHOR [uid:7]:Free request
*Mar  4 21:33:20.264: SSS PM [uid:7]: Event <found>, State: authorizing to end
*Mar  4 21:33:20.264: SSS PM [uid:7]: Handling Service Direction
*Mar  4 21:33:20.264: SSS PM [uid:7]: Policy reply - Forwarding
*Mar  4 21:33:20.264: SSS MGR [uid:7]: Got reply Forwarding from PM
*Mar  4 21:33:20.264: SSS MGR [uid:7]: Event policy-start-service-fsp, state changed from
wait-for-auth to wait-for-service
*Mar  4 21:33:20.264: SSS MGR [uid:7]: Handling Connect-Forwarding-Service event
*Mar  4 21:33:20.272: SSS MGR [uid:7]: Event service-fsp-connected, state changed from
wait-for-service to connected
*Mar  4 21:33:20.272: SSS MGR [uid:7]: Handling Forwarding-Service-Connected event
```

Related Commands

Command	Description
debug sss error	Displays diagnostic information about errors that may occur during Subscriber Service Switch call setup.
debug sss event	Displays diagnostic information about Subscriber Service Switch call setup events.
debug sss fsm	Displays diagnostic information about the Subscriber Service Switch call setup state.

debug subscriber error

To display diagnostic information about errors that may occur during Intelligent Services Gateway (ISG) subscriber session setup, use the **debug subscriber error**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug subscriber error

no debug subscriber error

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(28)SB	This command was introduced.

Examples

The following sample output for the **debug subscriber error** command indicates that the session is stale since the session handle has already been destroyed.

```
Router# debug subscriber error
*Sep 20 22:39:49.455: SSS MGR: Session handle [EF000002] destroyed already
```

Related Commands

Command	Description
debug sss aaa authorization event	Displays messages about AAA authorization events that are part of normal call establishment.
debug sss event	Displays diagnostic information about Subscriber Service Switch call setup events.
debug sss fsm	Displays diagnostic information about the Subscriber Service Switch call setup state.

debug subscriber event

To display diagnostic information about Intelligent Services Gateway (ISG) subscriber session setup events, use the **debug subscriber event**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug subscriber event

no debug subscriber event

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(28)SB	This command was introduced.

Examples

The following sample output for the **debug subscriber event** commands indicates that the system has determined that the session should be locally terminated. The local termination module determines that an interface description block (IDB) is not required for this session, and it sets up the data plane for packet switching.

```
Router# debug subscriber event
*Sep 20 22:21:08.223: SSS MGR [uid:2]: Handling Connect Local Service action
*Sep 20 22:21:08.223: SSS LTERM [uid:2]: Processing Local termination request
*Sep 20 22:21:08.223: SSS LTERM [uid:2]: L3 session - IDB not required for setting up service
*Sep 20 22:21:08.223: SSS LTERM [uid:2]: Interface already present or not required for
service
*Sep 20 22:21:08.223: SSS LTERM [uid:2]: Segment provision successful
```

Related Commands

Command	Description
debug sss aaa authorization event	Displays messages about AAA authorization events that are part of normal call establishment.
debug sss error	Displays diagnostic information about errors that may occur during Subscriber Service Switch call setup.
debug sss fsm	Displays diagnostic information about the Subscriber Service Switch call setup state.

debug subscriber feature

To display diagnostic information about the installation and removal of Intelligent Services Gateway (ISG) features on ISG subscriber sessions, use the **debug subscriber feature** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug subscriber feature {**all**| **detail**| **error**| **event**| **name** *feature-name* {**detail**| **error**| **event**| **packet**}| **packet** [**detail**| **full**] [**issu** {**event**| **error**}] [**ccm** {**event**| **error**}]}

no debug subscriber feature {**all**| **detail**| **error**| **event**| **name** *feature-name* {**detail**| **error**| **event**| **packet**}| **packet** [**detail**| **full**] [**issu** {**event**| **error**}] [**ccm** {**event**| **error**}]}

Syntax Description

all	Displays information about all features.
detail	The **detail** keyword can be used in one of the following three ways: • If used with no other keywords, displays detailed information about all features • If a feature name is specified with the **name** *feature-name* keyword and argument, displays detailed information about the specific feature. The **detail** keyword can be used with the following *feature-name* values: • **accounting** • **compression** • **modem-on-hold** • **policing** • **traffic-classification** • If used with the **packet** keyword, displays a partial dump of packets as ISG features are being applied to the packets.
error	Displays information about errors for all features or a specified feature.
event	Displays information about events for all features or a specified feature.
name	Displays information specific to feature.

feature-name	Name of the ISG feature. Possible values are the following: • **access-list** • **accounting** • **compression** • **filter** • **idle-timer** • **interface-config** • **ip-config** • **l4redirect** • **modem-on-hold** • **policing** • **portbundle** • **prepaid-idle** • **session-timer** • **static-routes** • **time-monitor** • **volume-monitor**
issu	Displays information about events and errors for all features or a specified feature as they occur.
ccm	Displays information about a specific feature checkpointing activity. If the **ccm** keyword is not specified, event and error logging is specific to the feature's interaction with the cluster control manager (CCM).
packet	Displays information about packets as ISG features are being applied to the packets. If a feature name is specified with the **name** *feature-name* keyword and argument, packet information about the specific feature is displayed. The **packet** keyword can be used with the following *feature-name* values: • **access-list** • **l4redirect** • **policing** • **portbundle**

full	(Optional) Displays a full dump of a packet as ISG features are being applied to it.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(28)SB	This command was introduced.
12.2(33)SRC	This command was integrated into Cisco IOS Release12.2(33)SRC.
Cisco IOS XE Release 3.5S	This command was modified. The **traffic-classification** keyword was removed as a choice for the *feature-name* argument.

Examples

The following sample output from the **debug subscriber feature** command indicates that the idle timeout feature has been successfully installed on the inbound segment.

```
Router# debug subscriber feature event
```

```
*Sep 20 22:28:57.903: SSF[myservice/uid:6/Idle Timeout]: Group feature install
*Sep 20 22:28:57.903: SSF[uid:6/Idle Timeout]: Adding feature to inbound segment(s)
```

debug subscriber fsm

To display diagnostic information about Intelligent Services Gateway (ISG) subscriber session state change, use the **debug subscriber fsm**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug subscriber fsm

no debug subscriber fsm

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(28)SB	This command was introduced.

Examples

The following sample output for the **debug subscriber fsm** command indicates that the session has been disconnected by the client, and the system is cleaning up the session by disconnecting the network service and removing any installed features.

```
Router# deb
ug subscriber fs
m
*Sep 20 22:35:10.495: SSS MGR [uid:5]: Event client-disconnect, state changed from connected
 to disconnecting-fsp-feat
```

debug subscriber packet

To display information about packets as they traverse the subscriber service switch (SSS) path, use the **debug subscriber packet** command in privileged EXEC mode. To disable debugging, use the **no** form of this command.

debug subscriber packet {**detail**| **error**| **event**| **full**}

no debug subscriber packet {**detail**| **error**| **event**| **full**}

Syntax Description

detail	Displays a partial dump of packets as they traverse the SSS path.
error	Displays any packet-switching errors that occur when a packet traverses the SSS path.
event	Displays packet-switching events that occur when a packet traverses the SSS path.
full	Displays a full dump of packets as they traverse the SSS path.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(28)SB	This command was introduced.

Examples

The following example show sample output for the **debug subscriber packet**command with the **full**keyword. This output is for a PPPoE session configured with forwarding.

```
SSS Switch: Pak encap size, old: 60, new: 24
SSS Switch: Pak 0285C458 sz 66 encap 14
*Feb  9 15:47:13.659: 000000 AA BB CC 00 0B 01 AA BB  D.......
*Feb  9 15:47:13.659: 000008 CC 00 0C 01 08 00 45 00  ......N.
*Feb  9 15:47:13.659: 000010 00 34 00 28 00 00 FE 11  .4.(....
*Feb  9 15:47:13.659: 000018 F2 9D AC 12 B8 E7 AC 12  ........
*Feb  9 15:47:13.659: 000020 B8 E6 06 A5 06 A5 00 20  .......
*Feb  9 15:47:13.659: 000028 00 00 C0 01 02 00 00 02  ........
*Feb  9 15:47:13.659: 000030 00 01 00 18 00 00 FC A7  ........
*Feb  9 15:47:13.659: 000038 2E B3 FF 03 C2 23 03 01  .....#..
*Feb  9 15:47:13.659: 000040 00 04                    ..
SSS Switch: Pak encap size, old: 60, new: 24
SSS Switch: Pak 0285C458 sz 72 encap 14
*Feb  9 15:47:13.691: 000000 AA BB CC 00 0B 01 AA BB  D.......
*Feb  9 15:47:13.691: 000008 CC 00 0C 01 08 00 45 00  ......N.
*Feb  9 15:47:13.691: 000010 00 3A 00 2A 00 00 FE 11  .:.*....
```

```
*Feb  9 15:47:13.691: 000018 F2 95 AC 12 B8 E7 AC 12  ........
*Feb  9 15:47:13.691: 000020 B8 E6 06 A5 06 A5 00 26  .......&
*Feb  9 15:47:13.691: 000028 00 00 C0 01 02 00 00 02  ........
*Feb  9 15:47:13.691: 000030 00 01 00 1E 00 00 FC A7  ........
*Feb  9 15:47:13.691: 000038 2E B3 FF 03 80 21 01 01  .....!..
*Feb  9 15:47:13.691: 000040 00 0A 03 06 3A 3A 3A 3A  ....::::
SSS Switch: Pak encap size, old: 24, new: 46
SSS Switch: Pak 027A5BE8 sz 36 encap 18
*Feb  9 15:47:13.691: 000000 AA BB CC 00 0B 00 AA BB  D.......
*Feb  9 15:47:13.691: 000008 CC 00 0A 00 81 00 01 41  .......a
*Feb  9 15:47:13.691: 000010 88 64 11 00 00 01 00 0C  .dN.....
*Feb  9 15:47:13.691: 000018 80 21 01 01 00 0A 03 06  .!......
*Feb  9 15:47:13.691: 000020 00 00 00 00              ....
SSS Switch: Pak encap size, old: 60, new: 24
SSS Switch: Pak 0285C458 sz 72 encap 14
*Feb  9 15:47:13.691: 000000 AA BB CC 00 0B 01 AA BB  D.......
*Feb  9 15:47:13.691: 000008 CC 00 0C 01 08 00 45 00  ......N.
*Feb  9 15:47:13.691: 000010 00 3A 00 2C 00 00 FE 11  .:.,....
*Feb  9 15:47:13.691: 000018 F2 93 AC 12 B8 E7 AC 12  ........
*Feb  9 15:47:13.691: 000020 B8 E6 06 A5 06 A5 00 26  .......&
*Feb  9 15:47:13.691: 000028 00 00 C0 01 02 00 00 02  ........
*Feb  9 15:47:13.691: 000030 00 01 00 1E 00 00 FC A7  ........
*Feb  9 15:47:13.691: 000038 2E B3 FF 03 80 21 03 01  .....!..
*Feb  9 15:47:13.691: 000040 00 0A 03 06 09 00 00 1F  ........
```

Related Commands

Command	Description
debug subscriber feature	Displays diagnostic information about the installation and removal of ISG features on subscriber sessions.

debug subscriber policy

To display diagnostic information about policy execution related to Intelligent Services Gateway (ISG) subscriber sessions, use the **debug subscriber policy** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug subscriber policy {**all**| **detail**| **error**| **event**| **fsm**| **prepaid**| {**condition**| **idmgr**| **profile**| **push**| **rule**| **service**} [**detail**| **error**| **event**]| **dpm** [**error**| **event**]| **webportal** {**detail**| **error**| **event**}}

no debug subscriber policy {**all**| **detail**| **error**| **event**| **fsm**| **prepaid**| {**condition**| **idmgr**| **profile**| **push**| **rule**| **service**} [**detail**| **error**| **event**]| **dpm** [**error**| **event**]| **webportal** {**detail**| **error**| **event**}}

Syntax Description

all	Displays information about all policies.
detail	Displays detailed information about all policies or the specified type of policy.
error	Displays policy execution errors for all policies or the specified type of policy.
event	Displays policy execution events for all policies or the specified type of policy.
fsm	Displays information about state changes during policy execution.
prepaid	Displays information about ISG prepaid policy execution.
condition	Displays information related to the evaluation of ISG control class maps.
idmgr	Displays information about policy execution related to identity.
profile	Displays information about the policy manager subscriber profile database.
push	Displays policy information about dynamic updates to subscriber profiles from policy servers.
rule	Displays information about control policy rules.
service	Displays policy information about service profile database events for subscriber sessions.

dpm	Displays information about Dynamic Host Configuration Protocol (DHCP) in relation to subscriber sessions.
webportal	Displays policy information about the web portal in relation to subscriber sessions.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(28)SB	This command was introduced.

Examples

The following example shows sample output for the **debug subscriber policy** command with the **events** keyword. This output indicates the creation of a new session. “Updated key list” indicates important attributes and information associated with the session.

```
*Feb  7 18:58:24.519: SSS PM [0413FC58]: Create context 0413FC58
*Feb  7 18:58:24.519: SSS PM [uid:4][0413FC58]: Authen status update; is now "unauthen"
*Feb  7 18:58:24.519: SSS PM [uid:4][0413FC58]: Updated NAS port for AAA ID 14
*Feb  7 18:58:24.519: SSS PM [uid:4][0413FC58]: Updated key list:
*Feb  7 18:58:24.519: SSS PM [uid:4][0413FC58]:   Access-Type = 15 (IP)
*Feb  7 18:58:24.519: SSS PM [uid:4][0413FC58]:   Protocol-Type = 4 (IP)
*Feb  7 18:58:24.519: SSS PM [uid:4][0413FC58]:   Media-Type = 2 (IP)
*Feb  7 18:58:24.519: SSS PM [uid:4][0413FC58]:   IP-Address = 10.0.0.2 (0A000002)
*Feb  7 18:58:24.519: SSS PM [uid:4][0413FC58]:   IP-Address-VRF = IP 10.0.0.2:0
*Feb  7 18:58:24.519: SSS PM [uid:4][0413FC58]:   source-ip-address = 037FBB78
*Feb  7 18:58:24.519: SSS PM [uid:4][0413FC58]:   Mac-Address = aabb.cc00.6500
*Feb  7 18:58:24.519: SSS PM [uid:4][0413FC58]:   Final = 1 (YES)
*Feb  7 18:58:24.519: SSS PM [uid:4][0413FC58]:   Authen-Status = 1 (Unauthenticated)
*Feb  7 18:58:24.519: SSS PM [uid:4][0413FC58]:   Nasport = PPPoEoE: slot 0 adapter 0 port
 0
*Feb  7 18:58:24.519: SSS PM [uid:4][0413FC58]: Updated key list:
*Feb  7 18:58:24.519: SSS PM [uid:4][0413FC58]:   Access-Type = 15 (IP)
*Feb  7 18:58:24.519: SSS PM [uid:4][0413FC58]:   Protocol-Type = 4 (IP)
*Feb  7 18:58:24.519: SSS PM [uid:4][0413FC58]:   Media-Type = 2 (IP)
*Feb  7 18:58:24.519: SSS PM [uid:4][0413FC58]:   IP-Address = 10.0.0.2 (0A000002)
*Feb  7 18:58:24.519: SSS PM [uid:4][0413FC58]:   IP-Address-VRF = IP 10.0.0.2:0
*Feb  7 18:58:24.519: SSS PM [uid:4][0413FC58]:   source-ip-address = 037FBB78
*Feb  7 18:58:24.519: SSS PM [uid:4][0413FC58]:   Mac-Address = aabb.cc00.6500
*Feb  7 18:58:24.519: SSS PM [uid:4][0413FC58]:   Final = 1 (YES)
*Feb  7 18:58:24.519: SSS PM [uid:4][0413FC58]:   Authen-Status = 1 (Unauthenticated)
*Feb  7 18:58:24.519: SSS PM [uid:4][0413FC58]:   Nasport = PPPoEoE: slot 0 adapter 0 port
 0
*Feb  7 18:58:24.519: SSS PM [uid:4][0413FC58]:   Session-Handle = 486539268 (1D000004)
*Feb 7 18:58:24.519: SSS PM [uid:4][0413FC58]: SM Policy invoke - Service Selection Request
*Feb  7 18:58:24.519: SSS PM [uid:4][0413FC58]: Access type IP
*Feb  7 18:58:24.519: SSS PM [uid:4][0413FC58]: Access type IP: final key
*Feb  7 18:58:24.519: SSS PM [uid:4][0413FC58]: Received Service Request
*Feb  7 18:58:24.519: SSS PM [uid:4][0413FC58]: Handling Authorization Check
*Feb  7 18:58:24.519: SSS PM [uid:4][0413FC58]: SIP [IP] can NOT provide more keys
*Feb  7 18:58:24.519: SSS PM [uid:4][0413FC58]: SIP [IP] can NOT provide more keys
*Feb  7 18:58:24.519: SSS PM [uid:4][0413FC58]: Handling Default Service
*Feb  7 18:58:24.519: SSS PM [uid:4][0413FC58]: Providing Service
*Feb  7 18:58:24.519: SSS PM [uid:4][0413FC58]: Policy reply - Local Terminate
```

```
*Feb  7 18:58:24.523: SSS PM [uid:4][0413FC58]: SM Policy invoke - Apply Config Success
*Feb  7 18:58:24.523: SSS PM [uid:4][0413FC58]: Handling Apply Config; SUCCESS
```

debug subscriber service

To display diagnostic information about the service profile database in an Intelligent Services Gateway (ISG), use the **debug subscriber service** command in privileged EXEC mode. To disable debugging, use the **no** form of this command.

debug subscriber service

no debug subscriber service

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(28)SB	This command was introduced.

Usage Guidelines

Use the **debug subscriber service** command to diagnose problems with service profiles or service policy maps.

Examples

The following example shows sample output for the **debug subscriber service** command. This output indicates that a service logon has occurred for the service "prep_service".

```
*Feb 7 18:52:31.067: SVM [prep_service]: needs downloading
*Feb 7 18:52:31.067: SVM [D6000000/prep_service]: allocated version 1
*Feb 7 18:52:31.067: SVM [D6000000/prep_service]: [8A000002]: client queued
*Feb 7 18:52:31.067: SVM [D6000000/prep_service]: [PM-Download:8A000002] locked 0->1
*Feb 7 18:52:31.067: SVM [D6000000/prep_service]: [AAA-Download:040DD9D0] locked 0->1
*Feb 7 18:52:31.127: SVM [D6000000/prep_service]: TC feature info found
*Feb 7 18:52:31.127: SVM [D0000001/prep_service]: added child
*Feb 7 18:52:31.127: SVM [D6000000/prep_service]: [TC-Child:040DD130] locked 0->1
*Feb 7 18:52:31.127: SVM [D0000001/CHILD/prep_service]: [TC-Parent:040DD1A8] locked 0->1
*Feb 7 18:52:31.127: SVM [D6000000/prep_service]: TC flow feature info not found
*Feb 7 18:52:31.127: SVM [D6000000/prep_service]: downloaded first version
*Feb 7 18:52:31.127: SVM [D6000000/prep_service]: [8A000002]: client download ok
*Feb 7 18:52:31.127: SVM [D6000000/prep_service]: [SVM-to-client-msg:8A000002] locked 0->1
*Feb 7 18:52:31.127: SVM [D6000000/prep_service]: [AAA-Download:040DD9D0] unlocked 1->0
*Feb 7 18:52:31.131: SVM [D6000000/prep_service]: alloc feature info
*Feb 7 18:52:31.131: SVM [D6000000/prep_service]: [SVM-Feature-Info:040E2E80] locked 0->1
*Feb 7 18:52:31.131: SVM [D6000000/prep_service]: has Policy info
*Feb 7 18:52:31.131: SVM [D6000000/prep_service]: [PM-Info:0416BAB0] locked 0->1
*Feb 7 18:52:31.131: SVM [D6000000/prep_service]: populated client
*Feb 7 18:52:31.131: SVM [D6000000/prep_service]: [PM-Download:8A000002] unlocked 1->0
*Feb 7 18:52:31.131: SVM [D6000000/prep_service]: [SVM-to-client-msg:8A000002] unlocked
1->0
*Feb 7 18:52:31.131: SVM [D6000000/prep_service]: [PM-Service:040E31E0] locked 0->1
*Feb 7 18:52:31.131: SVM [D0000001/CHILD/prep_service]: [SM-SIP-Apply:D0000001] locked 0->1
*Feb 7 18:52:31.131: SVM [D6000000/prep_service]: [FM-Bind:82000002] locked 0->1
*Feb 7 18:52:31.131: SVM [D6000000/prep_service]: [SVM-Feature-Info:040E2E80] unlocked 1->0
*Feb 7 18:52:31.139: SVM [D0000001/CHILD/prep_service]: alloc feature info
*Feb 7 18:52:31.139: SVM [D0000001/CHILD/prep_service]: [SVM-Feature-Info:040E2E80] locked
```

```
 0->1
*Feb 7 18:52:31.159: SVM [D0000001/CHILD/prep_service]: [FM-Bind:2C000003] locked 0->1
*Feb 7 18:52:31.159: SVM [D0000001/CHILD/prep_service]: [SVM-Feature-Info:040E2E80] unlocked
 1->0
*Feb 7 18:52:31.159: SVM [D0000001/CHILD/prep_service]: [SM-SIP-Apply:D0000001] unlocked
1->0
```

debug subscriber testing

To display diagnostic information for Intelligent Services Gateway (ISG) simulator testing, use the **debug subscriber testing** command in privileged EXEC mode. To disable debugging, use the **no** form of this command.

debug subscriber testing

no debug subscriber testing

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(28)SB	This command was introduced.

Examples

The following example shows the configuration of the **debug subscriber testing** command:

```
Router# debug subscriber testing
```

debug sw56

To display debugging information for switched 56K services, use the **debug sw56** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug sw56

no debug sw56

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
11.3T	This command was introduced.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

debug syscon perfdata

To display messages related to performance data collection, use the **debug syscon perfdata** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug syscon perfdata

no debug syscon perfdata

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

This command is primarily useful to your technical support representative.

Examples

The following is sample output from the **debug syscon perfdata** command. In this example, the CallFail poll group is configured and applied to shelf 1111. The system determines when the next polling cycle should occur and polls the shelf at the appropriate time. The data is stored in the file CallFail.891645120, and an older file is deleted.

```
Router# debug syscon perfdata
PERF: Applying 'CallFail' to shelf 1111
PERF: Setting up objects for SNMP polling: 'CallFail', shelf 1111
PERF: year hours mins secs msecs = 1998 15 11 1 5
PERF: Start 'CallFail' timer, next cycle in 0 mins, 59 secs
PERF: Timer event:  CallFail, 4 minutes
PERF: Polling 'CallFail', shelf 1111, pc 60AEFDF0
PERF: SNMP resp: Type 6, 'CallFail', shelf 1111, error_st 0
PERF: Logged polled data to disk0:/performance/shelf-1111/CallFail.891645120
PERF: Deleted disk0:/performance/shelf-1111/CallFail.891637469
```

debug syscon sdp

To display messages related to the Shelf Discovery Protocol (SDP), use the **debug syscon sdp** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug syscon sdp

no debug syscon sdp

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

Use this command to display information about SDP packets exchanged between the shelf and the system controller.

Examples

The following sample output from the **debug syscon sdp** command shows the system controller discovering a managed shelf. In the first few lines, the system controller receives a hello packet from shelf 99 at 172.23.66.106. The system controller responds with a hello packet. When the shelf sends another hello packet, the system controller resets the timer and sends another packet.

```
Syscon# debug syscon sdp
SYSCTLR: Hello packet received via UDP from 172.23.66.106
%SYSCTLR-6-SHELF_ADD: Shelf 99 discovered located at address 172.23.66.106
Hello packet sent to the RS located at 172.23.66.106
SYSCTLR: Hello packet received via UDP from 172.23.66.106
Timer for shelf 99 updated, shelf is alive
Hello packet sent to the RS located at 172.23.66.106
```

The following sample output from the **debug syscon sdp** command shows the shelf contacting the system controller. The shelf sends a hello packet to the system controller at 172.23.66.111. The system controller responds with the autoconfiguration commands. The remaining lines show the Hello packets were exchanged between the shelf and the system controller.

```
Shelf# debug syscon sdp
SYSCTLR: Hello packet sent to the SYSCTLR at 172.23.66.111
SYSCTLR: Command packet received from SYSCTLR
Feb 24 17:24:16.713: %SHELF-6-SYSCTLR_ESTABLISHED: Configured via system controller located
 at 172.23.66.111
SYSCTLR: Rcvd HELLO from SYSCTLR at 172.23.66.111
SYSCTLR: Hello packet sent to the SYSCTLR at 172.23.66.111
SYSCTLR: Rcvd HELLO from SYSCTLR at 172.23.66.111
```

debug syslog-server

To display information about the syslog server process, use the **debug syslog-server** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug syslog-server

no debug syslog-server

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

This command outputs a message every time the syslog server receives a message. It also displays information about subfile creation, removal, and renaming.

Use this command when subfiles are not being created as configured or data is not being written to subfiles. This command is also useful for detecting syslog file size mismatches.

Examples

The following is sample output from the **debug syslog-server** command. The sample output shows when the following command has been added to the configuration:

```
logging syslog-server 10 3 syslogs
```

This example shows the files being created. Use the **dir disk0:/syslogs.dir** command to display the contents of the newly created directory.

```
Router# debug syslog-server
SYSLOG_SERVER:Syslog file syslogs
SYSLOG_SERVER:Directory disk0:/syslogs.dir created.
SYSLOG_SERVER:Syslog file syslogs created successfully.
```

When a syslog message is received, the router checks to determine if the current file will be too large when the new data is added. In this example, two messages are added to the file.

```
SYSLOG_SERVER: Configured size : 10240 bytes
Current size : 0 bytes
Data size : 68 bytes
New size : 68 bytes
SYSLOG_SERVER: Wrote 68 bytes successfully.
SYSLOG_SERVER: Configured size : 10240 bytes
Current size : 68 bytes
Data size : 61 bytes
New size : 129 bytes
SYSLOG_SERVER: Wrote 61 bytes successfully.
```

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

Table 31: debug syslog-server Field Descriptions

Field	Description
Configured size	Maximum subfile size, as set in the **logging syslog-server** command.

Field	Description
Current size	Size of the current subfile before the new message is added.
Data size	Size of the syslog message.
New size	Size of the current subfile after the syslog message is added.

The following output indicates that the current file is too full to fit the next syslog message. The oldest subfile is removed, and the remaining files are renamed. A new file is created and opened for writing syslog messages.

```
SYSLOG_SERVER:Last archive subfile disk0:/syslogs.dir/syslogs.2 removed.
SYSLOG_SERVER: Subfile disk0:/syslogs.dir/syslogs.1 renamed as disk0:/syslogs.dir/syslogs.2.
SYSLOG_SERVER:subfile disk0:/syslogs.dir/syslogs.cur renamed as disk0:/syslogs.dir/syslogs.1.
SYSLOG_SERVER:Current subfile disk0:/syslogs.dir/syslogs.cur has been opened.
```

debug tacacs

To display information associated with TACACS, use the **debug tacacs**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug tacacs

no debug tacacs

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

TACACS is a distributed security system that secures networks against unauthorized access. Cisco supports TACACS under the authentication, authorization, and accounting (AAA) security system.

Use the **debug aaa authentication** command to get a high-level view of login activity. When TACACS is used on the router, you can use the **debug tacacs**command for more detailed debugging information.

Examples

The following is sample output from the **debug aaa authentication** command for a TACACS login attempt that was successful. The information indicates that TACACS+ is the authentication method used.

```
Router# debug aaa authentication
14:01:17: AAA/AUTHEN (567936829): Method=TACACS+
14:01:17: TAC+: send AUTHEN/CONT packet
14:01:17: TAC+ (567936829): received authen response status = PASS
14:01:17: AAA/AUTHEN (567936829): status = PASS
```

The following is sample output from the **debug tacacs**command for a TACACS login attempt that was successful, as indicated by the status PASS:

```
Router# debug tacacs
14:00:09: TAC+: Opening TCP/IP connection to 192.168.60.15 using source 10.116.0.79
14:00:09: TAC+: Sending TCP/IP packet number 383258052-1 to 192.168.60.15 (AUTHEN/START)
14:00:09: TAC+: Receiving TCP/IP packet number 383258052-2 from 192.168.60.15
14:00:09: TAC+ (383258052): received authen response status = GETUSER
14:00:10: TAC+: send AUTHEN/CONT packet
14:00:10: TAC+: Sending TCP/IP packet number 383258052-3 to 192.168.60.15 (AUTHEN/CONT)
14:00:10: TAC+: Receiving TCP/IP packet number 383258052-4 from 192.168.60.15
14:00:10: TAC+ (383258052): received authen response status = GETPASS
14:00:14: TAC+: send AUTHEN/CONT packet
14:00:14: TAC+: Sending TCP/IP packet number 383258052-5 to 192.168.60.15 (AUTHEN/CONT)
14:00:14: TAC+: Receiving TCP/IP packet number 383258052-6 from 192.168.60.15
14:00:14: TAC+ (383258052): received authen response status = PASS
14:00:14: TAC+: Closing TCP/IP connection to 192.168.60.15
```

The following is sample output from the **debug tacacs**command for a TACACS login attempt that was unsuccessful, as indicated by the status FAIL:

```
Router# debug tacacs
13:53:35: TAC+: Opening TCP/IP connection to 192.168.60.15 using source
192.48.0.79
13:53:35: TAC+: Sending TCP/IP packet number 416942312-1 to 192.168.60.15
(AUTHEN/START)
13:53:35: TAC+: Receiving TCP/IP packet number 416942312-2 from 192.168.60.15
13:53:35: TAC+ (416942312): received authen response status = GETUSER
```

```
13:53:37: TAC+: send AUTHEN/CONT packet
13:53:37: TAC+: Sending TCP/IP packet number 416942312-3 to 192.168.60.15
(AUTHEN/CONT)
13:53:37: TAC+: Receiving TCP/IP packet number 416942312-4 from 192.168.60.15
13:53:37: TAC+ (416942312): received authen response status = GETPASS
13:53:38: TAC+: send AUTHEN/CONT packet
13:53:38: TAC+: Sending TCP/IP packet number 416942312-5 to 192.168.60.15
(AUTHEN/CONT)
13:53:38: TAC+: Receiving TCP/IP packet number 416942312-6 from 192.168.60.15
13:53:38: TAC+ (416942312): received authen response status = FAIL
13:53:40: TAC+: Closing TCP/IP connection to 192.168.60.15
```

Related Commands

Command	Description
debug aaa accounting	Displays information on accountable events as they occur.
debug aaa authentication	Displays information on AAA/TACACS+ authentication.

debug tacacs events

To display information from the TACACS+ helper process, use the **debug tacacs events**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug tacacs events

no debug tacacs events

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

Use the **debug tacacs events**command only in response to a request from service personnel to collect data when a problem has been reported.

Caution Use the **debug tacacs events**command with caution because it can generate a substantial amount of output.

The TACACS protocol is used on routers to assist in managing user accounts. TACACS+ enhances the TACACS functionality by adding security features and cleanly separating out the authentication, authorization, and accounting (AAA) functionality.

Examples

The following is sample output from the **debug tacacs events** command. In this example, the opening and closing of a TCP connection to a TACACS+ server are shown, and the bytes read and written over the connection and the TCP status of the connection:

```
Router# debug tacacs events
%LINK-3-UPDOWN: Interface Async2, changed state to up
00:03:16: TAC+: Opening TCP/IP to 192.168.58.104/1049 timeout=15
00:03:16: TAC+: Opened TCP/IP handle 0x48A87C to 192.168.58.104/1049
00:03:16: TAC+: periodic timer started
00:03:16: TAC+: 192.168.58.104 req=3BD868 id=-1242409656 ver=193 handle=0x48A87C (ESTAB)
expire=14 AUTHEN/START/SENDAUTH/CHAP queued
00:03:17: TAC+: 192.168.58.104 ESTAB 3BD868 wrote 46 of 46 bytes
00:03:22: TAC+: 192.168.58.104 CLOSEWAIT read=12 wanted=12 alloc=12 got=12
00:03:22: TAC+: 192.168.58.104 CLOSEWAIT read=61 wanted=61 alloc=61 got=49
00:03:22: TAC+: 192.168.58.104 received 61 byte reply for 3BD868
00:03:22: TAC+: req=3BD868 id=-1242409656 ver=193 handle=0x48A87C (CLOSEWAIT) expire=9
AUTHEN/START/SENDAUTH/CHAP processed
00:03:22: TAC+: periodic timer stopped (queue empty)
00:03:22: TAC+: Closing TCP/IP 0x48A87C connection to 192.168.58.104/1049
00:03:22: TAC+: Opening TCP/IP to 192.168.58.104/1049 timeout=15
00:03:22: TAC+: Opened TCP/IP handle 0x489F08 to 192.168.58.104/1049
00:03:22: TAC+: periodic timer started
00:03:22: TAC+: 192.168.58.104 req=3BD868 id=299214410 ver=192 handle=0x489F08 (ESTAB)
expire=14 AUTHEN/START/SENDPASS/CHAP queued
00:03:23: TAC+: 192.168.58.104 ESTAB 3BD868 wrote 41 of 41 bytes
00:03:23: TAC+: 192.168.58.104 CLOSEWAIT read=12 wanted=12 alloc=12 got=12
00:03:23: TAC+: 192.168.58.104 CLOSEWAIT read=21 wanted=21 alloc=21 got=9
00:03:23: TAC+: 192.168.58.104 received 21 byte reply for 3BD868
00:03:23: TAC+: req=3BD868 id=299214410 ver=192 handle=0x489F08 (CLOSEWAIT) expire=13
AUTHEN/START/SENDPASS/CHAP processed
00:03:23: TAC+: periodic timer stopped (queue empty)
```

The TACACS messages are intended to be self-explanatory or for consumption by service personnel only. However, the messages shown are briefly explained in the following text.

The following message indicates that a TCP open request to host 192.168.58.104 on port 1049 will time out in 15 seconds if it gets no response:

```
00:03:16: TAC+: Opening TCP/IP to 192.168.58.104/1049 timeout=15
```

The following message indicates a successful open operation and provides the address of the internal TCP "handle" for this connection:

```
00:03:16: TAC+: Opened TCP/IP handle 0x48A87C to 192.168.58.104/1049
```

The following message indicates that a TACACS+ request has been queued:

```
00:03:16: TAC+: 192.168.58.104 req=3BD868 id=-1242409656 ver=193 handle=0x48A87C (ESTAB)
expire=14 AUTHEN/START/SENDAUTH/CHAP queued
```

The message identifies the following:

- Server that the request is destined for
- Internal address of the request
- TACACS+ ID of the request
- TACACS+ version number of the request
- Internal TCP handle the request uses (which will be zero for a single-connection server)
- TCP status of the connection--which is one of the following:
    - CLOSED
    - LISTEN
    - SYNSENT
    - SYNRCVD
    - ESTAB
    - FINWAIT1
    - FINWAIT2
    - CLOSEWAIT
    - LASTACK
    - CLOSING
    - TIMEWAIT
- Number of seconds until the request times out
- Request type

The following message indicates that all 46 bytes were written to address 192.168.58.104 for request 3BD868:

```
00:03:17: TAC+: 192.168.58.104 ESTAB 3BD868 wrote 46 of 46 bytes
```

The following message indicates that 12 bytes were read in reply to the request:

```
00:03:22: TAC+: 192.168.58.104 CLOSEWAIT read=12 wanted=12 alloc=12 got=12
```

The following message indicates that 49 more bytes were read, making a total of 61 bytes in all, which is all that was expected:

```
00:03:22: TAC+: 192.168.58.104 CLOSEWAIT read=61 wanted=61 alloc=61 got=49
```

The following message indicates that a complete 61-byte reply has been read and processed for request 3BD868:

```
00:03:22: TAC+: 192.168.58.104 received 61 byte reply for 3BD868 00:03:22: TAC+: req=3BD868
 id=-1242409656 ver=193 handle=0x48A87C (CLOSEWAIT) expire=9 AUTHEN/START/SENDAUTH/CHAP
processed
```

The following message indicates that the TACACS+ server helper process switched itself off when it had no more work to do:

```
00:03:22: TAC+: periodic timer stopped (queue empty)
```

Related Commands

Command	Description
debug aaa accounting	Displays information on accountable events as they occur.
debug aaa authentication	Displays information on AAA/TACACS+ authentication.
debug aaa authorization	Displays information on AAA/TACACS+ authorization.
debug sw56	Displays debugging information for switched 56K services.

debug tag-switching atm-cos

The **debug tag-switching atm-cos** command is replaced by the **debug mpls atm-cos** command. See the **debug mpls atm-cos** command for more information.

debug tag-switching atm-tdp api

The **debug tag-switching atm-tdp api** command is replaced by the **debug mpls atm-ldp api** command. See the **debug mpls atm-ldp api** command for more information.

debug tag-switching atm-tdp routes

The **debug tag-switching atm-tdp routes** command is replaced by the **debug mpls atm-ldp routes** command. See the **debug mpls atm-ldp routes** command for more information.

debug tag-switching atm-tdp states

The **debug tag-switching atm-tdp states** command is replaced by the **debug mpls atm-ldp states** command. See the **debug mpls atm-ldp states** command for more information.

debug tag-switching tdp advertisements

The **debug tag-switching tdp advertisement**s command is replaced by the **debug mpls ldp advertisements** command. See the **debug mpls ldp advertisements** command for more information.

debug tag-switching tdp bindings

The **debug tag-switching tdp bindings**command is replaced by the **debug mpls ldp bindings** command. See the **debug mpls ldp bindings** command for more information.

debug tag-switching tdp directed-neighbors

The **debug tag-switching tdp directed-neighbors**command is replaced by the **debug mpls ldp targeted-neighbors**command. See the **debug mpls ldp targeted-neighbors** command for more information.

debug tag-switching tdp peer state-machine

The **debug tag-switching tdp peer state-machine**command is replaced by the **debug mpls ldp peer state-machine**command. See the **debug mpls ldp peer state-machine** command for more information.

debug tag-switching tdp pies received

The **debug tag-switching tdp pies received** command is replaced by the **debug mpls ldp session io** command. See the **debug mpls ldp session io** command for more information.

debug tag-switching tdp pies sent

The **debug tag-switching tdp pies sent** command is replaced by the **debug mpls ldp messages**command. See the **debug mpls ldp messages** command for more information.

debug tag-switching tdp session io

The **debug tag-switching tdp session io**command is replaced by the **debug mpls ldp session io**command. See the **debug mpls ldp session io** command for more information

debug tag-switching tdp session state-machine

The **debug tag-switching tdp session state-machine** command is replaced by the **debug mpls ldp session state-machine** command. See the **debug mpls ldp session state-machine** command for more information.

debug tag-switching tdp transport connections

The **debug tag-switching tdp transport connections**command is replaced by the **debug mpls ldp tranport connections**command. See the **debug mpls ldp transport connections** command for more information.

debug tag-switching tdp transport events

The **debug tag-switching tdp transport events**command is replaced by the **debug mpls ldp tranport events**command. See the **debug mpls ldp transport events** command for more information.

debug tag-switching tdp transport timers

To print information about events that restart the “hold” timers that are part of the TDP discovery mechanism, use the **debug tag-switching tdp transport timers**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug tag-switching tdp transport timers

no debug tag-switching tdp transport timers

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(5)T	This command was introduced.
12.2(13)T	This command is no longer supported in Cisco IOS Mainline or Technology-based (T) releases. It may continue to appear in Cisco IOS 12.2S-family releases.

Usage Guidelines

TDP sessions are supported by data structures and state machines at three levels:

- Transport --The transport level establishes and maintains TCP connections used to support TDP sessions.
- Protocol --The protocol level implements the TDP session setup protocol. The construction and parsing of TDP PDUs and PIEs occur at this level.
- Tag distribution --The tag distribution level uses TDP sessions to exchange tags with TDP peer devices.

The **debug tag-switching tdp transport** command provides visibility of activity at the transport level, the **debug tag-switching tdp session** command at the protocol level, and the **debug tag-switching tdp peer**command at the tag distribution level.

Examples

The following is sample output from the **debug tag-switching tdp transport timers**command:

```
Router# debug tag-switching tdp transport timers
tdp: Start holding timer; adj 0x60D5BC10, 200.26.0.4
tdp: Start holding timer; adj 0x60EA9360, 10.105.0.9
tdp: Start holding timer; adj 0x60D5BC10, 200.26.0.4
tdp: Start holding timer; adj 0x60EA9360, 10.105.0.9
tdp: Start holding timer; adj 0x60D5BC10, 200.26.0.4
tdp: Start holding timer; adj 0x60EA9360, 10.105.0.9
```

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

Table 32: debug tag-switching tdp transport timers Field Descriptions

Field	Description
tdp	Identifies the source of the message as TDP.
adj 0xnnnnnnnn	Identifies the data structure used to represent the peer device at the transport level.
a.b.c.d	Network address of the peer device.

Related Commands

Command	Description
debug tag-switching tdp transport events	Prints information about the events related to the TDP peer discovery mechanism, which is used to determine the devices with which to establish TDP sessions.

debug tag-switching xtagatm cross-connect

The **debug tag-switching xtagatm cross-connect**command is replaced by the **debug mpls xtagatm cross-connect**command. See the **debug mpls xtagatm cross-connect** command for more information.

debug tag-switching xtagatm errors

The **debug tag-switching xtagatm errors**command is replaced by the **debug mpls xtagatm errors**command. See the **debug mpls xtagatm errors** command for more information.

debug tag-switching xtagatm events

The **debug tag-switching xtagatm events**command is replaced by the **debug mpls xtagatm events**command. See the **debug mpls xtagatm events** command for more information.

debug tag-switching xtagatm vc

The **debug tag-switching xtagatm vc**command is replaced by the **debug mpls xtagatm vc**command. See the **debug mpls xtagatm vc** command for more information.

debug tag-template event through debug voip application vxml

- debug tag-template event, page 254
- debug tarp events, page 255
- debug tarp packets, page 258
- debug tbridge virtual-port, page 260
- debug tcam_mgr, page 261
- debug tccs signaling, page 263
- debug tdm, page 265
- debug telco-return msg, page 268
- debug telnet, page 269
- debug text-to-fax, page 272
- debug tftp, page 273
- debug tgrep error, page 274
- debug tgrep events, page 276
- debug tgrep fsm, page 278
- debug tgrep io, page 281
- debug tgrep messages, page 284
- debug tgrep msgdump, page 286
- debug tgrep timer-event, page 289
- debug tgrep timers, page 293
- debug tgrep tripr, page 295
- debug tgrm, page 299
- debug tiff reader, page 303
- debug tiff writer, page 305

- debug time-range ipc, page 306
- debug token ring, page 307
- debug topology, page 310
- debug track, page 312
- debug trifecta, page 314
- debug tsp, page 316
- debug tunnel rbscp, page 318
- debug tunnel route-via, page 321
- debug txconn all, page 322
- debug txconn appc, page 324
- debug txconn config, page 326
- debug txconn data, page 328
- debug txconn event, page 330
- debug txconn tcp, page 332
- debug txconn timer, page 334
- debug udptn, page 336
- debug usb driver, page 337
- debug user-group, page 340
- debug v120 event, page 342
- debug v120 packet, page 343
- debug vfi checkpoint, page 345
- debug vg-anylan, page 347
- debug video vicm, page 349
- debug vlan packet, page 351
- debug voice aaa asnl, page 352
- debug voice all, page 353
- debug voice cp, page 354
- debug voice dsp crash-dump, page 356
- debug voice eecm, page 358
- debug voice enum, page 360
- debug voice fastpath, page 363
- debug voice h221, page 367
- debug voice h324, page 370

- debug voice mlpp, page 375
- debug voice protocol, page 377
- debug voice register errors, page 379
- debug voice register events, page 381
- debug voice signaling, page 385
- debug voice source-group, page 387
- debug voice statistics, page 389
- debug voice tdsm, page 394
- debug voice translation, page 396
- debug voice uri, page 399
- debug voice vofr, page 402
- debug voip aaa, page 404
- debug voip ais, page 405
- debug voip application, page 407
- debug voip application stcapp all, page 414
- debug voip application stcapp buffer-history, page 421
- debug voip application stcapp error, page 423
- debug voip application stcapp events, page 425
- debug voip application stcapp functions, page 427
- debug voip application stcapp port, page 429
- debug voip application vxml, page 437

debug tag-template event

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

debug tag-template event

no debug tag-template event

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging is turned off.

Command Modes

Privileged EXEC (#)

Command History

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

Examples

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

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

Related Commands

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

debug tarp events

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

debug tarp events

no debug tarp events

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

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

Examples

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

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

The table below describes the significant fields shown in display.

Table 33: debug tarp events Field Descriptions--tarp resolve Command

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

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

Related Commands

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

debug tarp packets

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

debug tarp packets

no debug tarp packets

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

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

Examples

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

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

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

Table 34: debug tarp packets Field Descriptions--tarp query Command

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

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

Related Commands

Command	Modification
debug tarp events	Displays information on TARP activity.

debug tbridge virtual-port

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

debug tbridge virtual-port

no debug tbridge virtual-port

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

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

Examples

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

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

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

Table 35: debug tbridge virtual-port Field Descriptions

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

debug tcam_mgr

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

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

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

Syntax Description

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

Command Default

No default behavior or values.

Command Modes

Privileged EXEC

Command History

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

Usage Guidelines

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

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

Examples

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

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

Related Commands

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

debug tccs signaling

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

debug tccs signaling

no debug tccs signaling

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled.

Command Modes

Privileged EXEC

Command History

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

Usage Guidelines

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

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

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

Examples

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

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

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

debug tdm

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

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

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

Syntax Description

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

Command Modes

Privileged EXEC

Usage Guidelines

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

Examples

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

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

Examples

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

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

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

debug telco-return msg

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

debug cable telco-return msg

no debug cable telco-return msg

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging for telco-return messages is not enabled.

Command Modes

Privileged EXEC

Command History

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

Examples

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

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

debug telnet

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

debug telnet

no debug telnet

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

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

Examples

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

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

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

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

Table 36: debug telnet Field Descriptions

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

Related Commands

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

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

debug text-to-fax

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

debug text-to-fax

no debug text-to-fax

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

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

Examples

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

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

debug tftp

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

debug tftp

no debug tftp

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Examples

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

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

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

Table 37: debug tftp Field Descriptions

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

debug tgrep error

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

debug tgrep error

no debug tgrep error

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(1)	This command was introduced.

Usage Guidelines

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

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

Examples

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

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

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

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

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

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

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

Related Commands

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

debug tgrep events

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

debug tgrep events

no debug tgrep events

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(1)	This command was introduced.

Usage Guidelines

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

Examples

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

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

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

Table 38: debug tgrep events Field Descriptions

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

Related Commands

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

debug tgrep fsm

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

debug tgrep fsm

no debug tgrep fsm

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(1)	This command was introduced.

Usage Guidelines

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

Examples

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

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

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

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

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

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

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

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

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

The prefix is shown here in hex format.

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

Related Commands

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

debug tgrep io

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

debug tgrep io

no debug tgrep io

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(1)	This command was introduced.

Usage Guidelines

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

Examples

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

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

At this point, the connection is initiated.

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

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

Errors begin to appear here.

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

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

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

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

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

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

Related Commands

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

debug tgrep messages

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

debug tgrep messages

no debug tgrep messages

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(1)	This command was introduced.

Usage Guidelines

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

Examples

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

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

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

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

Related Commands

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

debug tgrep msgdump

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

debug tgrep msgdump

no debug tgrep msgdump

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(1)	This command was introduced.

Usage Guidelines

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

Examples

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

```
tgrep-gw-1-02#Received an KEEPALIVE NBR:14.1.1.210
++++++++++++++++++++++++++++++++++++
 TMSG datagramstart : 0x69188648, 150 bytes
++++++++++++++++++++++++++++++++++++++
 0x0 0xFFFFFF96 0x2 0x0 0x1
 0x0 0x0 0x0 0x2 0x0
 0x9 0x0 0x5 0x0 0x0
 0x0 0x3 0x6D 0x63 0x69
 0x0 0x3 0x0 0x6 0x0
 0x0 0x4 0xFFFFFFD2 0x0 0x0
 0x0 0x4 0x0 0x6 0x2
 0x1 0x0 0x0 0x4 0xFFFFFFD2
 0x0 0x5 0x0 0x6 0x2
 0x1 0x0 0x0 0x4 0xFFFFFFD2
 0x0 0x7 0x0 0x4 0x0
 0x0 0x0 0x5 0x0 0xF
 0x0 0x4 0x0 0x0 0x0
 0x16 0x0 0x10 0x0 0x4
 0x0 0x0 0x0 0x17 0x0
 0x11 0x0 0x8 0x0 0x0
 0x0 0x74 0x0 0x0 0x0
 0x7B 0x0 0x12 0x0 0x3C
 0x0 0x4 0x31 0x31 0x32
 0x38 0x0 0x4 0x31 0x31
 0x32 0x37 0x0 0x4 0x31
 0x31 0x32 0x36 0x0 0x4
 0x31 0x31 0x32 0x35 0x0
 0x4 0x31 0x31 0x32 0x34
 0x0 0x4 0x31 0x31 0x32
 0x33 0x0 0x4 0x31 0x31
 0x32 0x32 0x0 0x5 0x39
 0x39 0x39 0x39 0x39 0x0
 0x9 0x31 0x32 0x33 0x34
 0x35 0x36 0x37 0x38 0x39
```

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

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

Related Commands

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

debug tgrep timer-event

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

debug tgrep timer-event

no debug tgrep timer-event

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(1)	This command was introduced.

Usage Guidelines

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

Examples

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

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

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

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

Timeouts are again reported until the next event.

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

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

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

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

Related Commands

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

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

debug tgrep timers

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

debug tgrep timers

no debug tgrep timers

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(1)	This command was introduced.

Usage Guidelines

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

Examples

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

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

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

Table 39: debug tgrep timers Field Descriptions

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

Related Commands

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

debug tgrep tripr

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

debug tgrep tripr

no debug tgrep tripr

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(1)	This command was introduced.

Usage Guidelines

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

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

Examples

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

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

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

The table below describes the significant fields in the display.

Table 40: debug tgrep tripr Field Descriptions

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

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

Related Commands

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

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

debug tgrm

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

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

no debug tgrm

Syntax Description

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

Command Default

Debugging is not enabled.

Command Modes

Privileged EXEC

Command History

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

Usage Guidelines

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

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

Examples

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

```
Router# debug tgrm all

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

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

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

Table 41: debug tgrm all Field Descriptions

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

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

debug tiff reader

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

debug tiff reader

no debug tiff reader

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled.

Command Modes

Privileged EXEC

Command History

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

Examples

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

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

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

Related Commands

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

debug tiff writer

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

debug tiff writer

debug tiff writer

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled.

Command Modes

Privileged EXEC

Command History

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

Examples

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

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

Related Commands

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

debug time-range ipc

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

debug time-range ipc

no debug time-range ipc

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values.

Command Modes

Privileged EXEC

Command History

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

Examples

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

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

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

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

Related Commands

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

debug token ring

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

debug token ring

no debug token ring

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

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

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

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

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

Examples

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

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

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

Table 42: debug token ring Field Descriptions

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

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

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

Table 43: debug token ring Field Descriptions

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

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

Table 44: debug token ring Field Descriptions

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

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

debug topology

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

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

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

Syntax Description

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

Command Default

Debugging output for topology related events is disabled.

Command Modes

Privileged EXEC (#)

Command History

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

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

Examples

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

```
Device# debug topology ha
```

Related Commands

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

debug track

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

debug track

no debug track

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC (#)

Command History

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

Usage Guidelines

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

Examples

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

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

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

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

Related Commands

Command	Description
show track	Displays tracking information.

debug trifecta

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

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

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

Syntax Description

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

Command Default

None

Command Modes

Global configuration

Command History

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

Usage Guidelines

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

Examples

The sample output for the command is as follows:

```
debug trifecta all
Router-sp#debug trifecta all
TRIFECTA severe debugging is on
TRIFECTA major debugging is on
TRIFECTA minor debugging is on
TRIFECTA debug debugging is on
Router-sp#
Jan 24 20:06:34.463 IST: SP: process_trifecta_msgs(): Polling TIMEINFO slot 2.
Jan 24 20:06:34.463 IST: SP: send_time_zone_info_to_x86(): sending time zone info to X86
slot 2
Jan 24 20:07:24.467 IST: SP: process_trifecta_msgs(): Polling TIMEINFO slot 2.
Jan 24 20:07:24.467 IST: SP: send_time_zone_info_to_x86(): sending time zone info to X86
slot 2
Jan 24 20:08:14.471 IST: SP: process_trifecta_msgs(): Polling TIMEINFO slot 2.
Jan 24 20:08:14.471 IST: SP: send_time_zone_info_to_x86(): sending time zone info to X86
slot 2
Jan 24 20:09:04.475 IST: SP: process_trifecta_msgs(): Polling TIMEINFO slot 2.
Jan 24 20:09:04.475 IST: SP: send_time_zone_info_to_x86(): sending time zone info to X86
slot 2
Jan 24 20:09:54.479 IST: SP: process_trifecta_msgs(): Polling TIMEINFO slot 2.
Jan 24 20:09:54.479 IST: SP: send_time_zone_info_to_x86(): sending time zone info to X86
slot 2
```

debug tsp

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

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

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

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

Syntax Description

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

Command Default

Disabled.

Command Modes

Privileged EXEC

Command History

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

Examples

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

```
Router# debug tsp all

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

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

Related Commands

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

debug tunnel rbscp

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

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

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

Syntax Description

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

Command Modes

Privileged EXEC

Command History

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

Usage Guidelines

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

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

Examples

The following example turns on debugging messages about RBSCP messages:

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

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

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

Table 45: debug tunnel rbscp msg Field Descriptions

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

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

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

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

Table 46: debug tunnel rbscp rto Field Descriptions

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

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

Related Commands

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

debug tunnel route-via

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

debug tunnel route-via

no debug tunnel route-via

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

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

Examples

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

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

Related Commands

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

debug txconn all

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

debug txconn all

no debug txconn all

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging is not enabled for the txconn subsystem.

Command Modes

Privileged EXEC

Command History

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

Examples

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

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

Related Commands

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

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

debug txconn appc

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

debug txconn appc

no debug txconn appc

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging is not enabled for the txconn subsystem.

Command Modes

Privileged EXEC

Command History

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

Examples

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

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

Related Commands

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

debug txconn config

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

debug txconn config

no debug txconn config

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging is not enabled for the txconn subsystem.

Command Modes

Privileged EXEC

Command History

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

Examples

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

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

Related Commands

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

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

debug txconn data

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

debug txconn data

no debug txconn data

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging is not enabled for the txconn subsystem.

Command Modes

Privileged EXEC

Command History

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

Examples

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

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

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

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

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

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

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

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

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

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

Related Commands

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

debug txconn event

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

debug txconn event

no debug txconn event

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging is not enabled for the txconn subsystem.

Command Modes

Privileged EXEC

Command History

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

Examples

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

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

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

Related Commands

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

debug txconn tcp

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

debug txconn tcp

no debug txconn tcp

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging is not enabled for the txconn subsystem.

Command Modes

Privileged EXEC

Command History

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

Examples

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

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

Related Commands

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

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

debug txconn timer

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

debug txconn timer

no debug txconn timer

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging is not enabled for the txconn subsystem.

Command Modes

Privileged EXEC

Command History

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

Examples

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

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

Related Commands

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

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

debug udptn

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

debug udptn

no debug udptn

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled.

Command Modes

Privileged EXEC

Command History

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

Examples

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

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

Related Commands

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

debug usb driver

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

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

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

Syntax Description

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

Command Default

None

Command Modes

Privileged EXEC

Command History

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

Usage Guidelines

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

Examples

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

```
Router# debug usb driver transfer bulk

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

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

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

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

debug user-group

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

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

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

Syntax Description

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

Command Modes

Privileged EXEC (#)

Command History

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

Usage Guidelines

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

Examples

The following example configures debugging for user-group additions:

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

Related Commands

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

debug v120 event

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

debug v120 event

no debug v120 event

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

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

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

Examples

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

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

Related Commands

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

debug v120 packet

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

debug v120 packet

no debug v120 packet

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

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

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

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

Examples

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

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

The table below describes the significant fields in the display.

Table 47: debug v120 packet Field Descriptions

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

Related Commands

Command	Description
debug tarp events	Displays information on TARP activity.

debug vfi checkpoint

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

debug vfi checkpoint

no debug vfi checkpoint

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC (#)

Command History

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

Examples

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

```
Router# debug vfi checkpoint

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

Related Commands

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

debug vg-anylan

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

debug vg-anylan

no debug vg-anylan

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

Thiscommand could create a substantial amount of command output.

Examples

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

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

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

Table 48: debug vg-anylan Message Descriptions

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

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

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

debug video vicm

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

debug video vicm

no debug video vicm

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

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

Examples

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

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

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

debug vlan packet

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

debug vlan packet

no debug vlan packet

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

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

Examples

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

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

debug voice aaa asnl

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

debug voice aaa asnl

no debug voice aaa asnl

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging of AAA ASNL is not enabled.

Command Modes

Privileged EXEC

Command History

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

Usage Guidelines

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

Examples

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

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

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

Table 49: debug voice aaa asnl Field Descriptions

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

debug voice all

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

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

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

Syntax Description

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

Command Modes

Privileged EXEC

Usage Guidelines

This command is valid on the Cisco MC3810 only.

Examples

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

Related Commands

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

debug voice cp

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

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

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

Syntax Description

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

Command Modes

Privileged EXEC

Usage Guidelines

This command is valid on the Cisco MC3810 only.

Examples

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

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

Related Commands

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

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

debug voice dsp crash-dump

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

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

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

Syntax Description

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

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

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

Usage Guidelines

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

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

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

Examples

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

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

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

*Mar 8 03:42:21.513:status cleared done, dsp 14 slot 1

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

```
Router# undebug all
```

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

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

Related Commands

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

debug voice eecm

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

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

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

Syntax Description

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

Command Modes

Privileged EXEC

Usage Guidelines

This command is valid on the Cisco MC3810 only.

Examples

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

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

Related Commands

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

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

debug voice enum

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

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

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

Syntax Description

detail	Displays detailed output.
summary	Displays summary output.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

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

Usage Guidelines

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

Examples

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

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

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

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

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

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

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

Table 50: debug voice enum Field Descriptions

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

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

Related Commands

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

debug voice fastpath

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

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

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

Syntax Description

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

Command Default

Voice fastpath debugging does not occur.

Command Modes

Privileged EXEC (#)

Command History

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

Usage Guidelines

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

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

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

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

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

Examples

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

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

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

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

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

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

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

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

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

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

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

Related Commands

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

debug voice h221

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

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

no debug voice h221

Syntax Description

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

Command Modes

Privileged EXEC

Command History

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

Usage Guidelines

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

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

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

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

Table 51: Indexes and Categories for the debug voice h221 individual command

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

Examples

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

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

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

Related Commands

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

debug voice h324

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

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

Syntax Description

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

Command Modes

Privileged EXEC (#)

Command History

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

Usage Guidelines

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

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

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

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

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

Table 52: Index Numbers and Descriptions for the debug voice h324 Command

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

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

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

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

debug voice mlpp

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

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

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

Syntax Description

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

Command Modes

Privileged EXEC (#)

Command History

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

Usage Guidelines

This command enables debugging for MLPP events.

Examples

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

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

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

```
Router# debug voice mlpp

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

Related Commands

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

debug voice protocol

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

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

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

Syntax Description

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

Command Modes

Privileged EXEC

This command is valid on the Cisco MC3810 only.

Usage Guidelines

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

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

Examples

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

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

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

Related Commands

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

debug voice register errors

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

debug voice register errors

no debug voice register errors

Syntax Description

This command has no arguments or keywords

Command Default

Disabled

Command Modes

Privileged EXEC mode

Command History

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

Usage Guidelines

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

Examples

Examples

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

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

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

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

Examples

The following is sample output from this command:

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

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

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

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

Table 53: debug voice register errors Field Descriptions

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

Related Commands

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

debug voice register events

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

debug voice register events

no debug voice register events

Syntax Description

This command has no arguments or keywords

Command Default

Disabled

Command Modes

Privileged EXEC mode

Command History

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

Usage Guidelines

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

Registration activity includes matching of pools, registration creation, and automatic creation of dial

peers. For more details and error conditions, you can use the debug voice register errors command.

Cisco Unified CME

The following example shows output from this command:

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

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

Cisco Unified SIP SRST

The following is sample output from this command:

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

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

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

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

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

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

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

Table 54: debug voice register events Field Descriptions

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

Related Commands

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

debug voice signaling

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

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

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

Syntax Description

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

Command Modes

Privileged EXEC

Usage Guidelines

This command is valid on the Cisco MC3810 only.

Examples

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

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

Related Commands

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

debug voice source-group

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

debug voice source-group

no debug voice source-group

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

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

Usage Guidelines

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

Examples

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

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

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

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

Table 55: debug voice source-group Field Descriptions

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

Related Commands

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

debug voice statistics

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

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

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

Syntax Description

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

Command Modes

Privileged EXEC

Command History

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

Examples

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

```
Router# debug voice statistics accounting

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

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

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

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

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

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

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

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

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

Related Commands

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

debug voice tdsm

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

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

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

Syntax Description

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

Command Modes

Privileged EXEC

Usage Guidelines

This command is valid on the Cisco MC3810 only.

Examples

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

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

Related Commands

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

debug voice translation

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

debug voice translation

no debug voice translation

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

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

Usage Guidelines

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

Examples

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

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

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

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

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

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

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

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

Table 56: debug voice translation Field Descriptions

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

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

Related Commands

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

debug voice uri

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

debug voice uri

no debug voice uri

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

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

Usage Guidelines

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

Examples

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

```
Router# debug voice uri
```

Examples

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

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

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

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

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

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

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

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

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

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

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

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

Examples

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

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

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

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

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

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

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

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

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

Related Commands

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

debug voice vofr

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

debug voice vofr

no debug voice vofr

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

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

Usage Guidelines

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

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

Examples

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

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

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

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

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

Related Commands

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

debug voip aaa

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

debug voip aaa

no debug voip aaa

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

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

Examples

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

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

debug voip ais

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

debug voip ais

no debug voip ais

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

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

Examples

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

```
Router# debug voip ais

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

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

Related Commands

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

debug voip application

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

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

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

Syntax Description

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

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

Command Modes

Privileged EXEC

Command History

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

Usage Guidelines

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

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

Examples

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

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

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

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

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

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

```
Router# debug voip application session

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

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

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

```
Router# debug voip application supplementary-service

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

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

Table 57: debug voip application Field Descriptions

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

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

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

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

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

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

Related Commands

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

debug voip application stcapp all

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

debug voip application stcapp all

no debug voip application stcapp all

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC (#)

Command History

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

Usage Guidelines

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

Examples

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

```
Router# debug voip application stcapp al
l
Port 2/0 performs a hook flash to activate the drop last call feature.
Mar  3 20:41:07.022: 2/0   : stcapp_screen_api_event
Mar  3 20:41:07.022: 2/0   :     event:STCAPP_CC_EV_CALL_FEATURE_HOOKFLASH received.
Mar  3 20:41:07.022: 2/0   : stcapp_screen_call_feature_hookflash
Mar  3 20:41:07.022: 2/0   :     lcb->num_ccbs=2, lcb->mode=CALL_TRANSFER(1),
lcb->state=ACTIVE (8)
Mar  3 20:41:07.022: 2/0   :     lcb->fm_mode=CALL_FM_NONE(0),
Mar  3 20:41:07.022: 2/0   :    ACTIVE call_state=5, lcb->mode=CALL_TRANSFER(1), lcb->state=8
Mar  3 20:41:07.022: 2/0   :     STANDBY call_state=8, lcb->mode=CALL_TRANSFER(1),
lcb->state=8
Mar  3 20:41:07.022: 2/0   :     enter FM: set event id to STCAPP_EV_FEATURE_MODE(143)
Mar  3 20:41:07.022: 2/0   : ==> Received event:STCAPP_EV_FEATURE_MODE for CallId: 63
Mar  3 20:41:07.022: 2/0   :     Call State:ACTIVE
Mar  3 20:41:07.022: 2/0   : stcapp_feature_mode_eh
```

The following line shows a message to generate feature tone.

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

Router# debug voip application stcapp all
```

The following lines show the voice gateway beginning call setup:

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

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

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

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

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

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

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

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

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

Related Commands

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

debug voip application stcapp buffer-history

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

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

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

Syntax Description

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

Command Modes

Privileged EXEC

Command History

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

Usage Guidelines

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

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

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

Examples

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

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

Related Commands

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

debug voip application stcapp error

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

debug voip application stcapp error

no debug voip application stcapp error

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

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

Usage Guidelines

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

Examples

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

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

Related Commands

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

debug voip application stcapp events

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

debug voip application stcapp events

no debug voip application stcapp events

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

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

Usage Guidelines

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

Examples

The following example displays call teardown and disconnect events:

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

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

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

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

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

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

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

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

Related Commands

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

debug voip application stcapp functions

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

debug voip application stcapp functions

no debug voip application stcapp functions

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

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

Usage Guidelines

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

Examples

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

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

Related Commands

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

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

debug voip application stcapp port

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

debug voip application stcapp port *port-number*

no debug voip application stcapp port *port-number*

Syntax Description

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

Command Modes

Privileged EXEC (#)

Command History

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

Usage Guidelines

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

Examples

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

The following lines show the call being disconnected.

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

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

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

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

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

The following lines show the call transitioning to off hook.

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

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

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

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

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

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

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

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

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

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

The following lines show modem relay parameters:

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

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

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

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

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

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

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

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

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

```
*Jan  9 06:57:24.407: 1/0/1: stcapp_set_up_voice_leg
*Jan  9 06:57:24.407: 1/0/1:     Sending ccIFCallSetupRequest for voice leg
*Jan  9 06:57:24.407: 1/0/1:     ccIFCallSetRequest returned voice call id:22.
CdPN:7702CgPN:7701
*Jan  9 06:57:24.407: 1/0/1:     No state change
```

The next lines show the application invoking the ringing state.

```
*Jan  9 06:57:24.407: ==> Received event:STCAPP_DC_EV_DEVICE_SET_RINGER
(evID:DC_EV_DEVICE_SET_RINGER)
*Jan  9 06:57:24.407: 1/0/1:     Call State:RINGIN
*Jan  9 06:57:24.407: 1/0/1: stcapp_set_ringer_eh
*Jan  9 06:57:24.407: 1/0/1: stcapp_set_ring_mode
*Jan  9 06:57:24.407: 1/0/1:     SCCP ring mode:2
*Jan  9 06:57:24.407: 1/0/1:     Invoking Feature:12. Mode:0 for callid:22
*Jan  9 06:57:24.407: 1/0/1:     No state change
```

Related Commands

Command	Description
debug voip application stcapp all	Displays debug output for all the debug commands for the STCAPP compiled into one display.
debug voip application stcapp error	Displays STCAPP error log contents.
debug voip application stcapp events	Traces STCAPP call flow events.
debug voip application stcapp functions	Displays STCAPP entry and exit function calls for all voice ports.

debug voip application vxml

To troubleshoot a VoiceXML application, use the **debug voip application vxml** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voip application vxml [**all**| **application**| **background**| **default**| **error** [**call [informational]**| **software [informational]**]| **event**| **function**| **grammar**| **gtd**| **inout**| **log**| **puts**| **ssml**| **trace**| **warning**]

no debug voip application vxml

Syntax Description

all	(Optional) Displays all VoiceXML debugging messages.
application	(Optional) Displays VoiceXML application states information.
background	(Optional) Displays VoiceXML background messages.
default	(Optional) Displays output for all of the following keywords: • **application** • **background** • **error** • **event** • **gtd** • **inout** • **puts** • **trace** • **warning** This option is also available if no keywords are added.
error	(Optional) Displays VoiceXML errors.
call	(Optional) Displays call processing errors.
informational	(Optional) Displays minor errors and major errors. Without the **informational** keyword, only major errors are displayed.
software	(Optional) Displays software errors.

event	(Optional) Displays VoiceXML asynchronous events.
function	(Optional) Displays VoiceXML functions.
grammar	(Optional) Enables syntax checking of XML grammar by the VoiceXML interpreter and displays syntax debugging messages.
gtd	(Optional) Displays VoiceXML generic transparency descriptors.
inout	(Optional) Displays VoiceXML in/out functions.
log	(Optional) Displays the results of the VoiceXML <log> tag.
puts	(Optional) Displays the results of VoiceXML <cisco-puts> and <cisco-putvar> tags. **Note** In Cisco IOS Release 12.4(6th)T and later releases, the **puts** keyword is obsolete. Use the **log** keyword instead.
ssml	(Optional) Enables syntax checking of Speech Synthesis Markup Language (SSML) by the VoiceXML interpreter and displays syntax debugging messages.
trace	(Optional) Displays a trace of all activities for the current VoiceXML document.
warning	(Optional) Displays VoiceXML warning messages.

Command Default

Debugging is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(8)T	This command replaces the **debug vxml** command.
12.4(15)T	The **puts** keyword was obsoleted. The **log** keyword was added to replace it.

Usage Guidelines

If this debug encounters a fetch failure when using an HTTP interface, an **error.badfetch.http**.*response code* message is displayed. The values for the response code are shown in the table below.

Table 58: error.badfetch.http Response Codes

Response Code	Description
0	No response from HTTP server
400	Bad request
401	Unauthorized
402	Payment required
403	Forbidden
404	Not found
405	Method not allowed
406	Not acceptable
407	Proxy authentication required
408	Request timeout
409	Conflict
410	Gone
411	Length required
412	Precondition failed
413	Request entity too large
414	Request-URI too large
415	Unsupported media type
416	Requested range not satisfiable
417	Expectation failed
500	Internal server error
501	Not implemented

Response Code	Description
502	Bad gateway
503	Service unavailable
504	Gateway timeout
505	Version not supported

Examples

The following is sample output from the **debug voip application vxml all** command if there is an HTTP badfetch error call:

```
Router# debug voip application vxml all
Aug  7 04:53:03.003: //-1/000000000000/VAPP:/vapp_evt_handler:
   State VAPP_ACTIVE got event CC_EV_CALL_SETUP_IND
Aug  7 04:53:03.003: //-1/000000000000/VAPP:/vapp_driver:
   pInterp[6383BA48]:
Aug  7 04:53:03.003: //-1/000000000000/VAPP:/vapp_driver:
   evtID: 29 vapp record state: 0
Aug  7 04:53:03.003: //-1/000000000000/VAPP:/vapp_evt_setup:
Aug  7 04:53:03.003: //-1//VAPP:/vapp_incoming_callblock:
Aug  7 04:53:03.003: vapp_incoming_callblock:
```

Before the incoming call block data comes in, the CallEntry ID is -1, which indicates that the call leg had not been identified. In the next excerpt, the call leg is shown as 1 and the GUID is also assigned.

```
Aug  7 04:53:03.003: //1/71E56a9AF8002/VAPP:/vapp_evt_setup:
   VXML call. GTD should be saved
Aug  7 04:53:03.003: //1/71E569AF8002/VAPP:/vapp_load_or_run_script:
Aug  7 04:53:03.003: //1/71E569AF8002/VAPP:/vapp_load_or_run_script:
```

The next excerpt show script-specific information.

```
Aug  7 04:53:03.007: The VXML Script with len=1519 starts:
-------------------------------------
<?xml version="3.0" encoding="iso-8859-1"?>
<vxml version="3.0">
 <form id = "transfer_me">
 <catch event="telephone.disconnect.transfer">
 </catch>
<var name="phone_num" expr="5550100"/>
        <v
```

The CallEntry ID becomes 0 in the following excerpt, which indicates that the output is from the application server, not a call leg.

```
Aug  7 04:53:03.007: //0//VXML:/vxml_start_element_handler: Enter
Aug  7 04:53:03.007: //0//VXML:/vxml_start_element_handler: Exit
Aug  7 04:53:03.007: //0//VXML:/vxml_character_data: Enter
Aug  7 04:53:03.007: //0//VXML:/vxml_character_data:
   at line 888: length <=0, exit
.
.
.
Aug  7 04:53:03.051: //0//VXML:/vxml_end_element_handler: Enter
Aug  7 04:53:03.051: //0//VXML:/vxml_end_element_handler: Exit
Aug  7 04:53:03.051: //0//VXML:/vxml_parse:

Aug  7 04:53:03.051: vxml_parse: XML_Parse success err=0
Aug  7 04:53:03.051: //0//VXML:/vxml_session_delete:

Aug  7 04:53:03.051: vxml_session_delete:mem_mgr_mempool_free: mempool=NULL
Aug  7 04:53:03.051: //-1//VXML:/vxml_create:
   enter url=tftp://dirt/jkuo/vxml/xfer.nosound.vxml tree_handle=63282BDC
   return_handle_add=63C84F80
```

In the following excerpt, the call with the GUID 71E569AF8002 is again being tracked as the application session is initiated.

```
Aug  7 04:53:03.083: //1/71E569AF8002/VXML:/vxml_offramp_mailhdrs_get:

Aug  7 04:53:03.087: //1/71E569AF8002/VXML:/vxml_create_gtd_sess_vars:
   Created object chain for com.cisco.signal.gtdlist
Aug  7 04:53:03.087: //1/71E569AF8002/VXML:/vxml_jse_add_gtd_obj_to_list:
   Sig-event name = setup_indication, gtd-len = 140, gtd-buf =
IAM,
PRN,isdn*,,,
USI,rate,c,s,c,1
USI,lay1,ulaw
TMR,00
CPN,34,,4,52950
CPC,09
FCI,,,,,,,y,
GCI,71e569af6b5511d4800200014232e6a8
Aug  7 04:53:03.087: //1/71E569AF8002/VXML:/vxml_jse_add_gtd_obj_to_list:
   gtd_obj for sig-event [setup_indication] added to session/shadow
   var array [0x63826914]
Aug  7 04:53:03.087: //1/71E569AF8002/VXML:/vxml_create: Exit
Aug  7 04:53:03.087: //1/71E569AF8002/VXML:/vxml_start:
   vxmlhandle=6372E9BC vapphandle=6383BA48 status=0 async_status=0
Aug  7 04:53:03.087: //1/71E569AF8002/VXML:/vxml_vxml_proc:
<vxml> URI(abs):tftp://dirt/jkuo/vxml/xfer.nosound.vxml
   scheme=tftp
   host=dirt
   path=/jkuo/vxml/xfer.nosound.vxml
   base= URI(abs):tftp://dirt/jkuo/vxml/xfer.nosound.vxml
   scheme=tftp
   host=dirt
   path=/jkuo/vxml/xfer.nosound.vxml lang=none version=3.0
Aug  7 04:53:03.087: //1/71E569AF8002/VXML:/vxml_form_proc:
Aug  7 04:53:03.087:  <form>: id=transfer_me   scope=dialog
```

In the following excerpt, the phone number of the caller is shown:

```
Aug  7 04:53:03.087: vxml_form_init current scope: dialog
 <var>: namep=phone_num expr=5550100
Aug  7 04:53:03.091: //1/71E569AF8002/VXML:/vxml_expr_eval:
   expr=var phone_num=5550100
 <var>: namep=mydur
Aug  7 04:53:03.095: //1/71E569AF8002/VXML:/vxml_expr_eval:
   expr=var mydur
vxml_counter_reset:
Aug  7 04:53:03.095: //1/71E569AF8002/VXML:/vxml_formitem_select:
   Status=VXML_STATUS_OK,
Aug  7 04:53:03.095: //1/71E569AF8002/VXML:/vxml_formitem_select:
    AsyncStatus=VXML_STATUS_OK
Aug  7 04:53:03.099: //1/71E569AF8002/VXML:/vxml_transfer_proc:
  <transfer>:
```

In the following excerpts, the attributes of the incoming phone call are shown:

```
Aug  7 04:53:03.099: //1/71E569AF8002/VXML:/vxml_item_attrs_proc:
   name=mycall dest_expr='phone://'+ phone_num bridge=1 connecttimeout=50 maxtime=50
desttype=-1 destplan=-1 anitype=-1 aniplan=-1 anipi=-1, anisi=-1 rdn_exprp='phone://' +
4085550111 rdntype=2 rdnplan=1 rdnpi=0, rdnsi=3, redirectreason=0
```

The next several excerpts show the initialization and playing of audio prompts. When troubleshooting voice applications, ensure that all your assigned prompts play when required.

```
Aug  7 04:53:03.099: //1/71E569AF8002/VXML:/vxml_play_prompts: Enter
Aug  7 04:53:03.099: //1/71E569AF8002/VXML:/vxml_prompt_proc: Enter
Aug  7 04:53:03.099: //1/71E569AF8002/VXML:/vxml_prompt_proc:
   <prompt>:(default_prompt) bargein=1 count=1 typeaheadflush=0
Aug  7 04:53:03.099: //1/71E569AF8002/VXML:/vxml_audio_proc: Enter
Aug  7 04:53:03.099: //1/71E569AF8002/VXML:/vxml_audio_proc:
    <audio>: URI(abs):http://px1-sun/nosound.au
   scheme=http
   host=px1-sun
   path=/nosound.au caching=fast fetchhint=invalid fetchtimeout=0
Aug  7 04:53:03.099: //1/71E569AF8002/VXML:/vxml_audio_proc: Exit
Aug  7 04:53:03.099: //1/71E569AF8002/VXML:/vxml_vapp_media_play:
   bargein=1 timeout=0 typeaheadflush=0 vcr=0 rate=0
Aug  7 04:53:03.099: //1/71E569AF8002/VXML:/vxml_vapp_media_play:
   str=http://px1-sun/nosound.au cachable=1 timeout0
Aug  7 04:53:03.099: //1/71E569AF8002/VAPP:/vapp_media_play:
Aug  7 04:53:03.099: //1/71E569AF8002/VAPP:/vapp_media_play:
   prompt=http://px1-sun/nosound.au:
Aug  7 04:53:03.687: //1/71E569AF8002/VXML:/vxml_vapp_media_play: Exit
Aug  7 04:53:03.687: //1/71E569AF8002/VXML:/vxml_prompt_proc: Exit
Aug  7 04:53:03.687: //1/71E569AF8002/VXML:/vxml_play_prompts: Exit
```

The next several excerpts indicate that something is wrong with the XML form to which data is being written:

```
Aug  7 04:53:03.699: //1/71E569AF8002/VXML:/vxml_transfer_proc:
   GTD not present in <transfer>
Aug  7 04:53:03.699: //1/71E569AF8002/VXML:/vxml_transfer_proc: Exit
Aug  7 04:53:03.699: //1/71E569AF8002/VXML:/vxml_elem_proc:
   at line 8521: Status not OK, exit
Aug  7 04:53:03.703: //1/71E569AF8002/VXML:/vxml_formitem_select:
   at line 4651: Status not OK, exit
Aug  7 04:53:03.703: //1/71E569AF8002/VXML:/vxml_form_proc:
   at line 4791: Status not OK, exit
Aug  7 04:53:03.703: //1/71E569AF8002/VXML:/vxml_elem_proc:
   at line 8521: Status not OK, exit
Aug  7 04:53:03.703: //1/71E569AF8002/VXML:/vxml_vxml_proc:
   at line 8703: Status not OK, exit
Aug  7 04:53:03.703: //1/71E569AF8002/VXML:/vxml_load_immediate_done:
   sidp->status=180000000
Aug  7 04:53:03.703: //1/71E569AF8002/VAPP:/vapp_checksessionstate:
Aug  7 04:53:03.703: //1/71E569AF8002/VAPP:/vapp_checkifdone:
Aug  7 04:53:03.703: //1/71E569AF8002/VAPP:/vapp_driver: Exit
Aug  7 04:53:03.703: //1/71E569AF8002/VAPP:/vapp_evt_handler:
   State VAPP_ACTIVE got event CC_EV_CALL_MODIFY_DONE
Aug  7 04:53:03.703: //1/71E569AF8002/VAPP:/vapp_driver:
   pInterp[6383BA48]:
Aug  7 04:53:03.703: //1/71E569AF8002/VAPP:/vapp_driver:
   evtID: 37 vapp record state: 0
Aug  7 04:53:03.703: //1/71E569AF8002/VAPP:/vapp_checksessionstate:
Aug  7 04:53:03.703: //1/71E569AF8002/VAPP:/vapp_checkifdone:
Aug  7 04:53:03.703: //1/71E569AF8002/VAPP:/vapp_driver: Exit
Aug  7 04:53:03.735: //1/71E569AF8002/VAPP:/vapp_evt_handler:
   State VAPP_ACTIVE got event MSW_EV_SYNTHESIZER
Aug  7 04:53:03.735: //1/71E569AF8002/VAPP:/vapp_driver:
   pInterp[6383BA48]:
Aug  7 04:53:03.735: //1/71E569AF8002/VAPP:/vapp_driver:
   evtID: 84 vapp record state: 0
Aug  7 04:53:03.735: //1/71E569AF8002/VAPP:/vapp_play_done:
   evID=84 reason=8, protocol=2, status_code=404, dur=-1, rate=0
Aug  7 04:53:03.735: //1/71E569AF8002/VXML:/vxml_media_done:
   status 2 async_status 180000000 duration=-1 rate=0
Aug  7 04:53:03.735: //1/71E569AF8002/VXML:/vxml_save_lastprompt_info:
Aug  7 04:53:03.735: //1/71E569AF8002/VXML:/vxml_bind_lastprompt:
Aug  7 04:53:03.735: //1/71E569AF8002/VXML:/vxml_vapp_vcr_control_disable:
Aug  7 04:53:03.735: //1/71E569AF8002/VAPP:/vapp_notify_play_done:
Aug  7 04:53:03.735: //1/71E569AF8002/VAPP:/vapp_notify_play_done: Exit
```

After checking the status of the application, the router finds a badfetch error, which indicates that a VoiceXML form was not found. See the table above for a description of error.badfetch.http response codes.

```
Aug  7 04:53:03.735: //1/71E569AF8002/VXML:/vxml_media_done:
C
ALL_ERROR: : fail with vapp error 2, protocol_status_code=404
Aug  7 04:53:03.735: //1/71E569AF8002/VXML:/vxml_media_done:
CALL_ERROR: : *** error.badfetch.http.404 event is thrown
Aug  7 04:53:03.739: //1/71E569AF8002/VXML:/vxml_event_driver:
Aug  7 04:53:03.739: //1/71E569AF8002/VXML:/vxml_event_proc:
   <event>: event=error.badfetch.http.404 status=0
Aug  7 04:53:03.739: //1/71E569AF8002/VXML:/vxml_default_event_handler:
   use default event handler
Aug  7 04:53:03.739: //1/71E569AF8002/VAPP:/vapp_session_exit_event_name:
   Exit Event error.badfetch.http.404
Aug  7 04:53:03.739: //1/71E569AF8002/VXML:/vxml_load_immediate_done:
   sidp->status=10
Aug  7 04:53:03.739: //1/71E569AF8002/VXML:/vxml_start:
   vxmlhandle=6372E9BC vapphandle=6383BA48 status=0 async_status=10
Aug  7 04:53:03.739: //1/71E569AF8002/VXML:/vxml_vapp_terminate:
   vapp_status=0 ref_count 0
Aug  7 04:53:03.739: //1/71E569AF8002/VAPP:/vapp_terminate:
Aug  7 04:53:03.739: //1/71E569AF8002/VXML:/vxml_destroy: Enter
Aug  7 04:53:03.739: //-1//VXML:/vxml_gtd_delete_callback:
   New ref-count = 0
```

Related Commands

Command	Description
debug condition application voice	Displays debugging messages for only the specified VoiceXML application.
debug http client	Displays debugging messages for the HTTP client.
debug voip ivr	Displays debugging messages for VoIP IVR interactions.

debug voip avlist through debug vpm signaling

- debug voip application lpcor, page 448
- debug voip avlist, page 450
- debug voip ccapi, page 451
- debug voip ccapi error, page 467
- debug voip ccapi inout, page 468
- debug voip confmsp, page 477
- debug voip dcapi, page 480
- debug voip dialpeer, page 482
- debug voip dsm, page 486
- debug voip dsmp, page 492
- debug voip dspapi, page 500
- debug voip dump-file-acct, page 507
- debug voip eddri, page 509
- debug voip enum, page 511
- debug voip event-log, page 514
- debug voip fastpath, page 516
- debug voip fileacct, page 520
- debug voip fpi call-rate , page 522
- debug voip h221, page 523
- debug voip h324, page 526
- debug voip hpi, page 531
- debug voip ipipgw, page 538
- debug voip ivr, page 542
- debug voip ivr all, page 543

- debug voip ivr applib, page 544
- debug voip ivr callsetup, page 545
- debug voip ivr digitcollect, page 546
- debug voip ivr dynamic, page 547
- debug voip ivr error, page 548
- debug voip ivr redirect, page 549
- debug voip ivr script, page 550
- debug voip ivr settlement, page 551
- debug voip ivr states, page 552
- debug voip ivr supplementary-service, page 553
- debug voip ivr tclcommands, page 554
- debug voip lpcor, page 555
- debug voip profile fax, page 557
- debug voip profile help, page 561
- debug voip profile modem, page 564
- debug voip profile voice, page 567
- debug voip rawmsg, page 570
- debug voip rtcp, page 573
- debug voip rtp, page 575
- debug voip settlement all, page 578
- debug voip settlement enter, page 580
- debug voip settlement error, page 581
- debug voip settlement exit, page 586
- debug voip settlement misc, page 587
- debug voip settlement network, page 588
- debug voip settlement security, page 591
- debug voip settlement ssl, page 592
- debug voip settlement transaction, page 594
- debug voip tsp, page 595
- debug voip vtsp, page 601
- debug voip xcodemsp, page 612
- debug vpdn, page 614
- debug vpdn pppoe-data, page 632

- debug vpdn pppoe-error, page 635
- debug vpdn pppoe-events, page 638
- debug vpdn pppoe-packet, page 640
- debug vpdn redundancy, page 642
- debug vpm all, page 649
- debug vpm dsp, page 651
- debug vpm error, page 652
- debug vpm port, page 654
- debug vpm signal, page 656
- debug vpm signaling, page 657

debug voip application lpcor

To enable debugging of the logical partitioning class of restriction (LPCOR) application system, use the **debug voip application lpcor** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voip application lpcor

no debug voip application lpcor

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
15.0(1)XA	This command was introduced.
15.1(1)T	This command was integrated into Cisco IOS Release 15.1(1)T.

Examples

The following is sample output from the **debug voip application lpcor**command for a call between two phones that was blocked by LPCOR policy validation:

```
Router# debug voip application lpcor
voip application AFW lpcor debugging is on CME#
*Jun 24 11:24:58.115: //44//Dest:/DestOutboundCallUsingPeer: Save Lpcor Index 1 to
Interworking Leg
*Jun 24 11:24:58.119: //44//Dest:/DestProcessLPCOR: Peer 20002 Source Callid 44 CallType 0
 *Jun 24 11:24:58.119: //44//Dest:/DestProcessLPCOR: lpcor source index(1) target index (2)
 ret_cause=63
*Jun 24 11:24:58.119: //44//Dest:/DestSetup: lpcor block with peerTag 20002
```

Related Commands

Command	Description
debug ephone lpcor	Displays debugging information for calls using the LPCOR feature.
debug voip lpcor	Displays debugging information for the LPCOR feature.
show voice lpcor policy	Displays the LPCOR policy for the specified resource group.
voice lpcor enable	Enables LPCOR functionality on the Cisco Unified CME router.

Command	Description
voice lpcor policy	Creates a LPCOR policy for a resource group.

debug voip avlist

To troubleshoot the attribute value list (AVLIST) contents, use the **debug voip avlist** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voip avlist [**all**| **default**| **detail**| **error** [**call [informational]**| **software [informational]**]| **inout**]

no debug voip avlist

Syntax Description

all	(Optional) Displays all AVLIST debugging messages.
default	(Optional) Displays AVLIST error and inout information. This option also runs if no keywords are added.
detail	(Optional) Displays AVLIST background messages.
error	(Optional) Displays AVLIST error messages.
call	(Optional) Displays call processing errors.
informational	(Optional) Displays minor errors and major errors. Without the **informational** keyword, only major errors are displayed.
software	(Optional) Displays software errors.
inout	(Optional) Displays AVLIST in/out functions.

Command Default

Debugging is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(8)T	This command was introduced.

Usage Guidelines

The **debug voip avlist** command does not support call debug filtering.

Examples

Output is primarily used by TAC.

debug voip ccapi

To troubleshoot the call control application programming interface (CCAPI) contents, use the **debug voip ccapi** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voip ccapi [**all**| **default**| **detail**| **error** [**call [informational]**| **software [informational]**]| **individual** *range*| **inout**| **function**| **protoheaders**| **service**]

no debug voip ccapi

Syntax Description

all	(Optional) Displays all CCAPI debugging messages.
default	(Optional) Displays CCAPI error and inout information. This option also runs if no keywords are added.
detail	(Optional) Displays CCAPI background messages.
error	(Optional) Displays CCAPI error messages. The **debug voip ccapi error** command traces the error logs in the call control API. Error logs are generated during normal call processing, if there are insufficient resources, or if there are problems in the underlying network-specific code, the higher call session application, or the call control API itself. This debug command shows error events or unexpected behavior in system software. Usually no events will be generated.
call	(Optional) Displays call processing errors.
informational	(Optional) Displays minor errors and major errors. Without the **informational** keyword, only major errors are displayed.
software	(Optional) Displays software errors.
individual	(Optional) Enables individual CCAPI debug outputs.
range	For the **individual** keyword, the range is an integer value from 1 to 146. For specific range values, see the table below.

inout	(Optional) Displays CCAPI in/out functions. The **debug voip ccapi inout** command traces the execution path through the call control API, which serves as the interface between the call session application and the underlying network-specific software. You can use the output from this command to understand how calls are being handled by the router. This command shows how a call flows through the system. Using this debug level, you can see the call setup and teardown operations performed on both the telephony and network call legs.
function	(Optional) Displays CCAPI function tracing.
protoheaders	(Optional) Displays CCAPI protocol headers passing information.
service	(Optional) Logs debug messages that are not call related.

Table 59: CCAPI Individual Debug Values

Value	CCAPI Debug Function
1	CC_IDMSG_API_DISPLAY_IES
2	CC_IDMSG_SETUP_IND_COMM_2
3	CC_IDMSG_SETUP_IND_COMM_3
4	CC_IDMSG_SETUP_IND_COMM_4
5	CC_IDMSG_ALERT_IND_5
6	CC_IDMSG_ALERT_IND_6
7	CC_IDMSG_CONNECT_IND_7
8	CC_IDMSG_CONNECT_IND_8
9	CC_IDMSG_RECONNECT_IND_9
10	CC_IDMSG_DISCONNECTED_IND_10
11	CC_IDMSG_DISCONNECTED_IND_11
12	CC_IDMSG_DISCONNECTED_IND_12

Value	CCAPI Debug Function
13	CC_IDMSG_DISCONNECT_DONE_IND_13
14	CC_IDMSG_DISCONNECT_DONE_IND_14
15	CC_IDMSG_DISCONNECT_DONE_IND_15
16	CC_IDMSG_PRE_DISC_CAUSE_16
17	CC_IDMSG_PRE_DISC_CAUSE_17
18	CC_IDMSG_DIGIT_BEGIN_IND_18
19	CC_IDMSG_DIGIT_END_IND_19
20	CC_IDMSG_DIGIT_END_IND_20
21	CC_IDMSG_DIGIT_END_NO_TERM_21
22	CC_IDMSG_TONE_IND_22
23	CC_IDMSG_FEATURE_IND_23
24	CC_IDMSG_MODIFY_DONE_IND_24
25	CC_IDMSG_MODIFY_MODE_DONE_IND_25
26	CC_IDMSG_INBAND_MSG_RCVD_IND_26
27	CC_IDMSG_INBAND_MSG_DONE_IND_27
28	CC_IDMSG_UPD_CALL_INFO_IND_28
29	CC_IDMSG_GEN_NTK_ALERT_EVENT_29
30	CC_IDMSG_VOICE_MODE_EVENT_30
31	CC_IDMSG_VOICE_MODE_EVENT_31
32	CC_IDMSG_DIALING_COMPLETE_IND_32
33	CC_IDMSG_DIGITS_DONE_IND_33
34	CC_IDMSG_DIGITS_DONE_IND_34
35	CC_IDMSG_VBD_XMIT_DONE_IND_35
36	CC_IDMSG_FWD_SETUP_IND_36

Value	CCAPI Debug Function
37	CC_IDMSG_RSVP_DONE_IND_37
38	CC_IDMSG_AUDIT_RSP_IND_38
39	CC_IDMSG_XFR_STATUS_IND_39
40	CC_IDMSG_XFR_STATUS_IND_40
41	CC_IDMSG_XFR_DONE_IND_41
42	CC_IDMSG_XFR_DONE_IND_42
43	CC_IDMSG_XFR_DONE_IND_43
44	CC_IDMSG_TGT_CID_ACTIVE_RCD_44
45	CC_IDMSG_MODIFY_MEDIA_IND_45
46	CC_IDMSG_MODIFY_MEDIA_ACK_IND_46
47	CC_IDMSG_MODIFY_MEDIA_REJ_IND_47
48	CC_IDMSG_MODEM_CALL_START_IND_48
49	CC_IDMSG_MODEM_CALL_DONE_IND_49
50	CC_IDMSG_ACCT_STATUS_IND_50
51	CC_IDMSG_NW_STATUS_IND_51
52	CC_IDMSG_DESTINFO_IND_52
53	CC_IDMSG_LOOPBACK_DONE_IND_53
54	CC_IDMSG_RT_PACKET_STATS_IND_54
55	CC_IDMSG_CUT_PROGRESS_IND_55
56	CC_IDMSG_CUT_PROGRESS_IND_56
57	CC_IDMSG_PROCEEDING_IND_57
58	CC_IDMSG_FACILITY_IND_58
59	CC_IDMSG_INFO_IND_59
60	CC_IDMSG_PROGRESS_IND_60

Value	CCAPI Debug Function
61	CC_IDMSG_USERINFO_IND_61
62	CC_IDMSG_DISC_PROG_IND_62
63	CC_IDMSG_DISC_PROG_IND_63
64	CC_IDMSG_PING_DONE_IND_64
65	CC_IDMSG_COT_TEST_DONE_IND_65
66	CC_IDMSG_PROCESS_DONE_IND_66
67	CC_IDMSG_ASSOCIATED_IND_67
68	CC_IDMSG_SUSPEND_IND_68
69	CC_IDMSG_SUSPEND_ACK_IND_69
70	CC_IDMSG_SUSPEND_REJ_IND_70
71	CC_IDMSG_RESUME_IND_71
72	CC_IDMSG_RESUME_ACK_IND_72
73	CC_IDMSG_RESUME_REJ_IND_73
74	CC_IDMSG_IF_SETUP_REQ_PRIV_74
75	CC_IDMSG_IF_SETUP_REQ_PRIV_75
76	CC_IDMSG_IF_ALLOCATE_DSP_76
77	CC_IDMSG_CONNECT_77
78	CC_IDMSG_CONNECT_78
79	CC_IDMSG_PING_79
80	CC_IDMSG_DISCONNECT_80
81	CC_IDMSG_DISCONNECT_81
82	CC_IDMSG_DISCONNECT_82
83	CC_IDMSG_ALERT_83
84	CC_IDMSG_ALERT_84

Value	CCAPI Debug Function
85	CC_IDMSG_CUT_PROGRESS_85
86	CC_IDMSG_CUT_PROGRESS_86
87	CC_IDMSG_CUT_PROGRESS_87
88	CC_IDMSG_DISC_PROG_88
89	CC_IDMSG_DISC_PROG_89
90	CC_IDMSG_SET_PEER_90
91	CC_IDMSG_SET_PEER_91
92	CC_IDMSG_PROCEEDING_92
93	CC_IDMSG_SETUP_REQ_93
94	CC_IDMSG_SETUP_REQ_94
95	CC_IDMSG_SETUP_REQ_95
96	CC_IDMSG_SETUP_REQ_96
97	CC_IDMSG_SETUP_REQ_97
98	CC_IDMSG_SETUP_REQ_98
99	CC_IDMSG_SETUP_REQ_99
100	CC_IDMSG_SETUP_REQ_100
101	CC_IDMSG_SETUP_REQ_101
102	CC_IDMSG_SETUP_ACK_102
103	CC_IDMSG_FACILITY_103
104	CC_IDMSG_TRANSFER_REQ_104
105	CC_IDMSG_GET_CONSULT_ID_105
106	CC_IDMSG_FORWARD_TO_106
107	CC_IDMSG_INFO_107
108	CC_IDMSG_NOTIFY_108

Value	CCAPI Debug Function
109	CC_IDMSG_PROGRESS_109
110	CC_IDMSG_PRE_DISC_110
111	CC_IDMSG_PRE_DISC_111
112	CC_IDMSG_USER_INFO_112
113	CC_IDMSG_MODIFY_113
114	CC_IDMSG_DIGIT_114
115	CC_IDMSG_DIGIT_DIAL_115
116	CC_IDMSG_DIGIT_DIAL_STOP_116
117	CC_IDMSG_FEATURE_117
118	CC_IDMSG_FEATURE_ENABLE_118
119	CC_IDMSG_ASSOCIATE_STREAM_119
120	CC_IDMSG_ASSOCIATE_STREAM_120
121	CC_IDMSG_DISASSOCIATE_STREAM_121
122	CC_IDMSG_DISASSOCIATE_STREAM_122
123	CC_IDMSG_GENERATE_TONE_INFO_123
124	CC_IDMSG_SET_DIGIT_TIMEOUTS_124
125	CC_IDMSG_SET_DIGIT_TIMEOUTS_125
126	CC_IDMSG_SUSPEND_126
127	CC_IDMSG_SUSPEND_ACK_127
128	CC_IDMSG_SUSPEND_REJ_128
129	CC_IDMSG_RESUME_129
130	CC_IDMSG_RESUME_ACK_130
131	CC_IDMSG_RESUME_REJ_131
132	CC_IDMSG_UPDATE_REDIRECT_NUM_132

Value	CCAPI Debug Function
133	CC_IDMSG_BABBLER_AUDIT_133
134	CC_IDMSG_CONFERENCE_CREATE_134
135	CC_IDMSG_CONFERENCE_CREATE_135
136	CC_IDMSG_CONFERENCE_CREATE_136
137	CC_IDMSG_CONFERENCE_DESTROY_137
138	CC_IDMSG_CONFERENCE_DESTROY_138
139	CC_IDMSG_CONFERENCE_DESTROY_139
140	CC_IDMSG_LOOPBACK_140
141	CC_IDMSG_COT_TEST_141
142	CC_IDMSG_HANDOFF_142
143	CC_IDMSG_APP_RETURN_143
144	CC_IDMSG_T38_FAX_START_144
145	CC_IDMSG_T38_FAX_DONE_145
146	CC_IDMSG_CALL_PREEMPT_IND_146

Command Default

Debugging is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
11.3(6)NA2	This command was introduced.
12.2(11)T	This command was implemented on the following platforms: Cisco 2600 series, Cisco 3620, Cisco 3660, Cisco AS5350, Cisco AS5400, Cisco AS5850, Cisco AS5300, Cisco AS5800, and Cisco MC3810.
12.3(8)T	The **all**, **default**, **detail**, **call**, **informational**, **software**, **individual**, **function**, **protoheaders**, and **service** keywords were added.

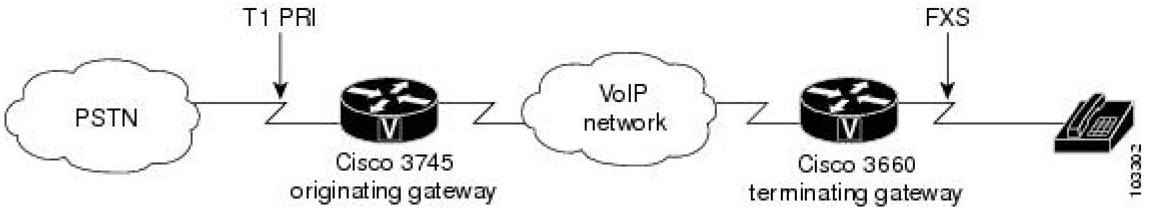
Release	Modification
12.4(4)XC	The range for the **individual** keyword was extended to 146, to include logs for call preemption indication information.
12.4(9)T	This command was integrated into Cisco IOS Release 12.4(9)T.

Examples

The following examples show output for variations of the **debug voip ccapi** command:

For these examples, the topology shown in the figure below is used.

Figure 1: Network Topology for debug voip ccapi Output Examples


T1 PRI
FXS
PSTN
VoIP network
Cisco 3745 originating gateway
Cisco 3660 terminating gateway
103302

Examples

```
Router# debug voip ccapi detail
voip ccapi detail debugging is on
Router#
*Apr 18 20:35:35.779: //-1/ABCE697D8005/CCAPI/cc_api_call_setup_ind_common:
   Interface Type=13, Protocol=0
*Apr 18 20:35:35.779: //-1/ABCE697D8005/CCAPI/ccCheckClipClir:
   Calling Party Number Is User Provided
*Apr 18 20:35:35.779: //11/xxxxxxxxxxxx/CCAPI/cc_insert_call_entry:
   Total Call Count=0, Call Entry(Call Count On=FALSE, Incoming Call=TRUE)
*Apr 18 20:35:35.779: //11/xxxxxxxxxxxx/CCAPI/cc_insert_call_entry:
   Total Call Count=1
```

The following event shows that the CallEntry ID 11 is used for the incoming call leg.

```
*Apr 18 20:35:35.779: //11/ABCE697D8005/CCAPI/cc_insert_guid_pod_entry:
   Incoming=TRUE, Call Id=11
*Apr 18 20:35:35.779: //11/ABCE697D8005/CCAPI/cc_setupind_registration_lookup:
   Matching Parameters; Called Number=83103, Call Transfer Consult Id=
*Apr 18 20:35:35.779: //11/ABCE697D8005/CCAPI/cc_setupind_registration_lookup:
   No Matching Node
*Apr 18 20:35:35.779: //11/ABCE697D8005/CCAPI/ccCheckClipClir:
   Calling Party Number Is User Provided
*Apr 18 20:35:35.779: //12/xxxxxxxxxxxx/CCAPI/cc_insert_call_entry:
   Total Call Count=1, Call Entry(Call Count On=FALSE, Incoming Call=FALSE)
```

The following event shows that the incoming call leg with CallEntry ID 11 is bound to the outgoing call leg with CallEntry ID 12.

```
*Apr 18 20:35:35.779: //11/ABCE697D8005/CCAPI/cc_peer_bind:
   Bind=TRUE, Binder Call Id=11, Bindee Call Id=12
```

The next event shows that CallEntry ID 12 is used for the outgoing call leg.

```
*Apr 18 20:35:35.779: //12/ABCE697D8005/CCAPI/cc_insert_guid_pod_entry:
   Incoming=FALSE, Call Id=12
*Apr 18 20:35:35.779: //-1/xxxxxxxxxxxx/CCAPI/cc_api_supported_data:
   data_mode=0x10082
```

The next event shows an IP address for a remote device on the outgoing call leg, which indicates that this is the VoIP call leg.

```
*Apr 18 20:35:35.779: //12/ABCE697D8005/CCAPI/cc_incr_if_call_volume:
   Remote IP Address=172.16.13.81, Hwidb=FastEthernet0/0
*Apr 18 20:35:35.779: //12/ABCE697D8005/CCAPI/cc_incr_if_call_volume:
   Total Call Count=1, Voip Call Count=1, MMoip Call Count=0
*Apr 18 20:35:35.795: //11/ABCE697D8005/CCAPI/ccCallGetContext:
   Context=0x652C0168, Call Id=11
*Apr 18 20:36:31.419: //11/ABCE697D8005/CCAPI/ccCallDisconnect:
   Start Calling Accounting;
   Call Entry(Incoming=TRUE)
*Apr 18 20:36:31.419: //11/ABCE697D8005/CCAPI/ccCallDisconnect:
   Cause Value=16, Call Entry(Disconnect Cause=16)
*Apr 18 20:36:31.419: //11/ABCE697D8005/CCAPI/ccCallDisconnect:
   Call Entry(Disconnect Cause=16)
```

At this point, the CallEntry ID changes as the call accounting process begins. The accounting data is sent over the outgoing call leg. The GUID, which identifies the unique call, remains the same.

```
*Apr 18 20:36:31.419: //12/ABCE697D8005/CCAPI/ccCallDisconnect:
   Start Calling Accounting;
   Call Entry(Incoming=FALSE)
*Apr 18 20:36:31.419: //12/ABCE697D8005/CCAPI/ccCallDisconnect:
   Cause Value=16, Call Entry(Disconnect Cause=0)
```

The change of the CallEntry ID indicates that the call is using the incoming call leg, which is the POTS call leg in this case.

```
*Apr 18 20:36:31.423: //11/ABCE697D8005/CCAPI/ccCallGetVoipFlag:
   Data Bitmask=0x1, Call Id=11
*Apr 18 20:36:31.423: //11/ABCE697D8005/CCAPI/ccCallGetVoipFlag:
   Flag=FALSE
*Apr 18 20:36:31.423: //11/ABCE697D8005/CCAPI/cc_delete_guid_pod_entry:
   Incoming=TRUE
*Apr 18 20:36:31.423: //11/ABCE697D8005/CCAPI/cc_delete_call_entry:
   ccFreeRawMsgInfo=0x63FF8198
*Apr 18 20:36:31.423: //11/ABCE697D8005/CCAPI/cc_delete_call_entry:
   Total Call Count=1, Call Entry(Call Count On=FALSE, Incoming Call=TRUE)
*Apr 18 20:36:31.423: //11/ABCE697D8005/CCAPI/cc_delete_call_entry:
   Total Call Count=0
*Apr 18 20:36:31.423: //11/ABCE697D8005/CCAPI/cc_delete_call_entry:
   Deleting profileTable[0x64F44700]
```

The next line shows the impairment calculation. This is the only CCAPI debug command that shows impairment.

```
*Apr 18 20:36:31.423: //-1/ABCE697D8005/CCAPI/g113_calculate_impairment:
   (delay=91(ms), loss=0%), Io=0 Iq=0 Idte=0 Idd=2 Ie=10 Itot=12
*Apr 18 20:36:31.423: //12/ABCE697D8005/CCAPI/ccCallGetVoipFlag:
   Data Bitmask=0x1, Call Id=12
*Apr 18 20:36:31.423: //12/ABCE697D8005/CCAPI/ccCallGetVoipFlag:
   Flag=FALSE
*Apr 18 20:36:31.423: //12/ABCE697D8005/CCAPI/cc_decr_if_call_volume:
   Remote IP Address=172.16.13.81, Hwidb=FastEthernet0/0
*Apr 18 20:36:31.423: //12/ABCE697D8005/CCAPI/cc_decr_if_call_volume:
   Total Call Count=0, Voip Call Count=0, MMoip Call Count=0
*Apr 18 20:36:31.423: //12/ABCE697D8005/CCAPI/cc_delete_guid_pod_entry:
   Incoming=FALSE
*Apr 18 20:36:31.423: //12/ABCE697D8005/CCAPI/cc_delete_call_entry:
   Total Call Count=0, Call Entry(Call Count On=FALSE, Incoming Call=FALSE)
*Apr 18 20:36:31.423: //12/ABCE697D8005/CCAPI/cc_delete_call_entry:
   Deleting profileTable[0x652E3310]
*Apr 18 20:36:31.427: //12/xxxxxxxxxxxx/CCAPI/cc_get_call_entry:
   Call Entry Is Not Found
```

Examples

```
Router# debug voip ccapi detail
voip ccapi detail debugging is on
Router#
```

```
*May  1 18:58:26.251: //-1/xxxxxxxxxxxx/CCAPI/cc_api_supported_data:
   data_mode=0x10082
*May  1 18:58:26.255: //8/xxxxxxxxxxxx/CCAPI/cc_get_call_entry:
   Call Entry Is Not Found
*May  1 18:58:26.255: //-1/ABCE697D8005/CCAPI/cc_api_call_setup_ind_common:
   Interface Type=0, Protocol=1
*May  1 18:58:26.255: //-1/ABCE697D8005/CCAPI/ccCheckClipClir:
   Calling Party Number Is User Provided
```

The following line shows the attributes of the calling number:

```
*May  1 18:58:26.255: //-1/ABCE697D8005/CCAPI/cc_api_call_setup_ind_common:
   After Number Translation Checking:
   Calling Number=4085550111(TON=National, NPI=ISDN, Screening=User, Passed,
Presentation=Allowed),
   Called Number=3600(TON=Unknown, NPI=Unknown)
*May  1 18:58:26.255: //8/xxxxxxxxxxxx/CCAPI/cc_insert_call_entry:
   Total Call Count=0, Call Entry(Call Count On=FALSE, Incoming Call=TRUE)
*May  1 18:58:26.255: //8/xxxxxxxxxxxx/CCAPI/cc_insert_call_entry:
   Total Call Count=1
*May  1 18:58:26.255: //8/ABCE697D8005/CCAPI/cc_insert_guid_pod_entry:
   Incoming=TRUE, Call Id=8
```

The following line shows the IP address of the originating gateway:

```
*May  1 18:58:26.255: //8/ABCE697D8005/CCAPI/cc_incr_if_call_volume:
   Remote IP Address=172.16.13.175, Hwidb=FastEthernet0/0
*May  1 18:58:26.255: //8/ABCE697D8005/CCAPI/cc_incr_if_call_volume:
   Total Call Count=1, Voip Call Count=1, MMoip Call Count=0
*May  1 18:58:26.255: //8/ABCE697D8005/CCAPI/cc_setupind_registration_lookup:
   Matching Parameters; Called Number=3600, Call Transfer Consult Id=
*May  1 18:58:26.255: //8/ABCE697D8005/CCAPI/cc_setupind_registration_lookup:
   No Matching Node
*May  1 18:58:26.255: //8/ABCE697D8005/CCAPI/ccCheckClipClir:
   Calling Party Number Is User Provided
*May  1 18:58:26.259: //9/xxxxxxxxxxxx/CCAPI/cc_insert_call_entry:
   Total Call Count=1, Call Entry(Call Count On=FALSE, Incoming Call=FALSE)
*May  1 18:58:26.259: //8/ABCE697D8005/CCAPI/cc_peer_bind:
   Bind=TRUE, Binder Call Id=8, Bindee Call Id=9
*May  1 18:58:26.259: //9/ABCE697D8005/CCAPI/cc_insert_guid_pod_entry:
   Incoming=FALSE, Call Id=9
*May  1 18:58:26.259: //9/ABCE697D8005/CCAPI/cc_set_voice_port_value:
   CC_IF_TELEPHONY: Echo=0, Playout=0
*May  1 18:58:26.263: //9/ABCE697D8005/CCAPI/ccCallGetContext:
   Context=0x64B6BB5C, Call Id=9
*May  1 18:59:21.871: //8/ABCE697D8005/CCAPI/ccCallDisconnect:
   Start Calling Accounting;
   Call Entry(Incoming=TRUE)
*May  1 18:59:21.875: //8/ABCE697D8005/CCAPI/ccCallDisconnect:
   Cause Value=16, Call Entry(Disconnect Cause=16)
*May  1 18:59:21.875: //8/ABCE697D8005/CCAPI/ccCallDisconnect:
   Call Entry(Disconnect Cause=16)
*May  1 18:59:21.875: //9/ABCE697D8005/CCAPI/ccCallDisconnect:
   Start Calling Accounting;
   Call Entry(Incoming=FALSE)
*May  1 18:59:21.875: //9/ABCE697D8005/CCAPI/ccCallDisconnect:
   Cause Value=16, Call Entry(Disconnect Cause=0)
```

The next line shows the impairment calculation. This is the only CCAPI debug command that shows impairment.

```
*May  1 18:59:21.875: //-1/ABCE697D8005/CCAPI/g113_calculate_impairment:
   (delay=99(ms), loss=0%), Io=0 Iq=0 Idte=0 Idd=2 Ie=10 Itot=12
*May  1 18:59:21.875: //8/ABCE697D8005/CCAPI/ccCallGetVoipFlag:
   Data Bitmask=0x1, Call Id=8
*May  1 18:59:21.875: //8/ABCE697D8005/CCAPI/ccCallGetVoipFlag:
   Flag=FALSE
*May  1 18:59:21.875: //8/ABCE697D8005/CCAPI/cc_decr_if_call_volume:
   Remote IP Address=172.16.13.175, Hwidb=FastEthernet0/0
*May  1 18:59:21.875: //8/ABCE697D8005/CCAPI/cc_decr_if_call_volume:
   Total Call Count=0, Voip Call Count=0, MMoip Call Count=0
*May  1 18:59:21.875: //8/ABCE697D8005/CCAPI/cc_delete_guid_pod_entry:
   Incoming=TRUE
*May  1 18:59:21.875: //8/ABCE697D8005/CCAPI/cc_delete_call_entry:
   ccFreeRawMsgInfo=0x644EB850
Router#
*May  1 18:59:21.875: //8/ABCE697D8005/CCAPI/cc_delete_call_entry:
   Total Call Count=1, Call Entry(Call Count On=FALSE, Incoming Call=TRUE)
*May  1 18:59:21.875: //8/ABCE697D8005/CCAPI/cc_delete_call_entry:
   Total Call Count=0
*May  1 18:59:21.879: //8/ABCE697D8005/CCAPI/cc_delete_call_entry:
   Deleting profileTable[0x64B78600]
*May  1 18:59:21.879: //8/xxxxxxxxxxxx/CCAPI/cc_get_call_entry:
   Call Entry Is Not Found
*May  1 18:59:21.879: //8/xxxxxxxxxxxx/CCAPI/cc_get_call_entry:
   Call Entry Is Not Found
Router#
*May  1 18:59:24.587: //9/ABCE697D8005/CCAPI/ccCallGetVoipFlag:
   Data Bitmask=0x1, Call Id=9
*May  1 18:59:24.587: //9/ABCE697D8005/CCAPI/ccCallGetVoipFlag:
   Flag=FALSE
*May  1 18:59:24.587: //9/ABCE697D8005/CCAPI/cc_api_call_disconnect_done:
   Prefix Is Not Defined From Peer; Peer=3600, Called Number=3600
*May  1 18:59:24.587: //9/ABCE697D8005/CCAPI/cc_delete_guid_pod_entry:
   Incoming=FALSE
*May  1 18:59:24.587: //9/ABCE697D8005/CCAPI/cc_delete_call_entry:
   Total Call Count=0, Call Entry(Call Count On=FALSE, Incoming Call=FALSE)
*May  1 18:59:24.587: //9/ABCE697D8005/CCAPI/cc_delete_call_entry:
   Deleting profileTable[0x6453F228]
```

Examples

```
Router# debug voip ccapi inout
voip ccapi inout debugging is on
Router#
*Apr 18 20:42:19.347: //-1/9C5A9CA88009/CCAPI/cc_api_display_ie_subfields:
   cc_api_call_setup_ind_common:
   acme-username=
   ----- ccCallInfo IE subfields -----
   acme-ani=4085550111
   acme-anitype=2
   acme-aniplan=1
   acme-anipi=0
   acme-anisi=1
   dest=83103
   acme-desttype=0
   acme-destplan=0
   acme-rdn=
   acme-rdntype=-1
   acme-rdnplan=-1
   acme-rdnpi=-1
   acme-rdnsi=-1
   acme-redirectreason=-1
```

The following lines show information about the calling and called numbers. The network presentation indicator (NPI) shows the type of transmission. The Incoming Dial-Peer field shows that the incoming dial peer has been matched.

```
*Apr 18 20:42:19.347: //-1/9C5A9CA88009/CCAPI/cc_api_call_setup_ind_common:
   Interface=0x64F26F10, Call Info(
   Calling Number=4085550111(TON=National, NPI=ISDN, Screening=User, Passed,
Presentation=Allowed),
   Called Number=83103(TON=Unknown, NPI=Unknown),
   Calling Translated=FALSE, Subsriber Type Str=RegularLine, FinalDestinationFlag=TRUE,
   Incoming Dial-peer=1, Progress Indication=NULL(0), Calling IE Present=TRUE,
   Source Trkgrp Route Label=, Target Trkgrp Route Label=, CLID Transparent=FALSE), Call
Id=-1
*Apr 18 20:42:19.347: //-1/9C5A9CA88009/CCAPI/ccCheckClipClir:
   In: Calling Number=4085550111(TON=National, NPI=ISDN, Screening=User, Passed,
Presentation=Allowed)
*Apr 18 20:42:19.347: //-1/9C5A9CA88009/CCAPI/ccCheckClipClir:
   Out: Calling Number=4085550111(TON=National, NPI=ISDN, Screening=User, Passed,
Presentation=Allowed)
```

In the following event, the call leg is established. The CallEntry ID field changes from-1 to 19.

```
*Apr 18 20:42:19.347: //19/9C5A9CA88009/CCAPI/cc_api_call_setup_ind_common:
   Set Up Event Sent;
   Call Info(Calling Number=4085550111(TON=National, NPI=ISDN, Screening=User, Passed,
Presentation=Allowed),
   Called Number=83103(TON=Unknown, NPI=Unknown))
*Apr 18 20:42:19.351: //19/9C5A9CA88009/CCAPI/cc_process_call_setup_ind:
   Event=0x63FF4730
*Apr 18 20:42:19.351: //19/9C5A9CA88009/CCAPI/ccCallSetContext:
   Context=0x652A9858
*Apr 18 20:42:19.351: //19/9C5A9CA88009/CCAPI/cc_process_call_setup_ind:
   >>>>CCAPI handed cid 19 with tag 1 to app "Default"
*Apr 18 20:42:19.351: //19/9C5A9CA88009/CCAPI/ccCallProceeding:
   Progress Indication=NULL(0)
*Apr 18 20:42:19.351: //19/9C5A9CA88009/CCAPI/ccCallSetupRequest:
   Destination=, Calling IE Present=TRUE, Mode=0,
   Outgoing Dial-peer=3600, Params=0x652AA4A8, Progress Indication=NULL(0)
*Apr 18 20:42:19.351: //19/9C5A9CA88009/CCAPI/ccCheckClipClir:
   In: Calling Number=4085550111(TON=National, NPI=ISDN, Screening=User, Passed,
Presentation=Allowed)
*Apr 18 20:42:19.351: //19/9C5A9CA88009/CCAPI/ccCheckClipClir:
   Out: Calling Number=4085550111(TON=National, NPI=ISDN, Screening=User, Passed,
Presentation=Allowed)
*Apr 18 20:42:19.351: //19/9C5A9CA88009/CCAPI/ccCallSetupRequest:
   Destination Pattern=360., Called Number=3600, Digit Strip=FALSE
*Apr 18 20:42:19.351: //19/9C5A9CA88009/CCAPI/ccCallSetupRequest:
   Calling Number=4085550111(TON=National, NPI=ISDN, Screening=User, Passed,
Presentation=Allowed),
   Called Number=3600(TON=Unknown, NPI=Unknown),
   Redirect Number=, Display Info=
   Account Number=, Final Destination Flag=TRUE,
   Guid=9C5A9CA8-5243-11D6-8009-00059A3A15A0, Outgoing Dial-peer=3600
*Apr 18 20:42:19.351: //19/9C5A9CA88009/CCAPI/cc_api_display_ie_subfields:
   ccCallSetupRequest:
   cisco-username=
   ----- ccCallInfo IE subfields -----
   cisco-ani=4085550111
   cisco-anitype=2
   cisco-aniplan=1
   cisco-anipi=0
   cisco-anisi=1
   dest=3600
   cisco-desttype=0
   cisco-destplan=0
   cisco-rdn=
   cisco-rdntype=-1
   cisco-rdnplan=-1
   cisco-rdnpi=-1
   cisco-rdnsi=-1
   cisco-redirectreason=-1
```

In the following lines, the outgoing dial peer is matched:

```
*Apr 18 20:42:19.351: //19/9C5A9CA88009/CCAPI/ccIFCallSetupRequestPrivate:
   Interface=0x63EAF24C, Interface Type=1, Destination=, Mode=0x0,
   Call Params(Calling Number=4085550111(TON=National, NPI=ISDN, Screening=User, Passed,
Presentation=Allowed),
   Called Number=3600(TON=Unknown, NPI=Unknown), Calling Translated=FALSE,
  Subsriber Type Str=RegularLine, FinalDestinationFlag=TRUE, Outgoing Dial-peer=3600, Call
 Count On=FALSE,
   Source Trkgrp Route Label=, Target Trkgrp Route Label=, tg_label_flag=0, Application
Call Id=)
*Apr 18 20:42:19.351: //20/9C5A9CA88009/CCAPI/ccIFCallSetupRequestPrivate:
   SPI Call Setup Request Is Success; Interface Type=1, FlowMode=1
*Apr 18 20:42:19.351: //20/9C5A9CA88009/CCAPI/ccCallSetContext:
   Context=0x652AA458
*Apr 18 20:42:19.351: //19/9C5A9CA88009/CCAPI/ccSaveDialpeerTag:
   Outgoing Dial-peer=3600
*Apr 18 20:42:19.351: //19/9C5A9CA88009/CCAPI/ccSaveDialpeerTag:
   Outgoing Dial-peer=3600
```

```
*Apr 18 20:42:19.367: //20/9C5A9CA88009/CCAPI/cc_api_call_proceeding:
   Interface=0x652F6388, Progress Indication=NULL(0)
```

The following lines show call progress. The progress and signal indications are shown.

```
*Apr 18 20:42:19.371: //20/9C5A9CA88009/CCAPI/cc_api_call_cut_progress:
   Interface=0x652F6388, Progress Indication=INBAND(8), Signal Indication=SIGNAL RINGBACK(1),

   Cause Value=0
*Apr 18 20:42:19.371: //20/9C5A9CA88009/CCAPI/cc_api_call_cut_progress:
   Call Entry(Responsed=TRUE)
*Apr 18 20:42:19.371: //19/9C5A9CA88009/CCAPI/ccCallCutProgress:
   Progress Indication=INBAND(8), Signal Indication=SIGNAL RINGBACK(1), Cause Value=0
   Voice Call Send Alert=FALSE, Call Entry(AlertSent=FALSE)
*Apr 18 20:42:19.371: //19/9C5A9CA88009/CCAPI/ccCallCutProgress:
   Call Entry(Responsed=TRUE)
```

The following lines show the tone generation information:

```
*Apr 18 20:42:19.371: //19/9C5A9CA88009/CCAPI/ccGenerateToneInfo:
   Stop Tone On Digit=FALSE, Tone=Null,
   Tone Direction=Network, Params=0x0, Call Id=19
*Apr 18 20:42:19.371: //19/9C5A9CA88009/CCAPI/ccConferenceCreate:
   Conference Id=0x652F723C, Call Id1=19, Call Id2=20, Tag=0x0
*Apr 18 20:42:19.371: //20/xxxxxxxxxxxx/CCAPI/cc_api_bridge_done:
   Conference Id=0x6, Source Interface=0x63EAF24C, Source Call Id=20,
   Destination Call Id=19, Disposition=0x0, Tag=0x0
*Apr 18 20:42:19.371: //19/9C5A9CA88009/CCAPI/ccConferenceCreate:
   Call Entry(Conference Id=0x6, Destination Call Id=20)
*Apr 18 20:42:19.371: //20/9C5A9CA88009/CCAPI/ccConferenceCreate:
   Call Entry(Conference Id=0x6, Destination Call Id=19)
*Apr 18 20:42:19.371: //19/xxxxxxxxxxxx/CCAPI/cc_api_bridge_done:
   Conference Id=0x6, Source Interface=0x64F26F10, Source Call Id=19,
   Destination Call Id=20, Disposition=0x0, Tag=0x0
*Apr 18 20:42:19.371: //19/9C5A9CA88009/CCAPI/cc_generic_bridge_done:
   Conference Id=0x6, Source Interface=0x64F26F10, Source Call Id=19,
   Destination Call Id=20, Disposition=0x0, Tag=0x0
*Apr 18 20:42:19.371: //19/9C5A9CA88009/CCAPI/cc_api_caps_ind:
   Destination Interface=0x63EAF24C, Destination Call Id=20, Source Call Id=19,
   Caps(Codec=0x2887F, Fax Rate=0xBF, Vad=0x3,
   Modem=0x2, Codec Bytes=0, Signal Type=3)
*Apr 18 20:42:19.371: //19/9C5A9CA88009/CCAPI/cc_api_caps_ind:
   Caps(Playout Mode=1, Playout Initial=60(ms), Playout Min=40(ms),
   Playout Max=300(ms), Fax Nom=300(ms))
*Apr 18 20:42:19.371: //19/9C5A9CA88009/CCAPI/cc_process_notify_bridge_done:
   Conference Id=0x6, Call Id1=19, Call Id2=20
*Apr 18 20:42:19.375: //20/9C5A9CA88009/CCAPI/cc_api_caps_ind:
   Destination Interface=0x64F26F10, Destination Call Id=19, Source Call Id=20,
   Caps(Codec=0x4, Fax Rate=0x1, Vad=0x2,
   Modem=0x2, Codec Bytes=20, Signal Type=2)
*Apr 18 20:42:19.375: //20/9C5A9CA88009/CCAPI/cc_api_caps_ind:
   Caps(Playout Mode=1, Playout Initial=60(ms), Playout Min=40(ms),
   Playout Max=300(ms), Fax Nom=300(ms))
```

The following lines show codec information:

```
*Apr 18 20:42:19.375: //20/9C5A9CA88009/CCAPI/cc_api_caps_ack:
   Destination Interface=0x64F26F10, Destination Call Id=19, Source Call Id=20,
   Caps(Codec=g729r8(0x4), Fax Rate=FAX_RATE_NONE(0x1), Vad=ON(0x2),
   Modem=ON(0x2), Codec Bytes=20, Signal Type=2, Seq Num Start=6872)
*Apr 18 20:42:19.375: //19/9C5A9CA88009/CCAPI/cc_api_caps_ack:
   Destination Interface=0x63EAF24C, Destination Call Id=20, Source Call Id=19,
   Caps(Codec=g729r8(0x4), Fax Rate=FAX_RATE_NONE(0x1), Vad=ON(0x2),
   Modem=ON(0x2), Codec Bytes=20, Signal Type=2, Seq Num Start=6872)
*Apr 18 20:42:19.375: //19/9C5A9CA88009/CCAPI/cc_api_voice_mode_event:
   Call Id=19
*Apr 18 20:42:19.375: //19/9C5A9CA88009/CCAPI/cc_api_voice_mode_event:
   Call Entry(Context=0x652A9858)
```

The following lines show progress indication information. In this case, the event shows that the destination is not ISDN.

```
*Apr 18 20:42:26.855: //20/9C5A9CA88009/CCAPI/cc_api_call_connected:
   Interface=0x652F6388, Data Bitmask=0x0, Progress Indication=DESTINATION IS NON ISDN(2),
   Connection Handle=0
*Apr 18 20:42:26.855: //20/9C5A9CA88009/CCAPI/cc_api_call_connected:
   Call Entry(Connected=TRUE, Responsed=TRUE, Retry Count=0)
*Apr 18 20:42:26.855: //19/9C5A9CA88009/CCAPI/ccCallConnect:
   Progress Indication=DESTINATION IS NON ISDN(2), Data Bitmask=0x0
*Apr 18 20:42:26.855: //19/9C5A9CA88009/CCAPI/ccCallConnect:
   Call Entry(Connected=TRUE, Responsed=TRUE)
*Apr 18 20:42:26.855: //20/9C5A9CA88009/CCAPI/ccSaveDialpeerTag:
   Incoming Dial-peer=1
*Apr 18 20:42:26.859: //19/9C5A9CA88009/CCAPI/ccSaveDialpeerTag:
   Outgoing Dial-peer=3600
*Apr 18 20:42:26.859: //20/9C5A9CA88009/CCAPI/ccCallFeature:
   Feature Type=24, Call Id=20
```

This event shows that the call is disconnected.

```
*Apr 18 20:43:16.795: //19/9C5A9CA88009/CCAPI/cc_api_call_disconnected:
   Cause Value=16, Interface=0x64F26F10, Call Id=19
*Apr 18 20:43:16.795: //19/9C5A9CA88009/CCAPI/cc_api_call_disconnected:
   Call Entry(Responsed=TRUE, Cause Value=16, Retry Count=0)
*Apr 18 20:43:16.795: //19/9C5A9CA88009/CCAPI/ccConferenceDestroy:
   Conference Id=0x6, Tag=0x0
*Apr 18 20:43:16.795: //20/xxxxxxxxxxxx/CCAPI/cc_api_bridge_drop_done:
   Conference Id=0x6, Source Interface=0x63EAF24C, Source Call Id=20,
   Destination Call Id=19, Disposition=0x0, Tag=0x0
*Apr 18 20:43:16.795: //19/xxxxxxxxxxxx/CCAPI/cc_api_bridge_drop_done:
   Conference Id=0x6, Source Interface=0x64F26F10, Source Call Id=19,
   Destination Call Id=20, Disposition=0x0, Tag=0x0
*Apr 18 20:43:16.795: //19/9C5A9CA88009/CCAPI/cc_generic_bridge_done:
   Conference Id=0x6, Source Interface=0x64F26F10, Source Call Id=19,
   Destination Call Id=20, Disposition=0x0, Tag=0x0
*Apr 18 20:43:16.795: //19/9C5A9CA88009/CCAPI/ccCallDisconnect:
   Cause Value=16, Tag=0x0, Call Entry(Previous Disconnect Cause=0, Disconnect Cause=16)
*Apr 18 20:43:16.795: //19/9C5A9CA88009/CCAPI/ccCallDisconnect:
   Cause Value=16, Call Entry(Responsed=TRUE, Cause Value=16)
*Apr 18 20:43:16.795: //19/9C5A9CA88009/CCAPI/cc_api_get_transfer_info:
   Transfer Number Is Null
*Apr 18 20:43:16.795: //20/9C5A9CA88009/CCAPI/ccCallDisconnect:
   Cause Value=16, Tag=0x0, Call Entry(Previous Disconnect Cause=0, Disconnect Cause=0)
*Apr 18 20:43:16.795: //20/9C5A9CA88009/CCAPI/ccCallDisconnect:
   Cause Value=16, Call Entry(Responsed=TRUE, Cause Value=16)
*Apr 18 20:43:16.795: //20/9C5A9CA88009/CCAPI/cc_api_get_transfer_info:
   Transfer Number Is Null
*Apr 18 20:43:16.803: //20/9C5A9CA88009/CCAPI/cc_api_call_disconnect_done:
   Disposition=0, Interface=0x652F6388, Tag=0x0, Call Id=20,
   Call Entry(Disconnect Cause=16, Voice Class Cause Code=0, Retry Count=0)
*Apr 18 20:43:16.803: //20/9C5A9CA88009/CCAPI/cc_api_call_disconnect_done:
   Call Disconnect Event Sent
*Apr 18 20:43:16.803: //19/9C5A9CA88009/CCAPI/cc_api_call_disconnect_done:
   Disposition=0, Interface=0x64F26F10, Tag=0x0, Call Id=19,
   Call Entry(Disconnect Cause=16, Voice Class Cause Code=0, Retry Count=0)
*Apr 18 20:43:16.803: //19/9C5A9CA88009/CCAPI/cc_api_call_disconnect_done:
   Call Disconnect Event Sent
```

Examples

```
Router# debug voip ccapi service

voip ccapi service debugging is on
*May  1 19:08:41.803: //-1/xxxxxxxxxxxx/CCAPI/cc_setupind_match_search:
   Searching Node;
   Called Number=3600, Call Transfer Consult Id=
```

This debug shows noncall related events. In this case, information about the timer is shown.

```
*May  1 19:08:48.027: //-1/xxxxxxxxxxxx/CCAPI/cc_handle_periodic_timer:
   Calling The Callback, ccTimerctx=0x63B368C0
*May  1 19:08:48.027: //-1/xxxxxxxxxxxx/CCAPI/ccTimerStart:
   ccTimerctx=0x63B368C0
*May  1 19:10:08.615: //-1/xxxxxxxxxxxx/CCAPI/cc_api_icpif:
   ExpectFactor=0xA
```

Examples

This debug shows the preemption tone timer getting initiated.

```
*Aug 24 18:28:16.919: //18958/B37648B6AF48/CCAPI/cc_api_call_preempt_ind:
   PreemptionToneTimer=10(s)
```

debug voip ccapi error

To trace error logs in the call control application programming interface (CCAPI), use the **debug voip ccapi error**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voip ccapi error

no debug voip ccapi error

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(11)T	This command was implemented on the following platforms: Cisco 2600 series, Cisco 3620, Cisco 3660, Cisco AS5350, Cisco AS5400, Cisco AS5850, Cisco AS5300, Cisco AS5800, and Cisco MC3810.

Usage Guidelines

The **debug voip ccapi error** command traces the error logs in the call control API. Error logs are generated during normal call processing, when there are insufficient resources, or when there are problems in the underlying network-specific code, the higher call session application, or the call control API itself.

This **debug** command shows error events or unexpected behavior in system software. In most cases, no events will be generated.

Note We recommend that you log output from the **debug voip ccapi error**command to a buffer rather than sending the output to the console; otherwise, the size of the output could severely impact the performance of the gateway.

Related Commands

Command	Description
debug voip ccapi inout	Traces the execution path through the CCAPI.
voice call debug	Allows configuration of the voice call debug output.

debug voip ccapi inout

To trace the execution path through the call control application programming interface (CCAPI), use the **debug voip ccapi inout command**in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voip ccapi inout

no debug voip ccapi inout

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(11)T	This command was implemented on the following platforms: Cisco 2600 series, Cisco 3620, Cisco 3660, Cisco AS5350, Cisco AS5400, Cisco AS5850, Cisco AS5300, Cisco AS5800, and Cisco MC3810.

Usage Guidelines

The **debug voip ccapi inout** command traces the execution path through the call control API, which serves as the interface between the call session application and the underlying network-specific software. You can use the output from this command to understand how calls are being handled by the voice gateway.

This command shows how a call flows through the system. Using this debug level, you can see the call setup and teardown operations performed on both the telephony and network call legs.

Note We recommend that you log output from the **debug voip ccapi inout**command to a buffer rather than sending the output to the console; otherwise, the size of the output could severely impact the performance of the gateway.

Examples

The following example shows the call setup indicated and accepted by the voice gateway:

```
Router# debug voip ccapi inout
*Mar  1 15:35:53.588: //-1/xxxxxxxxxxxx/CCAPI/ccTDConstructTDUsrContainer:
usrContainer[0x638C1BF0], magic[FACE0FFF]
*Mar  1 15:35:53.592: //-1/xxxxxxxxxxxx/CCAPI/ccTDUtilAddDataToUsrContainer:
container=0x638C1BF0, tagID=6, dataSize=16, instID=-1,modifier=1
*Mar  1 15:35:53.592: //-1/xxxxxxxxxxxx/CCAPI/ccTDConstructInstanceTDObject:
tdObject[0x638BC1AC], nxtElem[0x0], magic[0xFACE0FFF] tagID[6], dataLen[16],
modif[1]
*Mar  1 15:35:53.592: //-1/xxxxxxxxxxxx/CCAPI/ccTDPvtAddObjectToContainer:
Adding tdObject[0x638BC1AC] instID[-1] into container[0x638C1BF0]
*Mar  1 15:35:53.592: //-1/xxxxxxxxxxxx/CCAPI/ccTDUtilAddDataToUsrContainer:
container=0x638C1BF0, tagID=5, dataSize=276, instID=-1,modifier=1
*Mar  1 15:35:53.592: //-1/xxxxxxxxxxxx/CCAPI/ccTDConstructInstanceTDObject:
```

```
tdObject[0x63401148], nxtElem[0x0], magic[0xFACE0FFF] tagID[5], dataLen[276],
modif[1]
*Mar  1 15:35:53.592: //-1/xxxxxxxxxxxx/CCAPI/ccTDPvtAddObjectToContainer:
Adding tdObject[0x63401148] instID[-1] into container[0x638C1BF0]
```

In the following lines, the call control API (CCAPI) receives the call setup. The called number is 34999, and the calling number is 55555. The calling number matches dial peer 10002.

```
*Mar  1 15:35:53.592: //-1/xxxxxxxxxxxx/CCAPI/cc_api_display_ie_subfields:
*Mar  1 15:35:53.592: cc_api_call_setup_ind:
*Mar  1 15:35:53.592:  cisco-username=
*Mar  1 15:35:53.596: ----- ccCallInfo IE subfields -----
*Mar  1 15:35:53.596:  cisco-ani=55555
*Mar  1 15:35:53.596:  cisco-anitype=0
*Mar  1 15:35:53.596:  cisco-aniplan=0
*Mar  1 15:35:53.596:  cisco-anipi=0
*Mar  1 15:35:53.596:  cisco-anisi=0
*Mar  1 15:35:53.596:  dest=34999
*Mar  1 15:35:53.596:  cisco-desttype=0
*Mar  1 15:35:53.596:  cisco-destplan=0
*Mar  1 15:35:53.596:  cisco-rdn=
*Mar  1 15:35:53.596:  cisco-rdntype=-1
*Mar  1 15:35:53.596:  cisco-rdnplan=-1
*Mar  1 15:35:53.596:  cisco-rdnpi=-1
*Mar  1 15:35:53.596:  cisco-rdnsi=-1
*Mar  1 15:35:53.596:  cisco-redirectreason=-1
*Mar  1 15:35:53.596: //-1/xxxxxxxxxxxx/CCAPI/cc_api_call_setup_ind:
(vdbPtr=0x637EC1E0,
callInfo={called=34999,called_oct3=0x80,calling=55555,calling_oct3=0x80,calling_oct3a=0x0,calling_xlated=false,subscriber_type_str=RegularLine,fdest=1,
peer_tag=10002, prog_ind=0,callingIE_present 1, src_route_label=, tgt_route_label=
clid_transparent=0},callID=0x637B4278)
*Mar  1 15:35:53.596: //-1/xxxxxxxxxxxx/CCAPI/cc_api_call_setup_ind:
*Mar  1 15:35:53.596: //-1/xxxxxxxxxxxx/CCAPI/cc_api_call_setup_ind: type 13 , prot 0
*Mar  1 15:35:53.596: //-1/xxxxxxxxxxxx/CCAPI/ccCheckClipClir:
*Mar  1 15:35:53.596: ccCheckClipClir: calling number is: "55555", calling oct3a is: 0x0
*Mar  1 15:35:53.596: //-1/xxxxxxxxxxxx/CCAPI/ccCheckClipClir:
*Mar  1 15:35:53.596: Calling Party number is User Provided
*Mar  1 15:35:53.596: //-1/xxxxxxxxxxxx/CCAPI/ccCheckClipClir:
*Mar  1 15:35:53.596: Leaving ccCheckClipClir
  calling number is: "55555"
  calling oct3 is:  0x80
  calling oct3a is: 0x0
```

In the next line, 44 is the CallEntry ID.

```
*Mar  1 15:35:53.600: //44/xxxxxxxxxxxx/CCAPI/cc_insert_call_entry: Increment call volume:
 0
*Mar  1 15:35:53.600: //44/xxxxxxxxxxxx/CCAPI/cc_insert_call_entry: current call volume: 1
*Mar  1 15:35:53.600: //44/xxxxxxxxxxxx/CCAPI/cc_insert_call_entry: entry's incoming TRUE.
*Mar  1 15:35:53.600: //44/xxxxxxxxxxxx/CCAPI/cc_insert_call_entry: is_incoming is TRUE
*Mar  1 15:35:53.600: //-1/xxxxxxxxxxxx/CCAPI/ccTDConstructHashProfileTab:
profileTable[0x6380E11C], numBuckets[11], numEntries[0]
*Mar  1 15:35:53.600: //-1/xxxxxxxxxxxx/CCAPI/ccTDPvtProfileTableBuildManager: Invoking
necessary profileTable updaters...
*Mar  1 15:35:53.600: //-1/xxxxxxxxxxxx/CCAPI/ccTDPvtUpdateProfileTabFromContainer: Updating
 profileTable[0x6380E11C] with objects in container[0x638C1BF0]
*Mar  1 15:35:53.600: //-1/xxxxxxxxxxxx/CCAPI/ccTDPvtUpdateProfileTabFromContainer: obtained
 key[5] for the tag[6]
*Mar  1 15:35:53.600: //-1/xxxxxxxxxxxx/CCAPI/ccTDPvtAddObjectToProfileBucket:
profileTable[0x6380E11C], tdObject[0x638BC1AC]
*Mar  1 15:35:53.600: //-1/xxxxxxxxxxxx/CCAPI/ccTDPvtUpdateProfileTabFromContainer: obtained
 key[0] for the tag[5]
*Mar  1 15:35:53.600: //-1/xxxxxxxxxxxx/CCAPI/ccTDPvtAddObjectToProfileBucket:
profileTable[0x6380E11C], tdObject[0x63401148]
*Mar  1 15:35:53.600: //-1/xxxxxxxxxxxx/CCAPI/ccTDPvtProfileTableBuildManager:
*Mar  1 15:35:53.600: ccTDUtilDumpAllElemInProfileTab: profileTable[0x6380E11C],
 numBuckets[11], numEntries[2]
*Mar  1 15:35:53.600: Bucket { 0 } ------>0x63401148[0x0,t-5,l-276,d-0x63401168,
m-1,u-56153,g-FACE0FFF]
*Mar  1 15:35:53.604:
*Mar  1 15:35:53.604: Bucket { 5 }
------>0x638BC1AC[0x0,t-6,l-16,d-0x638BC1CC,m-1,u-56153,g-FACE0FFF]
```

```
*Mar  1 15:35:53.604:
*Mar  1 15:35:53.604: //-1/xxxxxxxxxxxx/CCAPI/ccTDDestructTDUsrContainer:
Container[0x638C1BF0]
*Mar  1 15:35:53.604: //-1/xxxxxxxxxxxx/CCAPI/cc_incr_if_call_volume: not the VoIP or MMoIP
*Mar  1 15:35:53.608: //-1/xxxxxxxxxxxx/CCAPI/cc_process_call_setup_ind: (event=
0x63073AA0)
```

In the next line, 45F2AAE28044 is the GUID. The tag 10002 entry shows that the incoming dial peer matched the CallEntry ID.

```
*Mar  1 15:35:53.608: //44/45F2AAE28044/CCAPI/cc_process_call_setup_ind: >>>>CCAPI handed
cid 44 with tag 10002 to app "DEFAULT"
*Mar  1 15:35:53.608: //44/xxxxxxxxxxxx/SSAPP:-1:-1/sess_appl: ev(24=CC_EV_CALL_SETUP_IND),
 cid(44), disp(0)
*Mar  1 15:35:53.608: //44/xxxxxxxxxxxx/SSAPP:-1:-1/sess_appl: ev(SSA_EV_CALL_SETUP_IND),
cid(44), disp(0)
*Mar  1 15:35:53.608: //44/xxxxxxxxxxxx/SSAPP:-1:-1/ssaCallSetupInd:
```

The next line shows CallEntry ID in hexadecimal form, 0x2C (44 in decimal). The CallID and GUID numbers have been identified. The incoming dial-peer is 10002.

```
*Mar  1 15:35:53.608: //44/xxxxxxxxxxxx/CCAPI/ccCallSetContext: (callID=0x2C,
context=0x634A430C)
*Mar  1 15:35:53.608: //44/45F2AAE28044/SSAPP:10002:-1/ssaCallSetupInd: cid(44),
st(SSA_CS_MAPPING),oldst(0), ev(24)ev->e.evCallSetupInd.nCallInfo.finalDestFlag
 = 1
*Mar  1 15:35:53.608: //44/45F2AAE28044/SSAPP:10002:-1/ssaCallSetupInd: src route label=,
tgt route label= tg_label_flag 0x0
*Mar  1 15:35:53.608: //44/45F2AAE28044/SSAPP:10002:-1/ssaCallSetupInd: finalDest
cllng(55555), clled(34999) tgt_route_label()tg_label_flag 0x0
*Mar  1 15:35:53.612: //44/45F2AAE28044/SSAPP:10002:-1/ssaCallSetupInd: cid(44),
 st(SSA_CS_CALL_SETTING),oldst(0), ev(24)dpMatchPeersMoreArg result= 0
```

For CallEntry ID 44, two dial-peer tags (10001 and 20002) were matched with called number 34999.

```
*Mar  1 15:35:53.612: //44/45F2AAE28044/SSAPP:10002:-1/ssaDebugPeers: ssaSetupPe
er cid(44) peer list: tag(10001) called number (34999) tag(20002) called number
(34999)
*Mar  1 15:35:53.612: //44/45F2AAE28044/SSAPP:10002:-1/ssaSetupPeer: dialpeer tags in rotary=
 10001  20002
```

The next line shows that 5 digits were matched for this dial peer and no prefix was added. The encapType (2) entry indicates a VoIP call.

```
*Mar  1 15:35:53.612: //44/45F2AAE28044/SSAPP:10002:-1/ssaSetupPeer: cid(44), de
stPat(34999), matched(5), prefix(), peer(637B0984), peer->encapType (2)
*Mar  1 15:35:53.612: //-1/xxxxxxxxxxxx/CCAPI/cc_can_gateway: Call legs: In=6, O
ut=1
```

The next line shows the voice gateway sending out a call-proceeding message to the incoming call leg with progress indicator of 0x0.

```
*Mar  1 15:35:53.612: //44/xxxxxxxxxxxx/CCAPI/ccCallProceeding: (callID=0x2C, pr
og_ind=0x0)
```

The next line shows the voice gateway sending out the call-setup request to the outgoing call leg. The dial-peer is 10001 with the incoming CallEntry ID being 0x2C.

```
*Mar  1 15:35:53.612: //44/xxxxxxxxxxxx/CCAPI/ccCallSetupRequest: (Inbound call
= 0x2C, outbound peer =10001, dest=,
        params=0x63085D80 mode=0, *callID=0x63086314, prog_ind = 0callingIE_pres
ent 1)
*Mar  1 15:35:53.612: //44/45F2AAE28044/CCAPI/ccCallSetupRequest:
*Mar  1 15:35:53.612: ccCallSetupRequest numbering_type 0x80
*Mar  1 15:35:53.612: //44/45F2AAE28044/CCAPI/ccCallSetupRequest:
*Mar  1 15:35:53.616: ccCallSetupRequest: calling number is:55555
*Mar  1 15:35:53.616: //44/45F2AAE28044/CCAPI/ccCallSetupRequest: calling oct3a
is:0x0
*Mar  1 15:35:53.616: //-1/xxxxxxxxxxxx/CCAPI/ccCheckClipClir:
*Mar  1 15:35:53.616: ccCheckClipClir: calling number is: "55555", calling oct3a
 is: 0x0
```

```
*Mar  1 15:35:53.616: //-1/xxxxxxxxxxxx/CCAPI/ccCheckClipClir:
*Mar  1 15:35:53.616: Calling Party number is User Provided
*Mar  1 15:35:53.616: //-1/xxxxxxxxxxxx/CCAPI/ccCheckClipClir:
*Mar  1 15:35:53.616: Leaving ccCheckClipClir
  calling number is: "55555"
  calling oct3 is:  0x80
  calling oct3a is: 0x0
*Mar  1 15:35:53.616: //44/45F2AAE28044/CCAPI/ccCallSetupRequest: after ccCheckC
lipClir - calling oct3a is:0x0
```

The next line shows that all digits are passed.

```
*Mar  1 15:35:53.616: //44/45F2AAE28044/CCAPI/ccCallSetupRequest: dest pattern 3
4999, called 34999, digit_strip 0
*Mar  1 15:35:53.616: //44/45F2AAE28044/CCAPI/ccCallSetupRequest:
*Mar  1 15:35:53.616: callingNumber=55555, calledNumber=34999, redirectNumber= d
isplay_info= calling_oct3a=0
*Mar  1 15:35:53.616: accountNumber=, finalDestFlag=1,
guid=45f2.aae2.1571.11cc.8044.95f5.fabb.6b0f
*Mar  1 15:35:53.616: peer_tag=10001
*Mar  1 15:35:53.616: //-1/xxxxxxxxxxxx/CCAPI/cc_api_display_ie_subfields:
*Mar  1 15:35:53.616: ccCallSetupRequest:
*Mar  1 15:35:53.616:  cisco-username=
*Mar  1 15:35:53.616: ----- ccCallInfo IE subfields -----
*Mar  1 15:35:53.616:  cisco-ani=55555
*Mar  1 15:35:53.616:  cisco-anitype=0
*Mar  1 15:35:53.616:  cisco-aniplan=0
*Mar  1 15:35:53.616:  cisco-anipi=0
*Mar  1 15:35:53.616:  cisco-anisi=0
*Mar  1 15:35:53.620:  dest=34999
*Mar  1 15:35:53.620:  cisco-desttype=0
*Mar  1 15:35:53.620:  cisco-destplan=0
*Mar  1 15:35:53.620:  cisco-rdn=
*Mar  1 15:35:53.620:  cisco-rdntype=-1
*Mar  1 15:35:53.620:  cisco-rdnplan=-1
*Mar  1 15:35:53.620:  cisco-rdnpi=-1
*Mar  1 15:35:53.620:  cisco-rdnsi=-1
*Mar  1 15:35:53.620:  cisco-redirectreason=-1
*Mar  1 15:35:53.620: //-1/xxxxxxxxxxxx/CCAPI/ccIFCallSetupRequestPrivate: (vdbP
tr=0x62EC61A4, dest=, callParams={called=34999,called_oct3=0x80,
calling=55555,calling_oct3=0x80, calling_oct3a= 0x0, calling_xlated=false,
subscriber_type_str
=RegularLine, fdest=1, voice_peer_tag=10001},mode=0x0)
*Mar  1 15:35:53.620: //-1/xxxxxxxxxxxx/CCAPI/ccIFCallSetupRequestPrivate:
*Mar  1 15:35:53.620: ccIFCallSetupRequestPrivate: src route label  tgt route label
tg_label_flag 0x0
*Mar  1 15:35:53.620: //-1/xxxxxxxxxxxx/CCAPI/ccIFCallSetupRequestPrivate:  vdbPtr type =
1
*Mar  1 15:35:53.620: //-1/xxxxxxxxxxxx/CCAPI/ccIFCallSetupRequestPrivate:
*Mar  1 15:35:53.620: //-1/xxxxxxxxxxxx/CCAPI/ccIFCallSetupRequestPrivate: (vdbPtr=0x62EC61A4,
 dest=, callParams={called=34999, called_oct3 0x80,  calling=55555,calling_oct3 0x80,
calling_oct3a 0x0, calling_xlated=false,  fdest=1, voice_pee
r_tag=10001}, mode=0x0, xltrc=-5)
*Mar  1 15:35:53.620: //-1/xxxxxxxxxxxx/CCAPI/ccIFCallSetupRequestPrivate:
In the next line, outgoing CallEntry ID 45 is bound to the same GUID 45F2AAE28044.
*Mar  1 15:35:53.620: //45/45F2AAE28044/CCAPI/cc_insert_call_entry: not incoming
 entry
*Mar  1 15:35:53.620: //45/45F2AAE28044/CCAPI/cc_insert_call_entry: entry's incoming FALSE.
*Mar  1 15:35:53.620: //45/45F2AAE28044/CCAPI/cc_insert_call_entry: is_incoming
is FALSE
*Mar  1 15:35:53.624: //44/xxxxxxxxxxxx/CCAPI/ccSaveDialpeerTag: (callID=0x2C,
dialpeer_tag=10001)
*Mar  1 15:35:53.624: //45/xxxxxxxxxxxx/CCAPI/ccCallSetContext: (callID=0x2D,
context=0x634A537C) 0x2D (decimal 45 is the second call leg ID).
*Mar  1 15:35:53.624: //44/xxxxxxxxxxxx/CCAPI/ccCallReportDigits: (callID=0x2C,
enable=0x0)
```

The voice gateway informs the incoming call leg that digits were forwarded.

```
*Mar  1 15:35:53.624: //44/xxxxxxxxxxxx/CCAPI/cc_api_call_report_digits_done:
(vdbPtr=0x637EC1E0, callID=0x2C, disp=0)
*Mar  1 15:35:53.624: //44/xxxxxxxxxxxx/SSAPP:-1:-1/sess_appl: ev(54=CC_EV_CALL_
REPORT_DIGITS_DONE), cid(44), disp(0)
```

```
*Mar  1 15:35:53.624: //44/45F2AAE28044/SS
Router#APP:10002:-1/ssaTraceSct: cid(44)st(SSA_CS_CALL_SETTING)ev(SSA_EV_CALL_RE
PORT_DIGITS_DONE)
oldst(SSA_CS_MAPPING)cfid(-1)csize(0)in(1)fDest(1)
*Mar  1 15:35:53.624: //44/45F2AAE28044/SSAPP:10002:-1/ssaTraceSct: -cid2(45)st2
(SSA_CS_CALL_SETTING)oldst2(SSA_CS_MAPPING)
*Mar  1 15:35:53.624: //44/45F2AAE28044/SSAPP:10002:-1/ssaDebugPeers: ssaReportDigitsDone
cid(44) peer list: tag(20002) called number (34999)
*Mar  1 15:35:53.624: //44/45F2AAE28044/SSAPP:10002:-1/ssaReportDigitsDone: call
id=44 Reporting disabled.
*Mar  1 15:35:53.628: //-1/xxxxxxxxxxxx/CCAPI/cc_api_supported_data: data_mode=0x10082
*Mar  1 15:35:53.628: //45/xxxxxxxxxxxx/CCAPI/cc_api_get_ic_leg_obtained_numbers: callID=0x2D
```

The next two lines shows the IP address of the terminating gateway and that the terminating gateway is reached through Ethernet port 0/0.

```
*Mar  1 15:35:53.628: //-1/xxxxxxxxxxxx/CCAPI/cc_incr_if_call_volume: remote IP
is 171.69.85.111
*Mar  1 15:35:53.632: //-1/xxxxxxxxxxxx/CCAPI/cc_incr_if_call_volume: hwidb is Ethernet0/0

*Mar  1 15:35:53.632: //-1/xxxxxxxxxxxx/CCAPI/cc_incr_if_call_volume: create entry in list:
 1
*Mar  1 15:35:53.636: //45/xxxxxxxxxxxx/CCAPI/ccTDUtilGetInstanceCount: For tagI
D[1] of callID[45]
*Mar  1 15:35:53.636: //45/45F2AAE28044/CCAPI/ccTDPvtProfileTableObjectAccessManager: No
profileTable set for callID[45]
*Mar  1 15:35:53.636: //45/xxxxxxxxxxxx/CCAPI/ccTDUtilGetInstanceCount: For tagID[2] of
callID[45]
*Mar  1 15:35:53.636: //45/45F2AAE28044/CCAPI/ccTDPvtProfileTableObjectAccessManager: No
profileTable set for callID[45]
```

The next line shows that the voice gateway received a call proceeding message from the terminating gateway, and then the following line shows that the voice gateway received a call alert from the terminating gateway.

```
*Mar  1 15:35:53.740: //45/xxxxxxxxxxxx/CCAPI/cc_api_call_proceeding: (vdbPtr=0x62EC61A4,
callID=0x2D,
      prog_ind=0x0)
*Mar  1 15:35:53.740: //45/xxxxxxxxxxxx/CCAPI/cc_api_call_alert: (vdbPtr=0x62EC61A4,
callID=0x2D, prog_ind=0x0, sig_ind=0x1)
*Mar  1 15:35:53.744: //45/xxxxxxxxxxxx/SSAPP:-1:-1/sess_appl: ev(21=CC_EV_CALL_PROCEEDING),
 cid(45), disp(0)
*Mar  1 15:35:53.744: //45/45F2AAE28044/SSAPP:0:-1/ssaTraceSct: cid(45)st(SSA_CS
_CALL_SETTING)ev(SSA_EV_CALL_PROCEEDING)
oldst(SSA_CS_MAPPING)cfid(-1)csize(0)in(0)fDest(0)
*Mar  1 15:35:53.744: //45/45F2AAE28044/SSAPP:0:-1/ssaTraceSct:
-cid2(44)st2(SSA_CS_CALL_SETTING)oldst2(SSA_CS_CALL_SETTING)
*Mar  1 15:35:53.744: //45/45F2AAE28044/SSAPP:0:-1/ssaCallProc:
*Mar  1 15:35:53.744: //44/xxxxxxxxxxxx/CCAPI/ccGetDialpeerTag: (callID=0x2C)
*Mar  1 15:35:53.744: //45/45F2AAE28044/SSAPP:0:-1/ssaIgnore: cid(45), st(SSA_CS
_CALL_SETTING),oldst(1), ev(21)
*Mar  1 15:35:53.744: //45/xxxxxxxxxxxx/SSAPP:-1:-1/sess_appl: ev(7=CC_EV_CALL_ALERT),
cid(45), disp(0)
*Mar  1 15:35:53.744: //45/45F2AAE28044/SSAPP:0:-1/ssaTraceSct: cid(45)st(SSA_CS
_CALL_SETTING)ev(SSA_EV_CALL_ALERT)
oldst(SSA_CS_CALL_SETTING)cfid(-1)csize(0)in(0)fDest(0)
*Mar  1 15:35:53.744: //45/45F2AAE28044/SSAPP:0:-1/ssaTraceSct: -cid2(44)st2(SSA
_CS_CALL_SETTING)oldst2(SSA_CS_CALL_SETTING)
*Mar  1 15:35:53.744: //44/45F2AAE28044/SSAPP:10002:-1/ssaAlert:
*Mar  1 15:35:53.744: //44/xxxxxxxxxxxx/CCAPI/ccGetDialpeerTag: (callID=0x2C)
Router#
```

The voice gateway forwarded a call alert to the originating gateway.

```
*Mar  1 15:35:53.744: //44/xxxxxxxxxxxx/CCAPI/ccCallAlert: (callID=0x2C, prog_ind=0x0,
sig_ind=0x1)
Router#
```

The phone is answered at the called number.

```
Router#!call answered
Router#
```

The voice gateway receives a connect message from the terminating gateway.

```
*Mar  1 15:36:05.016: //45/xxxxxxxxxxxx/CCAPI/cc_api_call_connected: (vdbPtr=0x6
2EC61A4, callID=0x2D), prog_ind = 0
*Mar  1 15:36:05.016: //45/45F2AAE28044/CCAPI/cc_api_call_connected: setting
callEntry->connected to TRUE
```

The next line shows that the call accounting starts. The leg_type=False message means this is for an outgoing call. The line that follows shows that AAA accounting is not configured.

```
*Mar  1 15:36:05.016: //45/45F2AAE28044/CCAPI/cc_api_call_connected: calling accounting
start for callID=45 leg_type=0
*Mar  1 15:36:05.020: //45/xxxxxxxxxxxx/CCAPI/ccCallSetAAA_Accounting: callID=0x
2D, accounting=0
*Mar  1 15:36:05.020: //45/xxxxxxxxxxxx/SSAPP:-1:-1/sess_appl: ev(8=CC_EV_CALL_CONNECTED),
 cid(45), disp(0)
*Mar  1 15:36:05.020: //45/45F2AAE28044/SSAPP:0:-1/ssaTraceSct: cid(45)st(SSA_CS
_ALERT_RCVD)ev(SSA_EV_CALL_CONNECTED)
oldst(SSA_CS_CALL_SETTING)cfid(-1)csize(0)in(0)fDest(0)
*Mar  1 15:36:05.020: //45/45F2AAE28044/SSAPP:0:-1/ssaTraceSct: -cid2(44)st2(SSA
_CS_ALERT_RCVD)oldst2(SSA_CS_CALL_SETTING)
*Mar  1 15:36:05.020: //45/45F2AAE28044/SSAPP:0:-1/ssaConnect:
*Mar  1 15:36:05.020: //44/xxxxxxxxxxxx/CCAPI/ccGetDialpeerTag: (callID=0x2C)
```

The next lines show a conference being set up between the two call legs 0x2C and 0x2D. Bridge complete messages are sent to both the terminating and originating gateways.

```
*Mar  1 15:36:05.020: //44/xxxxxxxxxxxx/CCAPI/ccConferenceCreate: (confID=0x6308
6424, callID1=0x2C, callID2=0x2D, tag=0x0)
*Mar  1 15:36:05.020: //45/xxxxxxxxxxxx/CCAPI/cc_api_bridge_done: (confID=0x15,
srcIF=0x62EC61A4, srcCallID=0x2D, dstCallID=0x2C, disposition=0, tag=0x0)
*Mar  1 15:36:05.024: //44/xxxxxxxxxxxx/CCAPI/cc_api_bridge_done: (confID=0x15,
srcIF=0x637EC1E0, srcCallID=0x2C, dstCallID=0x2D, disposition=0, tag=0x0)
```

Here, the voice gateway sets up negotiating capability with the originating telephony leg.

```
*Mar  1 15:36:05.024: //44/xxxxxxxxxxxx/CCAPI/cc_api_caps_ind: (dstVdbPtr=0x62EC
61A4, dstCallId=0x2D, srcCallId=0x2C,
     caps={codec=0x2887F, fax_rate=0xBF, vad=0x3, modem=0x2
           codec_bytes=0, signal_type=3})
*Mar  1 15:36:05.024: //44/xxxxxxxxxxxx/CCAPI/cc_api_caps_ind: (Playout: mode 0,
 initial 60,min 40, max 300)
*Mar  1 15:36:05.024: //44/xxxxxxxxxxxx/SSAPP:-1:-1/sess_appl: ev(29=CC_EV_CONF_
CREATE_DONE), cid(44), disp(0)
*Mar  1 15:36:05.024: //44/45F2AAE28044/SSAPP:10002:21/ssaTraceSct:
cid(44)st(SSA_CS_CONFERENCING)ev(SSA_EV_CONF_CREATE_DONE)
oldst(SSA_CS_CALL_SETTING)cfid(21)csize(2)in(1)fDest(1)
*Mar  1 15:36:05.024: //44/45F2AAE28044/SSAPP:10002:21/ssaTraceSct: -cid2(45)st2
(SSA_CS_CONFERENCING)oldst2(SSA_CS_ALERT_RCVD)
*Mar  1 15:36:05.024: //44/45F2AAE28044/SSAPP:10002:21/ssaConfCreateDone:
*Mar  1 15:36:05.024: //44/xxxxxxxxxxxx/CCAPI/ccCallConnect: (callID=0x2C), prog
_ind = 0
*Mar  1 15:36:05.024: //44/45F2AAE28044/CCAPI/ccCallConnect: setting callEntry->
connected to TRUE
*Mar  1 15:36:05.024: //44/45F2AAE28044/SSAPP:10002:21/ssaDebugPeers: ssaFlushPeerTagQueue
 cid(44) peer list: tag(20002) called number (34999)
*Mar  1 15:36:05.028: //-1/xxxxxxxxxxxx/CCAPI/cc_process_notify_bridge_done:
(event=0x63067FC0)
```

The voice gateway sets up negotiating capability with the terminating VoIP leg.

```
*Mar  1 15:36:05.028: //45/xxxxxxxxxxxx/CCAPI/cc_api_caps_ind: (dstVdbPtr=0x637E
C1E0, dstCallId=0x2C, srcCallId=0x2D,
     caps={codec=0x4, fax_rate=0x2, vad=0x2, modem=0x0
           codec_bytes=20, signal_type=2})
*Mar  1 15:36:05.028: //45/xxxxxxxxxxxx/CCAPI/
cc_api_caps_ind:
 (Playout: mode 0,
 initial 60,min 40, max 300)
```

The capabilities are acknowledged for both call legs.

```
*Mar  1 15:36:05.028: //45/xxxxxxxxxxxx/CCAPI/cc_api_caps_ack: (dstVdbPtr=0x637E
C1E0, dstCallId=0x2C, srcCallId=0x2D,
     caps={codec=0x4, fax_rate=0x2, vad=0x2, modem=0x0
           codec_bytes=20, signal_type=2, seq_num_start=2944})
*Mar  1 15:36:05.028: //44/xxxxxxxxxxxx/CCAPI/cc_api_caps_ack: (dstVdbPtr=0x62EC
61A4, dstCallId=0x2D, srcCallId=0x2C,
     caps={codec=0x4, fax_rate=0x2, vad=0x2, modem=0x0
           codec_bytes=20, signal_type=2, seq_num_start=2944})
*Mar  1 15:36:05.032: //44/xxxxxxxxxxxx/CCAPI/cc_api_voice_mode_event: callID=0x2C
*Mar  1 15:36:05.032: //44/45F2AAE28044/CCAPI/cc_api_voice_mode_event: Call Pointer =634A430C
*Mar  1 15:36:05.032: //44/xxxxxxxxxxxx/SSAPP:-1:-1/sess_appl: ev(52=CC_EV_VOICE
_MODE_DONE), cid(44), disp(0)
*Mar  1 15:36:05.032: //44/45F2AAE28044/SSAPP:10002:21/ssaTraceSct:
Router#
Router# cid(44)st(SSA_CS_ACTIVE)ev(SSA_EV_VOICE_MODE_DONE)
oldst(SSA_CS_CONFERENCING)cfid(21)csize(2)in(1)fDest(1)
*Mar  1 15:36:05.032: //44/45F2AAE28044/SSAPP:10002:21/ssaTraceSct: -cid2(45)st2
(SSA_CS_ACTIVE)oldst2(SSA_CS_ALERT_RCVD)
*Mar  1 15:36:05.032: //44/45F2AAE28044/SSAPP:10002:21/ssaIgnore: cid(44), st(SS
A_CS_ACTIVE),oldst(5), ev(52)
Router#
Router#! digit punched
Router#
```

The phone at the terminating gateway enters digit1.

```
*Mar  1 15:36:11.204: //45/xxxxxxxxxxxx/CCAPI/cc_api_call_digit_begin: (dstVdbPtr=0x637EC1E0,
 dstCallId=0x2C, srcCallId=0x2D,
    digit=1, digit_begin_flags=0x0, rtp_timestamp=0x0
    rtp_expiration=0x0, dest_mask=0x2)
*Mar  1 15:36:11.504: //45/xxxxxxxxxxxx/CCAPI/cc_api_call_digit_end: (dstVdbPtr=
0x637EC1E0, dstCallId=0x2C, srcCallId=0x2D,
   digit=1,duration=300,xruleCallingTag=0,xruleCalledTag=0, dest_mask=0x2), digit_tone_mode=0
```

The phone at the terminating gateway enters digit **2.**

```
*Mar  1 15:36:11.604: //45/xxxxxxxxxxxx/CCAPI/cc_api_call_digit_begin: (dstVdbPt
r=0x637EC1E0, dstCallId=0x2C, srcCallId=0x2D,
    digit=2, digit_begin_flags=0x0, rtp_timestamp=0x0
    rtp_expiration=0x0, dest_mask=0x2)
*Mar  1 15:36:11.904: //45/xxxxxxxxxxxx/CCAPI/cc_api_call_digit_end: (dstVdbPtr=
0x637EC1E0, dstCallId=0x2C, srcCallId=0x2D,
   digit=2,duration=300,xruleCallingTag=0,xruleCalledTag=0, dest_mask=0x2), digit_tone_mode=0
Router#
Router#
*Mar  1 15:36:14.476: //-1/xxxxxxxxxxxx/CCAPI/cc_handle_periodic_timer: Calling
the callback, ccTimerctx - 0x628B6330
*Mar  1 15:36:14.476: //-1/xxxxxxxxxxxx/CCAPI/ccTimerStart: ccTimerctx - 0x628B6330
Router#
Router# !call hung up  The user at the terminating gateway hangs up the call.
Router#
```

The voice gateway receives a disconnect message from the terminating gateway. The cause code is 0x10 which is normal call clearing.

```
*Mar  1 15:36:22.916: //45/xxxxxxxxxxxx/CCAPI/cc_api_call_disconnected: (vdbPtr=
0x62EC61A4, callID=0x2D, cause=0x10)
*Mar  1 15:36:22.920: //45/xxxxxxxxxxxx/SSAPP:-1:-1/sess_appl: ev(11=CC_EV_CALL_
DISCONNECTED), cid(45), disp(0)
*Mar  1 15:36:22.920: //45/45F2AAE28044/SSAPP:0:21/ssaTraceSct: cid(45)st(SSA_CS
_ACTIVE)ev(SSA_EV_CALL_DISCONNECTED)
oldst(SSA_CS_ALERT_RCVD)cfid(21)csize(2)in(0)fDest(0)
*Mar  1 15:36:22.920: //45/45F2AAE28044/SSAPP:0:21/ssaTraceSct: -cid2(44)st2(SSA
_CS_ACTIVE)oldst2(SSA_CS_ACTIVE)
*Mar  1 15:36:22.920: ssa: Disconnected cid(45) state(5) cause(0x10)
```

The voice gateway begins tearing down the conference and dropping the bridge.

```
*Mar  1 15:36:22.920: //-1/xxxxxxxxxxxx/CCAPI/ccConferenceDestroy: (confID=0x15,
 tag=0x0)
*Mar  1 15:36:22.920: //45/xxxxxxxxxxxx/CCAPI/cc_api_bridge_drop_done: (confID=0
x15, srcIF=0x62EC61A4, srcCallID=0x2D, dstCallID=0x2C, disposition=0 tag=0x0)
*Mar  1 15:36:22.920: //44/xxxxxxxxxxxx/CCAPI/cc_api_bridge_drop_done: (confID=0
x15, srcIF=0x637EC1E0, srcCallID=0x2C, dstCallID=0x2D, disposition=0 tag=0x0)
*Mar  1 15:36:22.924: //44/xxxxxxxxxxxx/SSAPP:-1:-1/sess_appl: ev(30=CC_EV_CONF_
DESTROY_DONE), cid(44), disp(0)
*Mar  1 15:36:22.924: //44/45F2AAE28044/SSAPP:10002:21/ssaTraceSct:
cid(44)st(SSA_CS_CONF_DESTROYING)ev(SSA_EV_CONF_DESTROY_DONE)
oldst(SSA_CS_ACTIVE)cfid(21)csize(2)in(1)fDest(1)
*Mar  1 15:36:22.924: //44/45F2AAE28044/SSAPP:10002:21/ssaTraceSct: -cid2(45)st2
(SSA_CS_CONF_DESTROYING)oldst2(SSA_CS_ACTIVE)
*Mar  1 15:36:22.924: //45/45F2AAE28044/SSAPP:0:-1/ssaConfDestroyDone:
*Mar  1 15:36:22.924: //44/xxxxxxxxxxxx/CCAPI/ccCallDisconnect: (callID=0x2C, cause=0x10
tag=0x0)
```

The voice gateway stops call accounting on the incoming call, indicated by the leg_type=True message. The cause code is then set for the originating leg.

```
*Mar  1 15:36:22.924: //44/45F2AAE28044/CCAPI/ccCallDisconnect: calling accounting start
for callID=44 leg_type=1
*Mar  1 15:36:22.924: //44/45F2AAE28044/CCAPI/ccCallDisconnect: existing_cause =
 0x0, new_cause = 0x10
*Mar  1 15:36:22.924: //44/xxxxxxxxxxxx/CCAPI/cc_api_get_transfer_info: (callID=0x2C)
*Mar  1 15:36:22.924: //45/xxxxxxxxxxxx/CCAPI/ccCallDisconnect: (callID=0x2D, cause=0x10
tag=0x0)
```

The voice gateway stops call accounting for the outgoing call, indicated by the leg_type=False message. The cause code is verified for the terminating leg.

```
*Mar  1 15:36:22.924: //45/45F2AAE28044/CCAPI/ccCallDisconnect: calling accounting start
for callID=45 leg_type=0
*Mar  1 15:36:22.924: //45/45F2AAE28044/CCAPI/ccCallDisconnect: existing_cause =
 0x10, new_cause = 0x10
*Mar  1 15:36:22.924: //45/45F2AAE28044/CCAPI/ccCallDisconnect: using the existing_cause
0x10
*Mar  1 15:36:22.928: //45/xxxxxxxxxxxx/CCAPI/cc_api_get_transfer_info: (callID=0x2D)
*Mar  1 15:36:22.932: //-1/xxxxxxxxxxxx/CCAPI/cc_api_icpif: expect factor = 0
*Mar  1 15:36:22.932: //-1/xxxxxxxxxxxx/CCAPI/g113_calculate_impairment: (delay=79,
      loss=0), Io=0 Iq=0 Idte=0 Idd=0 Ie=10 Itot=10
*Mar  1 15:36:22.932: //-1/xxxxxxxxxxxx/CCAPI/cc_decr_if_call_volume: the remote
 IP is 171.69.85.111
*Mar  1 15:36:22.932: //-1/xxxxxxxxxxxx/CCAPI/cc_decr_if_call_volume: hwidb is Ethernet0/0
*Mar  1 15:36:22.932: //-1/xxxxxxxxxxxx/CCAPI/cc_decr_if_call_volume: reduce callnum of
entry: 0, voip: 0, mmoip: 0
*Mar  1 15:36:22.932: //-1/xxxxxxxxxxxx/CCAPI/cc_decr_if_call_volume: remove an entry
*Mar  1 15:36:22.932: //45/xxxxxxxxxxxx/CCAPI/cc_api_call_disconnect_done: (vdbPtr=0x62EC61A4,
 callID=0x2D, disp=0, tag=0x0)
*Mar  1 15:36:22.932: //45/45F2AAE28044/CCAPI/ccTDPvtProfileTableObjectAccessManager: No
profileTable set for callID[45]
*Mar  1 15:36:22.936: //45/xxxxxxxxxxxx/CCAPI/ccTDUtilGetDataByRef: No tdObject
found in profileTable for tagID[6] of callID[45]
*Mar  1 15:36:22.936: //45/45F2AAE28044/CCAPI/cc_delete_call_entry: not incoming
 entry
*Mar  1 15:36:22.936: //45/45F2AAE28044/CCAPI/cc_delete_call_entry: entry's incoming FALSE.
*Mar  1 15:36:22.936: //45/45F2AAE28044/CCAPI/cc_delete_call_entry: is_incoming
is FALSE
*Mar  1 15:36:22.940: //45/xxxxxxxxxxxx/SSAPP:-1:-1/sess_appl: ev(12=CC_EV_CALL_
DISCONNECT_DONE), cid(45), disp(0)
*Mar  1 15:36:22.940: //45/45F2AAE28044/SSAPP:0:-1/ssaTraceSct: cid(45)st(SSA_CS
_DISCONNECTING)ev(SSA_EV_CALL_DISCONNECT_DONE)
oldst(SSA_CS_ACTIVE)cfid(-1)csize(2)in(0)fDest(0)
*Mar  1 15:36:22.940: //45/45F2AAE28044/SSAPP:0:-1/ssaTraceSct: -cid2(44)st2(SSA
_CS_DISCONNECTING)oldst2(SSA_CS_CONF_DESTROYING)
*Mar  1 15:36:22.940: //45/45F2AAE28044/SSAPP:0:-1/ssaDisconnectDone:
*Mar  1 15:36:22.940: //45/45F2AAE28044/SSAPP:0:-1/ssaAAA_CheckAccounting: accounting
generation enabled
*Mar  1 15:36:22.940: //45/xxxxxxxxxxxx/CCAPI/ccCallSetAAA_Accounting: callID=0x2D,
accounting=0
*Mar  1 15:36:22.944: //-1/xxxxxxxxxxxx/CCAPI/cc_decr_if_call_volume: not the VoIP or MMoIP
*Mar  1 15:36:22.948: //44/xxxxxxxxxxxx/CCAPI/cc_api_call_disconnect_done: (vdbPtr=0x637EC1E0,
 callID=0x2C, disp=0, tag=0x0)
*Mar  1 15:36:22.948: //44/45F2AAE28044/CCAPI/cc_delete_call_entry: ccFreeRawMsg
Info(0x6307595C)
*Mar  1 15:36:22.948: //44/45F2AAE28044/CCAPI/cc_delete_call_entry: Decrement call volume
counter 1
*Mar  1 15:36:22.948: //44/45F2AAE28044/CCAPI/cc_delete_call_entry: current call volume: 0
*Mar  1 15:36:22.948: //44/45F2AAE28044/CCAPI/cc_delete_call_entry: entry's incoming TRUE.
*Mar  1 15:36:22.948: //44/45F2AAE28044/CCAPI/cc_delete_call_entry: is_incoming
is TRUE
*Mar  1 15:36:22.948: //44/45F2AAE28044/CCAPI/cc_delete_call_entry: Deleting
profileTable[0x6380E11C]
*Mar  1 15:36:22.948: //-1/xxxxxxxxxxxx/CCAPI/ccTDDestructTDHashProfileTab: Destructor
Profile Table (0x6380E11C)
*Mar  1 15:36:22.948: //-1/xxxxxxxxxxxx/CCAPI/ccTDDestructInstanceTDObject:
tdObject[0x63401148] tagID[5]
*Mar  1 15:36:22.948: //-1/xxxxxxxxxxxx/CCAPI/ccTDDestructInstanceTDObject:
tdObject[0x638BC1AC] tagID[6]
*Mar  1 15:36:22.956: //44/xxxxxxxxxxxx/SSAPP:-1:-1/sess_appl: ev(12=CC_EV_CALL_
DISCONNECT_DONE), cid(44), disp(0)
*Mar  1 15:36:22.956: //44/45F2AAE28044/SSAPP:10002:-1/ssaTraceSct:
cid(44)st(SSA_CS_DISCONNECTING)ev(SSA_EV_CALL_DISCONNECT_DONE)
oldst(SSA_CS_CONF_DESTROYING)cfid(-1)csize(1)in(1)fDest(1)
Router#
*Mar  1 15:36:22.956: //44/45F2AAE28044/SSAPP:10002:-1/ssaDisconnectDone:
```

Related Commands

Command	Description
debug voip ccapi error	Traces error logs in the CCAPI.
voice call debug	Allows configuration of the voice call debug output.

debug voip confmsp

To display debugging information from the Conference Media Service Provider (CONFMSP) and its related applications, use the **debug voip confmsp** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voip confmsp

no debug voip confmsp

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.3(8)T	This command was introduced.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.

Examples

The following is sample output from the **debug voip confmsp** command:

```
Router# debug voip confmsp

CONFMSP debugging is on
.
.
.
00:06:44:confmsp_setup_request:callID (6),
00:06:44:confmsp_setup_request:conf structure  (63DD27E4) created,
00:06:44:confmsp_bridge:confID(4), callIDs(6,5) xmitFunc 61D46D4C, dstIF 64912880

00:06:44:confmsp_bridge:confID(4), callIDs(6,5) event queued
00:06:44:confmsp_act_bridge: state = CONFMSP_STATE_SETUP, event=EV_CONFMSP_BRIDGING
00:06:44:confmsp_act_bridge:codec 1, codec_bytes 160, vad 1
00:06:44:confmsp_act_bridge:codec 1, codec_bytes 160, vad 1
00:06:44:CNFSM:cur_container:confmsp container, cur_state:CONFMSP_STATE_SETUP,
event:EV_CONFMSP_BRIDGING, next_state:CONFMSP_STATE_CONNECT_CONFEREE
00:06:44:confmsp_get_dsmp_req_status:condition to be returned FALSE
00:06:44:confmsp_connect_response:found conf (63DD27E4) dsmp ret is 1

00:06:44:confmsp_act_bridge_success: state = CONFMSP_STATE_CONNECT_CONFEREE,
event=EV_CONFMSP_CNFRE_CONNECT_RESP_SUCCESS confmsp_caps_ind:context = 65241B34

00:06:44:CNFSM:cur_container:confmsp container, cur_state:CONFMSP_STATE_CONNECT_CONFEREE,
event:EV_CONFMSP_CNFRE_CONNECT_RESP_SUCCESS, next_state:CONFMSP_STATE_CONNECTED
00:06:44:confmsp_bridge:confID(5), callIDs(6,7) xmitFunc 61D46D4C, dstIF 64912880

00:06:44:confmsp_bridge:confID(5), callIDs(6,7) event queued
00:06:44:confmsp_act_bridge: state = CONFMSP_STATE_SETUP, event=EV_CONFMSP_BRIDGING
00:06:44:confmsp_act_bridge:codec 1, codec_bytes 160, vad 1
```

```
00:06:44:confmsp_act_bridge:codec 1, codec_bytes 160, vad 1
00:06:44:CNFSM:cur_container:confmsp container, cur_state:CONFMSP_STATE_SETUP,
event:EV_CONFMSP_BRIDGING, next_state:CONFMSP_STATE_CONNECT_CONFEREE
00:06:44:confmsp_get_dsmp_req_status:condition to be returned FALSE
00:06:44:confmsp_connect_response:found conf (6358A338) dsmp ret is 1

00:06:44:confmsp_act_bridge_success: state = CONFMSP_STATE_CONNECT_CONFEREE,
event=EV_CONFMSP_CNFRE_CONNECT_RESP_SUCCESS confmsp_caps_ind:context = 63588E70

00:06:44:CNFSM:cur_container:confmsp container, cur_state:CONFMSP_STATE_CONNECT_CONFEREE,
event:EV_CONFMSP_CNFRE_CONNECT_RESP_SUCCESS, next_state:CONFMSP_STATE_CONNECTED
00:06:44:confmsp_bridge:confID(6), callIDs(6,8) xmitFunc 61D46D4C, dstIF 64912880

00:06:44:confmsp_bridge:confID(6), callIDs(6,8) event queued
00:06:44:confmsp_act_bridge: state = CONFMSP_STATE_SETUP, event=EV_CONFMSP_BRIDGING
00:06:44:confmsp_act_bridge:codec 1, codec_bytes 160, vad 1
00:06:44:confmsp_act_bridge:codec 1, codec_bytes 160, vad 1
00:06:44:CNFSM:cur_container:confmsp container, cur_state:CONFMSP_STATE_SETUP,
event:EV_CONFMSP_BRIDGING, next_state:CONFMSP_STATE_CONNECT_CONFEREE
00:06:44:confmsp_get_dsmp_req_status:condition to be returned FALSE
00:06:44:confmsp_connect_response:found conf (6358CE50) dsmp ret is 1

00:06:44:confmsp_act_bridge_success: state = CONFMSP_STATE_CONNECT_CONFEREE,
event=EV_CONFMSP_CNFRE_CONNECT_RESP_SUCCESS confmsp_caps_ind:context = 63DD2524

00:06:44:CNFSM:cur_container:confmsp container, cur_state:CONFMSP_STATE_CONNECT_CONFEREE,
event:EV_CONFMSP_CNFRE_CONNECT_RESP_SUCCESS, next_state:CONFMSP_STATE_CONNECTED
00:07:28:confmsp_bdrop:confID(4), callIDs(6,5)
00:07:28:confmsp_bdrop:confID(4), callIDs(6,5) event queued
00:07:28:confmsp_act_bdrop: state = CONFMSP_STATE_CONNECTED, event=EV_CONFMSP_BRIDGEDROP
00:07:28:CNFSM:cur_container:confmsp container, cur_state:CONFMSP_STATE_CONNECTED,
event:EV_CONFMSP_BRIDGEDROP, next_state:CONFMSP_STATE_DISCONNECT_CONFEREE
00:07:28:confmsp_get_dsmp_req_status:condition to be returned FALSE
00:07:28:confmsp_disconnect_response:found conf (63DD27E4)

00:07:28:confmsp_connect_response:found conf (63DD27E4) dsmp ret is 10

00:07:28:confmsp_act_bdrop_success: state = CONFMSP_STATE_DISCONNECT_CONFEREE,
event=EV_CONFMSP_CNFRE_DISCONNECT_RESP_SUCCESS
00:07:28:CNFSM:cur_container:confmsp container,
cur_state:CONFMSP_STATE_DISCONNECT_CONFEREE,
event:EV_CONFMSP_CNFRE_DISCONNECT_RESP_SUCCESS, next_state:CONFMSP_STATE_BRIDGE_DROPPED
00:07:28:CNFSM:cur_container:confmsp container, cur_state:CONFMSP_STATE_BRIDGE_DROPPED,
event:CNFSM_LAMBDA_EVENT, next_state:CNFSM_NO_STATE_CHANGE
00:07:29:confmsp_bdrop:confID(6), callIDs(6,8)
00:07:29:confmsp_bdrop:confID(6), callIDs(6,8) event queued
00:07:29:confmsp_bdrop:confID(5), callIDs(6,7)
00:07:29:confmsp_bdrop:confID(5), callIDs(6,7) event queued
00:07:29:confmsp_act_bdrop: state = CONFMSP_STATE_CONNECTED, event=EV_CONFMSP_BRIDGEDROP
00:07:29:CNFSM:cur_container:confmsp container, cur_state:CONFMSP_STATE_CONNECTED,
event:EV_CONFMSP_BRIDGEDROP, next_state:CONFMSP_STATE_DISCONNECT_CONFEREE
00:07:29:confmsp_get_dsmp_req_status:condition to be returned FALSE
00:07:29:confmsp_act_bdrop: state = CONFMSP_STATE_CONNECTED, event=EV_CONFMSP_BRIDGEDROP
00:07:29:CNFSM:cur_container:confmsp container, cur_state:CONFMSP_STATE_CONNECTED,
event:EV_CONFMSP_BRIDGEDROP, next_state:CONFMSP_STATE_DISCONNECT_CONFEREE
00:07:29:confmsp_get_dsmp_req_status:condition to be returned FALSE
00:07:29:confmsp_disconnect_response:found conf (6358CE50)

00:07:29:confmsp_connect_response:found conf (6358CE50) dsmp ret is 10

00:07:29:confmsp_act_bdrop_success: state = CONFMSP_STATE_DISCONNECT_CONFEREE,
event=EV_CONFMSP_CNFRE_DISCONNECT_RESP_SUCCESS
00:07:29:CNFSM:cur_container:confmsp container,
cur_state:CONFMSP_STATE_DISCONNECT_CONFEREE,
event:EV_CONFMSP_CNFRE_DISCONNECT_RESP_SUCCESS, next_state:CONFMSP_STATE_BRIDGE_DROPPED
00:07:29:confmsp_act_terminate: state = CONFMSP_STATE_BRIDGE_DROPPED,
event=CNFSM_LAMBDA_EVENT
00:07:29:CNFSM:cur_container:confmsp container, cur_state:CONFMSP_STATE_BRIDGE_DROPPED,
event:CNFSM_LAMBDA_EVENT, next_state:CNFSM_NULL_STATE
00:07:29:confmsp_free_conf:freeing 6358CE50

00:07:29:confmsp_disconnect_response:found conf (6358A338)
```

```
00:07:29:confmsp_connect_response:found conf (6358A338) dsmp ret is 10

00:07:29:confmsp_act_bdrop_success: state = CONFMSP_STATE_DISCONNECT_CONFEREE,
event=EV_CONFMSP_CNFRE_DISCONNECT_RESP_SUCCESS
00:07:29:CNFSM:cur_container:confmsp container,
cur_state:CONFMSP_STATE_DISCONNECT_CONFEREE,
event:EV_CONFMSP_CNFRE_DISCONNECT_RESP_SUCCESS, next_state:CONFMSP_STATE_BRIDGE_DROPPED
00:07:29:confmsp_act_terminate: state = CONFMSP_STATE_BRIDGE_DROPPED,
event=CNFSM_LAMBDA_EVENT
00:07:29:CNFSM:cur_container:confmsp container, cur_state:CONFMSP_STATE_BRIDGE_DROPPED,
event:CNFSM_LAMBDA_EVENT, next_state:CNFSM_NULL_STATE
00:07:29:confmsp_free_conf:freeing 6358A338

00:07:29:confmsp_disconnect:callID (6)
00:07:29:confmsp_disconnect:callID (6) event queued
00:07:29:confmsp_act_disconnected: state = CONFMSP_STATE_BRIDGE_DROPPED,
event=EV_CONFMSP_DISCONNECT
00:07:29:CNFSM:cur_container:confmsp container, cur_state:CONFMSP_STATE_BRIDGE_DROPPED,
event:EV_CONFMSP_DISCONNECT, next_state:CNFSM_NULL_STATE
00:07:29:confmsp_free_conf:freeing 63DD27E4
```

debug voip dcapi

To debug the device control application programming interface (DCAPI), use the **debug voip dcapi**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voip dcapi [**error**| **inout**]

no debug voip dcapi [**error**| **inout**]

Syntax Description

error	(Optional) Displays error logs in the DCAPI.
inout	(Optional) Displays the execution path through the DCAPI.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(14)T	This command was introduced.

Usage Guidelines

The **debug voip dcapi error** command traces the error logs in the DCAPI, which is the software layer that interfaces the SCCP Telephony Control Application (STCAPP) with the Cisco CallManager using the Skinny Client Control Protocol (SCCP). Error logs are generated during normal call processing when there are insufficient resources, or when there are problems in the device control API. This debug command shows error events or unexpected behavior in system software.

The **debug voip dcapi inout** command shows how a call executes through the software. This command traces the execution path through the DCAPI during communications with the SCCP service provider interface (SPI) and the call control API (CCAPI) that controls the physical voice port. You can use the output from this command to understand how devices are being handled by the APIs and to see the call setup and teardown operations performed on the telephony call leg.

Examples

Following is sample output from the **debug voip dcapi inout**command during call setup:

```
Router# debug voip dcapi inout
*Jan 27 16:26:23.957: dc_api_device_set_ringer_res: Set Ringer message success
*Jan 27 16:26:23.957: //-1/xxxxxxxxxxxx/CCAPI/dc_api_device_stop_tone_res:

*Jan 27 16:26:23.957: dc_api_device_stop_tone_res: Stop Tone message success
*Jan 27 16:26:23.957: //-1/xxxxxxxxxxxx/CCAPI/dc_api_media_open_rcv_channel:

*Jan 27 16:26:23.957: dc_api_media_open_rcv_channel: evt DC_EV_MEDIA_OPEN_RCV_CHNL is
successsfully enqueued to app
*Jan 27 16:26:23.957: //-1/xxxxxxxxxxxx/CCAPI/dc_api_device_stop_tone_res:

*Jan 27 16:26:23.957: dc_api_device_stop_tone_res: Stop Tone message success
```

```
*Jan 27 16:26:23.957: //-1/xxxxxxxxxxxx/CCAPI/dc_api_device_call_state_res:

*Jan 27 16:26:23.957: dc_api_device_call_state_res: Call State message success
*Jan 27 16:26:23.957: //-1/xxxxxxxxxxxx/CCAPI/dc_api_device_call_info_res:
```

The table below describes the significant fields shown in the display.

Table 60: debug voip dcapi Field Descriptions

Field	Description
nn :*nn* :*nn* :	Timestamp time in hours (military format), minutes, and seconds that indicates when the DCAPI event occurred.
dc_api_*message:*	The DCAPI event in which the SCCP SPI translation occurred.

Related Commands

Command	Description
debug voip application stcapp events	Debugs STCAPP events.
debug voip application stcapp functions	Debugs STCAPP functions.

debug voip dialpeer

To display information about the voice dial peers, use the **debug voip dialpeer** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voip dialpeer [**all**| **default**| **detail**| **error** [**call [informational]**| **software [informational]**]| **function**| **inout**]

no debug voip dialpeer

Syntax Description

all	(Optional) Displays all dialpeer debugging messages.
default	(Optional) Displays dialpeer inout and error debugging messages. This option also runs if no keywords are added.
detail	(Optional) Displays detailed dialpeer information.
error	(Optional) Displays dialpeer error messages.
call	(Optional) Displays call processing errors.
informational	(Optional) Displays minor errors and major errors. Without the **informational** keyword, only major errors are displayed.
software	(Optional) Displays software processing errors.
function	(Optional) Displays dialpeer functions.
inout	(Optional) Displays dialpeer in/out functions.

Command Default

Debugging is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(8)T	This command replaces the **debug dialpeer** command.

Usage Guidelines

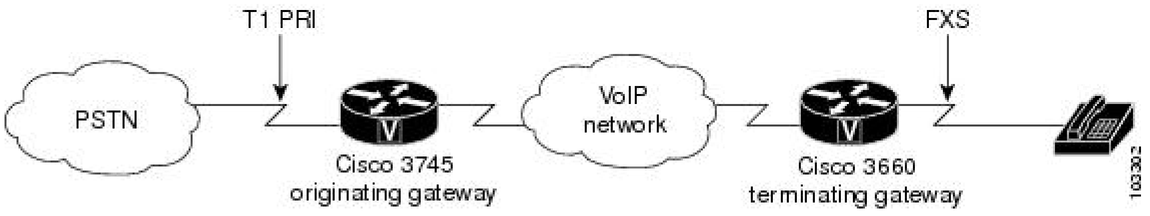
Disable console logging and use buffered logging before using the **debug voip dialpeer**command. Using the **debug voip dialpeer** command generates a large volume of debugging messages, which can affect router performance.

Examples

The following examples show output for variations of the **debug voip dialpeer** command:

For these examples, the topology shown in the figure below is used.

Figure 2: Network Topology for debug voip dialpeer Output Examples


T1 PRI
FXS
PSTN
VoIP network
Cisco 3745 originating gateway
Cisco 3660 terminating gateway
103302

Examples

```
Router# debug voip dialpeer detail

voip dialpeer detail debugging is on
Router#
```

The following event identifies the called number:

```
*Apr 18 21:07:35.291: //-1/xxxxxxxxxxxx/DPM/dpAssociateIncomingPeerCore:
   Match Rule=DP_MATCH_INCOMING_DNIS; Called Number=83103
*Apr 18 21:07:35.291: //-1/xxxxxxxxxxxx/DPM/dpMatchPeertype:
   Is Incoming=TRUE, Number Expansion=FALSE
*Apr 18 21:07:35.291: //-1/xxxxxxxxxxxx/DPM/dpMatchCore:
   Dial String=83103, Expanded String=83103, Calling Number=
   Timeout=TRUE, Is Incoming=TRUE, Peer Info Type=DIALPEER_INFO_SPEECH
```

The following event identifies the incoming dial peer and shows that it has been matched:

```
*Apr 18 21:07:35.291: //-1/xxxxxxxxxxxx/DPM/MatchNextPeer:
   Result=Success(0); Incoming Dial-peer=1 Is Matched
*Apr 18 21:07:35.291: //-1/xxxxxxxxxxxx/DPM/dpAssociateIncomingPeerCore:
   Match Rule=DP_MATCH_INCOMING_DNIS; Called Number=83103
*Apr 18 21:07:35.291: //-1/xxxxxxxxxxxx/DPM/dpMatchPeertype:
   Is Incoming=TRUE, Number Expansion=FALSE
*Apr 18 21:07:35.291: //-1/xxxxxxxxxxxx/DPM/dpMatchCore:
   Dial String=83103, Expanded String=83103, Calling Number=
   Timeout=TRUE, Is Incoming=TRUE, Peer Info Type=DIALPEER_INFO_FAX
*Apr 18 21:07:35.291: //-1/xxxxxxxxxxxx/DPM/dpMatchCore:
   Result=-1
*Apr 18 21:07:35.291: //-1/xxxxxxxxxxxx/DPM/dpAssociateIncomingPeerCore:
   Match Rule=DP_MATCH_ANSWER; Calling Number=4085550111
*Apr 18 21:07:35.291: //-1/xxxxxxxxxxxx/DPM/dpMatchPeertype:
   Is Incoming=TRUE, Number Expansion=FALSE
*Apr 18 21:07:35.291: //-1/xxxxxxxxxxxx/DPM/dpMatchCore:
   Dial String=, Expanded String=, Calling Number=4085550111T
   Timeout=TRUE, Is Incoming=TRUE, Peer Info Type=DIALPEER_INFO_FAX
*Apr 18 21:07:35.291: //-1/xxxxxxxxxxxx/DPM/dpMatchCore:
   Result=-1
*Apr 18 21:07:35.291: //-1/xxxxxxxxxxxx/DPM/dpAssociateIncomingPeerCore:
   Match Rule=DP_MATCH_ORIGINATE; Calling Number=4085550111
*Apr 18 21:07:35.291: //-1/xxxxxxxxxxxx/DPM/dpMatchPeertype:
   Is Incoming=TRUE, Number Expansion=FALSE
*Apr 18 21:07:35.291: //-1/xxxxxxxxxxxx/DPM/dpMatchCore:
   Dial String=, Expanded String=, Calling Number=4085550111T
   Timeout=TRUE, Is Incoming=TRUE, Peer Info Type=DIALPEER_INFO_FAX
*Apr 18 21:07:35.295: //-1/xxxxxxxxxxxx/DPM/dpMatchCore:
   Result=-1
```

The following event shows the number expansion. This is the only dial peer debug command that shows the number expansion.

```
*Apr 18 21:07:35.295: //-1/xxxxxxxxxxxx/DPM/dpMatchCore:
   Dial String=83103, Expanded String=3600, Calling Number=
   Timeout=TRUE, Is Incoming=FALSE, Peer Info Type=DIALPEER_INFO_SPEECH
```

The next few lines show matching for the outgoing dial peer. These lines show the matching sequence if the first match is not available.

```
*Apr 18 21:07:35.295: //-1/xxxxxxxxxxxx/DPM/MatchNextPeer:
   Result=Success(0); Outgoing Dial-peer=3600 Is Matched
*Apr 18 21:07:35.295: //-1/xxxxxxxxxxxx/DPM/MatchNextPeer:
   Result=Success(0); Outgoing Dial-peer=36 Is Matched
*Apr 18 21:07:35.295: //-1/xxxxxxxxxxxx/DPM/MatchNextPeer:
   Result=Success(0); Outgoing Dial-peer=360 Is Matched
*Apr 18 21:07:35.295: //-1/23ED4B1B8010/DPM/dpMatchCore:
   Dial String=83103, Expanded String=3600, Calling Number=
   Timeout=TRUE, Is Incoming=FALSE, Peer Info Type=DIALPEER_INFO_SPEECH
```

Examples

```
Router# debug voip dialpeer inout

voip dialpeer inout debugging is on
```

The following event shows the calling and called numbers:

```
*May  1 19:32:11.731: //-1/6372E2598012/DPM/dpAssociateIncomingPeerCore:
   Calling Number=4085550111, Called Number=3600, Voice-Interface=0x0,
   Timeout=TRUE, Peer Encap Type=ENCAP_VOIP, Peer Search Type=PEER_TYPE_VOICE,
   Peer Info Type=DIALPEER_INFO_SPEECH
```

The following event shows the incoming dial peer:

```
*May  1 19:32:11.731: //-1/6372E2598012/DPM/dpAssociateIncomingPeerCore:
   Result=Success(0) after DP_MATCH_INCOMING_DNIS; Incoming Dial-peer=100
*May  1 19:32:11.731: //-1/6372E2598012/DPM/dpAssociateIncomingPeerCore:
   Calling Number=4085550111, Called Number=3600, Voice-Interface=0x0,
   Timeout=TRUE, Peer Encap Type=ENCAP_VOIP, Peer Search Type=PEER_TYPE_VOICE,
   Peer Info Type=DIALPEER_INFO_SPEECH
*May  1 19:32:11.731: //-1/6372E2598012/DPM/dpAssociateIncomingPeerCore:
   Result=Success(0) after DP_MATCH_INCOMING_DNIS; Incoming Dial-peer=100
*May  1 19:32:11.735: //-1/6372E2598012/DPM/dpMatchPeersCore:
   Calling Number=, Called Number=3600, Peer Info Type=DIALPEER_INFO_SPEECH
*May  1 19:32:11.735: //-1/6372E2598012/DPM/dpMatchPeersCore:
   Match Rule=DP_MATCH_DEST; Called Number=3600
*May  1 19:32:11.735: //-1/6372E2598012/DPM/dpMatchPeersCore:
   Result=Success(0) after DP_MATCH_DEST
*May  1 19:32:11.735: //-1/6372E2598012/DPM/dpMatchPeersMoreArg:
   Result=SUCCESS(0)
```

The following event shows the matched dial peers in the order of priority:

```
List of Matched Outgoing Dial-peer(s):
     1: Dial-peer Tag=3600
     2: Dial-peer Tag=36
```

Related Commands

Command	Description
call-block (dial peer)	Enables blocking of incoming calls on the dial peer.

Command	Description
carrier-id (dial-peer)	Identifies the carrier handling the incoming call.
session target (ENUM)	Specifies the ENUM search table for the target session.
show dial-peer voice	Displays the configuration of the dial peer.
translation-profile (dial-peer)	Assigns a translation profile to the dial peer.
trunkgroup (dial-peer)	Assigns a trunk group to the dial peer.
trunk-group-label (dial-peer)	Identifies the trunk group handling the incoming call.

debug voip dsm

To troubleshoot the DSP stream manager (DSM) subsystem, use the **debug voip dsm** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voip dsm [**all**| **dsp**| **error**| **rtp**| **session**| **stats**| **tone**| **vofr**]

no debug voip dsm

Syntax Description

all	(Optional) Displays all DSM debugging messages.
dsp	(Optional) Enables a digital signal processor (DSP) message trace.
error	(Optional) Displays DSM error messages.
rtp	(Optional) Enables Real-Time Protocol (RTP) debugging on DSM.
session	(Optional) Enables session debugging.
stats	(Optional) Displays DSM statistics.
tone	(Optional) Displays tone debugging.
vofr	(Optional) Enables Voice over Frame Relay (VoFR) debugging on the VPM.

Command Default

Debugging is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(8)T	This command replaces the **debug vtsp dsp** command.
12.3(14)T	The **vofr** keyword is no longer available in Cisco IOS Release 12.3(14)T.

Usage Guidelines

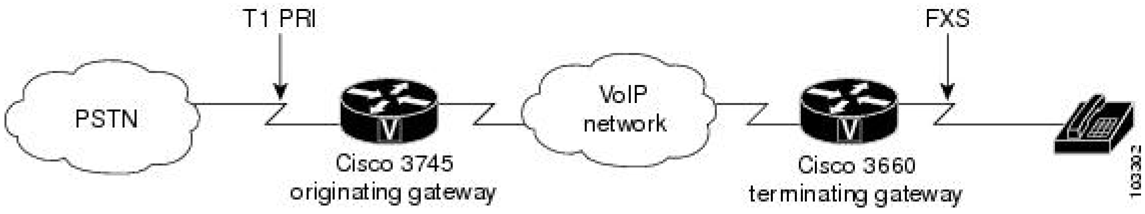
To debug VoIP calls, use this command in conjunction with **debug voip vtsp** command and **debug voip dsmp** commands. All the related information for media processing is now available by using Distributed Stream Media Processor (DSMP) . DSM is responsible for creating streams and issuing connections between them.

Examples

The following examples show output for variations of the **debug voip dsm** command:

For these examples, the topology shown in the figure below is used.

Figure 3: Network Topology for debug voip dsm Examples


T1 PRI
FXS
PSTN
VoIP network
Cisco 3745 originating gateway
Cisco 3660 terminating gateway
103302

Examples

```
Router# debug voip dsm session

DSP Stream Manager session debugging is on
*Apr 18 21:15:39.679: //-1/44A507668015/DSM:(2/1:23):0:0:0/dsm_start_basic_sm: .
*Apr 18 21:15:39.679: //-1/44A507668015/DSM:(2/1:23):0:8:4/dsm_start_basic_sm: dsp resource
 manager opened. ret 4
*Apr 18 21:15:39.679: //-1/44A507668015/DSM:(2/1:23):0:8:4/dsm_open_voice_and_set_params:
.
*Apr 18 21:15:39.679: //-1/44A507668015/DSM:(2/1:23):0:8:4/set_playout_dmgr: playout default
```

The following event shows the echo cancellation setting:

```
*Apr 18 21:15:39.679: //-1/44A507668015/DSM:(2/1:23):0:8:4/dsm_dsp_echo_canceller_control:
 echo_cancel: 1
*Apr 18 21:15:39.695: //43/44A507668015/DSM:(2/1:23):0:8:4/dsp_stream_mgr_play_tone: .
*Apr 18 21:15:39.695: //43/44A507668015/DSM:(2/1:23):0:8:4/dsm_exec: [Feat SM: S:NONE B SM:
 S:S_DSM_INIT E:E_DSM_CC_GEN_TONE]
*Apr 18 21:15:39.695: //43/44A507668015/DSM:(2/1:23):0:8:4/dsm_act_gen_tone: Tone is not
on, ignoring
*Apr 18 21:15:39.695: //43/44A507668015/DSM:(2/1:23):0:8:4/dsm_exec: [B SM: R:FSM_OK ->
S:S_DSM_INIT]
*Apr 18 21:15:39.695: //43/44A507668015/DSM:(2/1:23):0:8:4/dsm_exec: [Feat SM: S:NONE B SM:
 S:S_DSM_INIT E:E_DSM_CC_BRIDGE]
*Apr 18 21:15:39.695: //43/44A507668015/DSM:(2/1:23):0:8:4/dsm_act_bridge: .
```

The following event indicates that modem relay is not supported on the specified port, which is port 2/1:23:

```
*Apr 18 21:15:39.695: //43/44A507668015/DSM:(2/1:23):0:8:4/dsm_act_bridge: Modem Relay NOT
 Supported on this end-point/voice-port.
disabling it...
*Apr 18 21:15:39.695: //43/44A507668015/DSM:(2/1:23):0:8:4/dsm_exec: [B SM: R:FSM_OK ->
S:S_DSM_BRIDGING]
*Apr 18 21:15:39.695: //43/44A507668015/DSM:(2/1:23):0:8:4/dsm_exec: [Feat SM: S:NONE B SM:
 S:S_DSM_BRIDGING E:E_DSM_CC_CAPS_IND]
*Apr 18 21:15:39.695: //43/44A507668015/DSM:(2/1:23):0:8:4/dsm_act_caps_ind: .
*Apr 18 21:15:39.695: //43/44A507668015/DSM:(2/1:23):0:8:4/dsm_act_caps_ind: RTP
PT:NTE[101],NSE[100],FaxInd[96],FaxAck[97],CiscoDT
121],FaxRelay[122],CASsig[123],ClearChan[125],PCMu[0],PCMa[8]Codec[4], TxDynamicPayload[0],
 RxDynamicPayload[0]
*Apr 18 21:15:39.695: //43/44A507668015/DSM:(2/1:23):0:8:4/dsm_act_caps_ind: dtmf relay:
mode=1, codec=1
*Apr 18 21:15:39.695: //43/44A507668015/DSM:(2/1:23):0:8:4/dsm_act_caps_ind: Modem Relay
NOT Supported on this end-point/voice-port
disabling it...
*Apr 18 21:15:39.695: //43/44A507668015/DSM:(2/1:23):0:8:4/dsm_act_caps_ind: passthrough:
cap_modem_proto 4,cap_modem_codec 1, cap_
em_redundancy 0, payload 103, modem_relay 0, gw-xid 0
*Apr 18 21:15:39.695: //43/44A507668015/DSM:(2/1:23):0:8:4/dsm_debug_caps_encap: Encap 1,
Vad 2, Codec 0x4, CodecBytes 20,
             FaxRate 1, FaxBytes 20, FaxNsf 0xAD0051
             SignalType 2
             DtmfRelay 1, Modem 2, SeqNumStart 0x5A1
*Apr 18 21:15:39.695: //43/44A507668015/DSM:(2/1:23):0:8:4/dsm_act_caps_ind: FORKING
Parameters are forking mask: 0, simple_forking
dec_mask: 0, complex_forking_codec_mask 0
*Apr 18 21:15:39.695: //43/44A507668015/DSM:(2/1:23):0:8:4/dsm_act_caps_ind: playout: [
mode:1,init:60, min:40, max:200]. data_mode
*Apr 18 21:15:39.695: //43/44A507668015/DSM:(2/1:23):0:8:4/dsm_exec: [B SM: R:FSM_OK ->
S:S_DSM_BRIDGING]
*Apr 18 21:15:39.695: //43/44A507668015/DSM:(2/1:23):0:8:4/dsm_exec: [Feat SM: S:NONE B SM:
 S:S_DSM_BRIDGING E:E_DSM_CC_CAPS_ACK]
*Apr 18 21:15:39.695: //43/44A507668015/DSM:(2/1:23):0:8:4/dsm_act_caps_ack: .
*Apr 18 21:15:39.695: //43/44A507668015/DSM:(2/1:23):0:8:4/dsm_act_caps_ack: passthrough:
cap_modem_proto 4, cap_modem_codec 1, cap
dem_redundancy 0, payload 103, modem_relay 0, gw-xid 0
*Apr 18 21:15:39.695: //43/44A507668015/DSM:(2/1:23):0:8:4/dsm_act_caps_ack: Named Telephone
 Event payload rcv 101, tx 101
*Apr 18 21:15:39.695: //43/44A507668015/DSM:(2/1:23):0:8:4/dsm_switch_codec: Required codec
 is 16,  current dsp codec is -1
*Apr 18 21:15:39.695: //43/44A507668015/DSM:(2/1:23):0:8:4/dsm_switch_codec: codec = 16
*Apr 18 21:15:39.695: //43/44A507668015/DSM:(2/1:23):0:8:4/dsm_exec: [B SM: R:FSM_OK ->
S:S_DSM_PENDING_CODEC_SWITCH]
*Apr 18 21:15:39.695: //43/44A507668015/DSM:(2/1:23):0:8:4/dsm_exec: [Feat SM: S:NONE B SM:
 S:S_DSM_PENDING_CODEC_SWITCH E:E_DSM_DS_PEND_SUCCESS]
*Apr 18 21:15:39.695: //43/44A507668015/DSM:(2/1:23):0:8:4/dsm_act_pend_codec_success: .
*Apr 18 21:15:39.695: //43/44A507668015/DSM:(2/1:23):0:8:4/dsm_open_voice_and_set_params:
*Apr 18 21:15:39.695: //43/44A507668015/DSM:(2/1:23):0:8:4/set_playout_dmgr: playout default
```

The following event shows the echo cancellation setting:

```
*Apr 18 21:15:39.695: //43/44A507668015/DSM:(2/1:23):0:8:4/dsm_dsp_echo_canceller_control:
 echo_cancel: 1
```

The following event shows that the codec has changed:

```
*Apr 18 21:15:39.695:
//43/44A507668015/DSM:(2/1:23):0:8:4/dsm_setup_stream_after_switch_codec_succ: codec change
 success
*Apr 18 21:15:39.695: //43/44A507668015/DSM:(2/1:23):0:8:4/dsm_add_fork: .
*Apr 18 21:15:39.695: //43/44A507668015/DSM:(2/1:23):0:8:4/dsm_update_fork_info: add_fork=0
*Apr 18 21:15:39.695: //43/44A507668015/DSM:(2/1:23):0:8:4/dsm_get_xmit_info_node: .
*Apr 18 21:15:39.695: //43/44A507668015/DSM:(2/1:23):0:8:4/dsm_update_fork_info:  xmit func
 is 61A7CDC4, context is 64F42DA0 peer_c
_id: 44, stream_count: 1, update_flag 0
*Apr 18 21:15:39.695: //43/44A507668015/DSM:(2/1:23):0:8:4/dsm_update_fork_info: The stream
 bit-mask is 1
*Apr 18 21:15:39.695: //43/44A507668015/DSM:(2/1:23):0:8:4/dsm_update_fork_info: The stream
 type is 0
*Apr 18 21:15:39.695: //43/44A507668015/DSM:(2/1:23):0:8:4/dsm_update_fork_info: The logical
 ssrc is 64 for stream 0
*Apr 18 21:15:39.699: //43/44A507668015/DSM:(2/1:23):0:8:4/dsm_update_stream_count:
*Apr 18 21:15:39.699:  g711_voice_count=0 g711_avt_count = 0
 g711_voice_avt_count = 0 complex_voice_count = 1
 complex_avt_count = 0 complex_voice_avt_count = 0
*Apr 18 21:15:39.699: //43/44A507668015/DSM:(2/1:23):0:8:4/dsm_exec: [B SM: R:FSM_OK ->
S:S_DSM_BRIDGED]
```

Examples

```
Router# debug voip dsm stats

DSP Stream Manager stats debugging is on
```

The following event shows that the DSM is requesting statistics:

```
*May  1 19:40:09.775: //43/7EE6F84B8016/DSM:(4/0/0):-1:1:1/dsm_exec: [Feat SM: S:NONE B SM:
 S:S_DSM_BRIDGED E:E_DSM_CC_REQ_PACK_STAT]
*May  1 19:40:09.775: //43/7EE6F84B8016/DSM:(4/0/0):-1:1:1/dsm_act_cc_stats_req: .
```

```
*May  1 19:40:09.775: //43/7EE6F84B8016/DSM:(4/0/0):-1:1:1/dsm_act_cc_stats_req:
-1->dmgr=0x645461E0, stats_reqs=0
```

The following events show statistics for DSM transmission, reception, delay, and errors:

```
*May  1 19:40:09.775: //43/7EE6F84B8016/DSM:(4/0/0):-1:1:1/dsm_exec: [B SM: R:FSM_OK ->
S:S_DSM_BRIDGED]
*May  1 19:40:09.775: //43/7EE6F84B8016/DSM:(4/0/0):-1:1:1/dsm_exec: [Feat SM: S:NONE B SM:
 S:S_DSM_BRIDGED E:E_DSM_DSP_GET_TX]
*May  1 19:40:09.775: //43/7EE6F84B8016/DSM:(4/0/0):-1:1:1/dsm_act_packet_stats_res: .
*May  1 19:40:09.775: //43/7EE6F84B8016/DSM:(4/0/0):-1:1:1/dsm_act_packet_stats_res:
-4->dmgr=0x645461E0,stats_reqs=3
*May  1 19:40:09.775: //43/7EE6F84B8016/DSM:(4/0/0):-1:1:1/dsm_exec: [B SM: R:FSM_OK ->
S:S_DSM_BRIDGED]
*May  1 19:40:09.775: //43/7EE6F84B8016/DSM:(4/0/0):-1:1:1/dsm_exec: [Feat SM: S:NONE B SM:
 S:S_DSM_BRIDGED E:E_DSM_DSP_GET_RX]
*May  1 19:40:09.775: //43/7EE6F84B8016/DSM:(4/0/0):-1:1:1/dsm_act_packet_stats_res: .
*May  1 19:40:09.775: //43/7EE6F84B8016/DSM:(4/0/0):-1:1:1/dsm_act_packet_stats_res:
-4->dmgr=0x645461E0,stats_reqs=2
*May  1 19:40:09.775: //43/7EE6F84B8016/DSM:(4/0/0):-1:1:1/dsm_exec: [B SM: R:FSM_OK ->
S:S_DSM_BRIDGED]
*May  1 19:40:09.775: //43/7EE6F84B8016/DSM:(4/0/0):-1:1:1/dsm_exec: [Feat SM: S:NONE B SM:
 S:S_DSM_BRIDGED E:E_DSM_DSP_GET_VP_DELAY]
*May  1 19:40:09.775: //43/7EE6F84B8016/DSM:(4/0/0):-1:1:1/dsm_act_packet_stats_res: .
*May  1 19:40:09.775: //43/7EE6F84B8016/DSM:(4/0/0):-1:1:1/dsm_act_packet_stats_res:
-4->dmgr=0x645461E0,stats_reqs=1
*May  1 19:40:09.775: //43/7EE6F84B8016/DSM:(4/0/0):-1:1:1/dsm_exec: [B SM: R:FSM_OK ->
S:S_DSM_BRIDGED]
*May  1 19:40:09.775: //43/7EE6F84B8016/DSM:(4/0/0):-1:1:1/dsm_exec: [Feat SM: S:NONE B SM:
 S:S_DSM_BRIDGED E:E_DSM_DSP_GET_VP_ERROR]
*May  1 19:40:09.775: //43/7EE6F84B8016/DSM:(4/0/0):-1:1:1/dsm_act_packet_stats_res: .
Router#
*May  1 19:40:09.775: //43/7EE6F84B8016/DSM:(4/0/0):-1:1:1/dsm_act_packet_stats_res:
-4->dmgr=0x645461E0,stats_reqs=0
```

Examples

The following is sample output from the **debug voip dsm** command, with Cisco IOS Release 12.3(14)T software, when a VoIP call is in transition to the connected state:

```
Router# debug voip dsm

*Jun  8 20:10:33.205: //28/6F1FA7868003/DSM:(2/0:23):-1/dsp_stream_mgr_create: .
*Jun  8 20:10:33.209: //28/6F1FA7868003/DSM:(2/0:23):-1/set_echo_canceller_data: echo_cancel:
 1
*Jun  8 20:10:33.209: //28/6F1FA7868003/DSM:(2/0:23):-1/set_echo_canceller_data: echo_flags:
 55, echo_len: 512
*Jun  8 20:10:33.217: //28/6F1FA7868003/DSM:(2/0:23):-1/dsmp_dsmapi_reserve_resource_cb: .
*Jun  8 20:10:33.217: //28/6F1FA7868003/DSM:(2/0:23):-1/dsp_stream_mgr_set_elog_enable: .
*Jun  8 20:10:33.477: //28/6F1FA7868003/DSM:(2/0:23):-1/dsp_stream_mgr_reinit_platform_info:
 .
*Jun  8 20:10:33.477: //28/6F1FA7868003/DSM:(2/0:23):-1/dsmp_dsmapi_reserve_resource_cb: .
*Jun  8 20:10:33.485: //28/6F1FA7868003/DSM:(2/0:23):-1/dsp_stream_mgr_bridge: .
*Jun  8 20:10:33.485: //28/6F1FA7868003/DSM:(2/0:23):-1/dsp_stream_mgr_bridge:
*Jun  8 20:10:33.485:  dsp_stream_mgr_bridge, src_call_id 28, dst_call_id 27
*Jun  8 20:10:33.485: //28/6F1FA7868003/DSM:(2/0:23):-1/dsp_stream_mgr_bridge: creating
packet streams
*Jun  8 20:10:33.489: //28/6F1FA7868003/DSM:(2/0:23):-1/dsm_insert_conn_array_entry:
dmgr->connArr.count = 2
*Jun  8 20:10:33.489: //28/6F1FA7868003/DSM:(2/0:23):-1/dsp_stream_mgr_do_caps_ind: .
*Jun  8 20:10:33.493: //28/6F1FA7868003/DSM:(2/0:23):-1/dsp_stream_mgr_cap_ind_ack:
src_call_id = 28, dst_call_id = 27, is_cap_ack = 1
*Jun  8 20:10:33.493: //28/6F1FA7868003/DSM:(2/0:23):-1/dsp_stream_mgr_cap_ind_ack:
src_call_id = 28, dst_call_id = 27, is_cap_ack = 0
*Jun  8 20:10:33.493: //28/6F1FA7868003/DSM:(2/0:23):-1/dsm_caps_ind_negotiation: Caps in
caps indication:
*Jun  8 20:10:33.497: //28/6F1FA7868003/DSM:(2/0:23):-1/dsm_caps_dump: playout: [
mode:1,init:60, min:40, max:250]. data_mode:0
*Jun  8 20:10:33.497: //28/6F1FA7868003/DSM:(2/0:23):-1/dsm_caps_dump: RTP
PT:NTE[101],NSE[100],FaxInd[96],FaxAck[97],CiscoDTMF[121],FaxRelay[122],CASsig[123],ClearChan[125],PCMu[0],PCMa[8]Codec[4],
 TxDynamicPayload[0], RxDynamicPayload[0]
*Jun  8 20:10:33.497: //28/6F1FA7868003/DSM:(2/0:23):-1/dsm_caps_dump: passthrough:
```

```
cap_modem_proto 4,cap_modem_codec 1, cap_modem_redundancy 1, payload 100, modem_relay 0,
gw-xid 0
*Jun  8 20:10:33.497: //28/6F1FA7868003/DSM:(2/0:23):-1/dsm_debug_caps_encap: Encap 1, Vad
 2, Codec 0x4, CodecBytes 20,
             FaxRate 2, FaxBytes 20, FaxNsf 0xAD0051
             SignalType 2
             DtmfRelay 1, Modem 2, SeqNumStart 0x0
*Jun  8 20:10:33.497: //28/6F1FA7868003/DSM:(2/0:23):-1/dsm_caps_ind_negotiation: Caps after
 caps negotiation:
*Jun  8 20:10:33.497: //28/6F1FA7868003/DSM:(2/0:23):-1/dsm_caps_dump: playout: [
mode:1,init:60, min:40, max:250]. data_mode:0
*Jun  8 20:10:33.497: //28/6F1FA7868003/DSM:(2/0:23):-1/dsm_caps_dump: RTP
PT:NTE[101],NSE[100],FaxInd[96],FaxAck[97],CiscoDTMF[121],FaxRelay[122],CASsig[123],ClearChan[125],PCMu[0],PCMa[8]Codec[4],
 TxDynamicPayload[0], RxDynamicPayload[0]
*Jun  8 20:10:33.501: //28/6F1FA7868003/DSM:(2/0:23):-1/dsm_caps_dump: passthrough:
cap_modem_proto 4,cap_modem_codec 1, cap_modem_redundancy 1, payload 100, modem_relay 0,
gw-xid 0
*Jun  8 20:10:33.501: //28/6F1FA7868003/DSM:(2/0:23):-1/dsm_debug_caps_encap: Encap 1, Vad
 2, Codec 0x4, CodecBytes 20,
             FaxRate 2, FaxBytes 20, FaxNsf 0xAD0051
             SignalType 2
             DtmfRelay 1, Modem 2, SeqNumStart 0x0
*Jun  8 20:10:33.501: //28/6F1FA7868003/DSM:(2/0:23):-1/dsp_stream_mgr_cap_ind_ack: packet
 streams already created during bridging
*Jun  8 20:10:33.501: //28/6F1FA7868003/DSM:(2/0:23):-1/dsm_caps_ind_negotiation: Caps in
caps indication:
*Jun  8 20:10:33.501: //28/6F1FA7868003/DSM:(2/0:23):-1/dsm_caps_dump: playout: [
mode:1,init:60, min:40, max:250]. data_mode:0
*Jun  8 20:10:33.501: //28/6F1FA7868003/DSM:(2/0:23):-1/dsm_caps_dump: RTP
PT:NTE[101],NSE[100],FaxInd[96],FaxAck[97],CiscoDTMF[121],FaxRelay[122],CASsig[123],ClearChan[125],PCMu[0],PCMa[8]Codec[4],
 TxDynamicPayload[0], RxDynamicPayload[0]
*Jun  8 20:10:33.505: //28/6F1FA7868003/DSM:(2/0:23):-1/dsm_caps_dump: passthrough:
cap_modem_proto 4,cap_modem_codec 1, cap_modem_redundancy 1, payload 100, modem_relay 0,
gw-xid 0
*Jun  8 20:10:33.505: //28/6F1FA7868003/DSM:(2/0:23):-1/dsm_debug_caps_encap: Encap 1, Vad
 2, Codec 0x4, CodecBytes 20,
             FaxRate 2, FaxBytes 20, FaxNsf 0xAD0051
             SignalType 2
             DtmfRelay 1, Modem 2, SeqNumStart 0x0
*Jun  8 20:10:33.505: //28/6F1FA7868003/DSM:(2/0:23):-1/dsm_caps_ind_negotiation: Caps after
 caps negotiation:
*Jun  8 20:10:33.505: //28/6F1FA7868003/DSM:(2/0:23):-1/dsm_caps_dump: playout: [
mode:1,init:60, min:40, max:250]. data_mode:0
*Jun  8 20:10:33.505: //28/6F1FA7868003/DSM:(2/0:23):-1/dsm_caps_dump: RTP
PT:NTE[101],NSE[100],FaxInd[96],FaxAck[97],CiscoDTMF[121],FaxRelay[122],CASsig[123],ClearChan[125],PCMu[0],PCMa[8]Codec[4],
 TxDynamicPayload[0], RxDynamicPayload[0]
*Jun  8 20:10:33.505: //28/6F1FA7868003/DSM:(2/0:23):-1/dsm_caps_dump: passthrough:
cap_modem_proto 4,cap_modem_codec 1, cap_modem_redundancy 1, payload 100, modem_relay 0,
gw-xid 0
*Jun  8 20:10:33.505: //28/6F1FA7868003/DSM:(2/0:23):-1/dsm_debug_caps_encap: Encap 1, Vad
 2, Codec 0x4, CodecBytes 20,
             FaxRate 2, FaxBytes 20, FaxNsf 0xAD0051
             SignalType 2
             DtmfRelay 1, Modem 2, SeqNumStart 0x0
*Jun  8 20:10:33.509: //28/6F1FA7868003/DSM:(2/0:23):-1/dsmp_dsmapi_connect_cb: .
*Jun  8 20:10:36.229: //28/6F1FA7868003/DSM:(2/0:23):-1/dsp_stream_mgr_req_stats: .
*Jun  8 20:10:36.233: //28/6F1FA7868003/DSM:(2/0:23):-1/dsmp_dsmapi_stats_cb: .
*Jun  8 20:10:38.265: //28/6F1FA7868003/DSM:(2/0:23):-1/dsp_stream_mgr_req_stats: .
*Jun  8 20:10:38.269: //28/6F1FA7868003/DSM:(2/0:23):-1/dsmp_dsmapi_stats_cb: .
*Jun  8 20:10:43.481: //28/6F1FA7868003/DSM:(2/0:23):-1/dsp_stream_mgr_req_stats: .
*Jun  8 20:10:43.489: //28/6F1FA7868003/DSM:(2/0:23):-1/dsmp_dsmapi_stats_cb: .
```

The following is sample output from the **debug voip dsm** command, with Cisco IOS Release 12.3(14)T software, when a VoIP call is in transition from connected to the disconnected state:

```
Router# debug voip dsm

*Jun  8 20:12:14.701: //28/6F1FA7868003/DSM:(2/0:23):-1/dsp_stream_mgr_req_stats: .
*Jun  8 20:12:14.705: //28/6F1FA7868003/DSM:(2/0:23):-1/dsmp_dsmapi_stats_cb: .
*Jun  8 20:12:18.721: //28/6F1FA7868003/DSM:(2/0:23):-1/dsp_stream_mgr_bridge_drop:
*Jun  8 20:12:18.721:  dsp_stream_mgr_bridge_drop, src_call_id 28, dst_call_id 27
*Jun  8 20:12:18.721: //28/6F1FA7868003/DSM:(2/0:23):-1/dsm_delete_conn_array_entry:
```

```
dmgr->connArr.count = 0
*Jun  8 20:12:18.737: //28/6F1FA7868003/DSM:(2/0:23):-1/dsmp_dsmapi_stats_cb: .
*Jun  8 20:12:18.765: //28/6F1FA7868003/DSM:(2/0:23):-1/dsp_stream_mgr_destroy: .
*Jun  8 20:12:18.765:
//28/6F1FA7868003/DSM:(2/0:23):-1/dsmapi_accept_modem_passthrough_session:  : dmgr: 6561C520,
 active sessions 0, max sessions: 16 rejected sessions till now: 0
*Jun  8 20:12:18.769: //28/6F1FA7868003/DSM:(2/0:23):-1/dsmp_dsmapi_reserve_resource_cb: .
```

Related Commands

Command	Description
debug voip dsmp	Displays debugging information from the DSMP and its related applications.
debug voip vtsp	Displays information about the VTSP.

debug voip dsmp

To display debugging information from the Distributed Stream Media Processor (DSMP) and its related applications, use the **debug voip dsmp** command in privileged EXEC mode. To disable debugging output, use the **no**form of this command.

debug voip dsmp [**all**| **default**| **error**| **event**| **function**| **individual**| **inout**| **rtp**| **session**| **stats**| **tone**| **vofr**]

no debug voip dsmp

Syntax Description

all	(Optional) Enables all DSMP debugging (except **stats**).
default	(Optional) Activates inout, error, and event debugging.
error	(Optional) Enables DSMP error debugging.
event	(Optional) Enables state machine debugging.
function	(Optional) Enables procedure tracing.
individual	(Optional) Enables individual DSMP debugging.
inout	(Optional) Enables subsystem inout debugging.
rtp	(Optional) Enables Real-Time Protocol (RTP) debugging on DSMP.
session	(Optional) Enables session debugging.
stats	(Optional) Enables DSMP statistics debugging.
tone	(Optional) Enables tone debugging.
vofr	(Optional) Enables Voice over Frame Relay (VoFR) debugging on the VPM.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.3(8)T	This command was introduced.

Release	Modification
12.3(14)T	The **all**, **default**, **error**, **event**, **function**, **individual**, **inout**, **rtp**, **session**, **stats**, **tone**, and **vofr**keywords were added to the command.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.

Usage Guidelines

To debug VoIP calls, use this command in conjunction with **debug voip vtsp** command and **debug voip dsm** commands. All the related information for media processing is now available by using DSMP . DSM is responsible for creating streams and issuing connections between them.

Examples

The following is sample output from the **debug voip dsmp** command for transcoding call:

```
Router# debug voip dsmp

Syslog logging:enabled (11 messages dropped, 2 messages rate-limited,
                0 flushes, 0 overruns, xml disabled, filtering disabled)
    Console logging:disabled
    Monitor logging:level debugging, 0 messages logged, xml disabled,
                     filtering disabled
    Buffer logging:level debugging, 236 messages logged, xml disabled,
                    filtering disabled
    Logging Exception size (4096 bytes)
    Count and timestamp logging messages:disabled
    Trap logging:level informational, 25 message lines logged

Log Buffer (3000000 bytes):

00:10:18:dsmpAddStream::
00:10:18:dsmpAddStream::
00:10:18:dsmpAddStream::
00:10:18:dsmpAddStream::
00:10:18:dsmpConnectXcodeBinderReq
00:10:18:dsmp_process_event:
00:10:18:dsmp_process_event:evt->requestType = E_DSMP_CC_XCODE_REQ
00:10:18:dsmpxc_act_alloc_rsc::state = S_DSMPXC_INITIAL event = E_DSMP_CC_XCODE_REQ
00:10:18:dsp_ret = 4 (if failed, cause = 0)
00:10:18:dsp_intf = 0x63DD21D8
00:10:18:dsmp_set_state_var:message to DSP
    successful,ret::4
00:10:18:CNFSM:cur_container:xcoder_container, cur_state:S_DSMPXC_INITIAL,
event:E_DSMP_CC_XCODE_REQ, next_state:S_DSMPXC_RSC_ALLOCING
00:10:18:dsmp_is_ret_succ::
00:10:18:dsmpxc_act_init_rsc::state = S_DSMPXC_RSC_ALLOCING event = CNFSM_LAMBDA_EVENT
00:10:18:dsmp_set_state_var:message to DSP
    successful,ret::4
00:10:18:CNFSM:cur_container:xcoder_container, cur_state:S_DSMPXC_RSC_ALLOCING,
event:CNFSM_LAMBDA_EVENT, next_state:S_DSMPXC_RSC_INITING
00:10:18:dsmp_is_ret_succ::
00:10:18:dsmpxc_act_open_rsc::state = S_DSMPXC_RSC_INITING event = CNFSM_LAMBDA_EVENT
00:10:18:dsmp_set_state_var:message to DSP
    successful,ret::4
00:10:18:CNFSM:cur_container:xcoder_container, cur_state:S_DSMPXC_RSC_INITING,
event:CNFSM_LAMBDA_EVENT, next_state:S_DSMPXC_RSC_OPENING
00:10:18:dsmp_is_ret_succ::
00:10:18:dsmpxc_act_program_rsc::state = S_DSMPXC_RSC_OPENING event = CNFSM_LAMBDA_EVENT
00:10:18:dsmp_set_state_var:message to DSP
    successful,ret::4
00:10:18:CNFSM:cur_container:xcoder_container, cur_state:S_DSMPXC_RSC_OPENING,
event:CNFSM_LAMBDA_EVENT, next_state:S_DSMPXC_XCODE_PEND
```

```
00:10:18:CNFSM:new_container:xcoding_container
00:10:18:dsmp_is_ret_succ::
00:10:18:dsmpxc_act_succ_conn_req::state = S_DSMPXC_XCODE_PEND event = CNFSM_LAMBDA_EVENT
00:10:18:CNFSM:cur_container:xcoding_container, cur_state:S_DSMPXC_XCODE_PEND,
event:CNFSM_LAMBDA_EVENT, next_state:S_DSMPXC_XCODE
00:10:18:no_stream_in_session
Router# clear log
Clear logging buffer [confirm]
Router#
Router# clear logsh logg

Syslog logging:enabled (11 messages dropped, 2 messages rate-limited,
                0 flushes, 0 overruns, xml disabled, filtering disabled)
    Console logging:disabled
    Monitor logging:level debugging, 0 messages logged, xml disabled,
                     filtering disabled
    Buffer logging:level debugging, 274 messages logged, xml disabled,
                    filtering disabled
    Logging Exception size (4096 bytes)
    Count and timestamp logging messages:disabled
    Trap logging:level informational, 25 message lines logged

Log Buffer (3000000 bytes):

00:10:50:dsmpDeleteStream
00:10:50:dsmpDeleteStream
00:10:50:dsmpDeleteStream
00:10:50:dsmpDeleteStream
00:10:50:dsmp_process_event:
00:10:50:dsmp_process_event:evt->requestType = E_DSMP_CC_DELETE_STREAM
00:10:50:dsmpxc_act_delete_stream::state = S_DSMPXC_XCODE event = E_DSMP_CC_DELETE_STREAM
00:10:50:CNFSM:cur_container:xcoding_container, cur_state:S_DSMPXC_XCODE,
event:E_DSMP_CC_DELETE_STREAM, next_state:CNFSM_NO_STATE_CHANGE
00:10:50:no_stream_in_session
00:10:50:dsmp_process_event:evt->requestType = E_DSMP_CC_DELETE_STREAM
00:10:50:dsmpxc_act_delete_stream::state = S_DSMPXC_XCODE event = E_DSMP_CC_DELETE_STREAM
00:10:50:CNFSM:cur_container:xcoding_container, cur_state:S_DSMPXC_XCODE,
event:E_DSMP_CC_DELETE_STREAM, next_state:CNFSM_NO_STATE_CHANGE
00:10:50:no_stream_in_session
00:10:50:dsmp_process_event:evt->requestType = E_DSMP_CC_DELETE_STREAM
00:10:50:dsmpxc_act_delete_stream::state = S_DSMPXC_XCODE event = E_DSMP_CC_DELETE_STREAM
00:10:50:CNFSM:cur_container:xcoding_container, cur_state:S_DSMPXC_XCODE,
event:E_DSMP_CC_DELETE_STREAM, next_state:CNFSM_NO_STATE_CHANGE
00:10:50:no_stream_in_session
00:10:50:dsmp_process_event:evt->requestType = E_DSMP_CC_DELETE_STREAM
00:10:50: dsmpxc_act_delete_stream::state = S_DSMPXC_XCODE event = E_DSMP_CC_DELETE_STREAM

00:10:50:CNFSM:cur_container:xcoding_container, cur_state:S_DSMPXC_XCODE,
event:E_DSMP_CC_DELETE_STREAM, next_state:CNFSM_NO_STATE_CHANGE
00:10:50:no_stream_in_session
00:10:50:dsmpxc_act_stop_rsc::state = S_DSMPXC_XCODE event = CNFSM_LAMBDA_EVENT
00:10:50:dsmp_set_state_var:message to DSP
    successful,ret::4
00:10:50:CNFSM:cur_container:xcoding_container, cur_state:S_DSMPXC_XCODE,
event:CNFSM_LAMBDA_EVENT, next_state:S_DSMPXC_CLOSING
00:10:50:dsmp_is_ret_succ::
00:10:50:dsmpxc_act_dealloc_rsc::state = S_DSMPXC_CLOSING event = CNFSM_LAMBDA_EVENT
00:10:50:dsmp_set_state_var:message to DSP
    successful,ret::4
00:10:50:CNFSM:cur_container:xcoding_container, cur_state:S_DSMPXC_CLOSING,
event:CNFSM_LAMBDA_EVENT, next_state:S_DSMPXC_STILL_CLOSING
00:10:50:CNFSM:new_container:xcoder_container
00:10:50:dsmp_is_ret_succ::
00:10:50:dsmpxc_act_start_timer::state = S_DSMPXC_STILL_CLOSING event = CNFSM_LAMBDA_EVENT

00:10:50:CNFSM:cur_container:xcoder_container, cur_state:S_DSMPXC_STILL_CLOSING,
event:CNFSM_LAMBDA_EVENT, next_state:CNFSM_NO_STATE_CHANGE
00:10:50:dsmp_process_event:evt->requestType = E_DSMP_DSPRM_CLOSE_COMPLETE
00:10:50:dsmpxc_act_terminate::state = S_DSMPXC_STILL_CLOSING event =
E_DSMP_DSPRM_CLOSE_COMPLETE
00:10:50:dsmpxc_act_terminate Removing the program based stream = 1
00:10:50:CNFSM:cur_container:xcoder_container, cur_state:S_DSMPXC_STILL_CLOSING,
event:E_DSMP_DSPRM_CLOSE_COMPLETE, next_state:CNFSM_NULL_STATE
```

```
00:10:50:dsmp_free_session
00:10:50:dsmp_process_event:
```

The following is sample output from the **debug voip dsmp** command, with Cisco IOS Release 12.3(14)T software, when a VoIP call is in transition to the connected state:

```
Router# debug voip dsmp

*May 22 04:12:17.775: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmpAddStream: callID = 24, confID = -1,
 streamType = TDM, streamDir = INPUT, ownerContext = 0x0000000C, codec = 0x0, fax_modem_type
 =0, XmitFn = 0x00000000, multicastStreamDtmfType = 0
*May 22 04:12:17.775: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmpAddStream: streamID = 2
*May 22 04:12:17.779: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmpAddStream: callID = 24, confID = -1,
 streamType = TDM, streamDir = OUTPUT, ownerContext = 0x0000000C, codec = 0x0, fax_modem_type
 =0, XmitFn = 0x00000000, multicastStreamDtmfType = 0
*May 22 04:12:17.779: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmpAddStream: streamID = 3
*May 22 04:12:17.779: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmpReserveGwResource: sIdLegInput1 =
2, sIdLegOutput1 = 3, requesterCallID = 24, respFunc = 0x61CD1EE0
*May 22 04:12:17.787: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmp_new_or_existing_gw_session: created
 a new session = 0x657EEE20
*May 22 04:12:17.787: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmp_enlist_stream: Stream 2 is enlisted,
 total = 1
*May 22 04:12:17.787: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmp_enlist_stream: Stream 3 is enlisted,
 total = 2
*May 22 04:12:17.787: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmp_exec:
   [state:S_DSMP_INITIAL, event:E_DSMP_CC_RESERVE_RESOURCE_REQ]
*May 22 04:12:17.791: CNFSM: cur_container:rsc_main_container, cur_state:S_DSMP_INITIAL,
event:E_DSMP_CC_RESERVE_RESOURCE_REQ
*May 22 04:12:17.791: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmp_gw_act_alloc_rsc:
*May 22 04:12:17.791: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmp_decide_pgm_based_stm:
*May 22 04:12:17.791: //-1/F401BFC88006/DSMP:():-1/dsmp_set_state_var: Message to DSP is
successful
*May 22 04:12:17.791: CNFSM: new_container:rsc_allocating_container
*May 22 04:12:17.791: CNFSM: next_state:S_DSMP_RSC_ALLOCATING
*May 22 04:12:17.791: //-1/F401BFC88006/DSMP:():-1/dsmp_gw_act_rsc_alloc_succ:
*May 22 04:12:17.791: CNFSM: next_state:S_DSMP_RSC_ALLOCATED
*May 22 04:12:18.047: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmpReserveGwResource: sIdLegInput1 =
2, sIdLegOutput1 = 3, requesterCallID = 24, respFunc = 0x61CD1EE0
*May 22 04:12:18.047: //-1/F401BFC88006/DSMP:():-1/dsmp_exec:
   [state:S_DSMP_RSC_ALLOCATED, event:E_DSMP_CC_RESERVE_RESOURCE_REQ]
*May 22 04:12:18.051: CNFSM: cur_container:rsc_allocating_container,
cur_state:S_DSMP_RSC_ALLOCATED, event:E_DSMP_CC_RESERVE_RESOURCE_REQ
*May 22 04:12:18.051: CNFSM: new_container:rsc_main_container
*May 22 04:12:18.051: //-1/F401BFC88006/DSMP:():-1/dsmp_gw_act_rsc_reopen:
*May 22 04:12:18.051: //-1/F401BFC88006/DSMP:():-1/dsmp_set_state_var: Message to DSP is
successful
*May 22 04:12:18.051: CNFSM: history stored state: S_DSMP_RSC_ALLOCATED, container:
rsc_allocating_container
*May 22 04:12:18.051: CNFSM: updated current container: rsc_main_container
*May 22 04:12:18.051: CNFSM: next_state:S_DSMP_RSC_REOPENING
*May 22 04:12:18.051: //-1/F401BFC88006/DSMP:():-1/dsmp_gw_act_rsc_alloc_succ:
*May 22 04:12:18.051: CNFSM: restoring history state: S_DSMP_RSC_ALLOCATED
*May 22 04:12:18.055: CNFSM: restoring history container: rsc_allocating_container
*May 22 04:12:18.055: CNFSM: new_container:rsc_allocating_container
*May 22 04:12:18.055: CNFSM: next_state:S_DSMP_RSC_ALLOCATED
*May 22 04:12:18.055: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmpStopPlay: sIdLegOutput = 3
*May 22 04:12:18.059: //-1/F401BFC88006/DSMP:():-1/dsmp_exec:
   [state:S_DSMP_RSC_ALLOCATED, event:E_DSMP_CC_STOP_PLAY_REQ]
*May 22 04:12:18.059: CNFSM: cur_container:rsc_allocating_container,
cur_state:S_DSMP_RSC_ALLOCATED, event:E_DSMP_CC_STOP_PLAY_REQ
*May 22 04:12:18.059: CNFSM: new_container:rsc_main_container
*May 22 04:12:18.059: //-1/F401BFC88006/DSMP:():-1/dsmp_gw_act_ignore:
*May 22 04:12:18.059: CNFSM: next_state:CNFSM_NO_STATE_CHANGE
*May 22 04:12:18.063: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmpAddStream: callID = 23, confID = 12,
 streamType = PACKET, streamDir = INPUT, ownerContext = 0x0000000C, codec = 0x1,
fax_modem_type =0, XmitFn = 0x6158E1F8, multicastStreamDtmfType = -1
*May 22 04:12:18.067: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmpAddStream: streamID = 1
*May 22 04:12:18.067: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmpAddStream: callID = 23, confID = 12,
 streamType = PACKET, streamDir = OUTPUT, ownerContext = 0x0000000C, codec = 0x1,
fax_modem_type =0, XmitFn = 0x6158E1F8, multicastStreamDtmfType = -1
*May 22 04:12:18.067: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmpAddStream: streamID = 4
*May 22 04:12:18.071: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmpUpdateRtpMode: sId = 1, callID = 24,
 mode = 1
*May 22 04:12:18.071: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmpModifyReq: sIdLegInput = 1,
sIdLegOutput = 4, callID = 24
*May 22 04:12:18.075: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmpConnectGwBinderReq:
   sIdLegInputTdm1=2, sIdLegOutputPak1=4, sIdLegInputPak2=1, sIdLegOutputTdm2=3, Call Id=24
*May 22 04:12:18.075: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmpModifyReq: sIdLegInput = 2,
sIdLegOutput = 3, callID = 24FORKING Parameters are forking mask: 7,
simple_forking_codec_mask: 327679, complex_forking_codec_mask 327679
*May 22 04:12:18.075: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmp_process_event: No session is
associated to the streams.
*May 22 04:12:18.075: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmp_save_modify_caps:
*May 22 04:12:18.079: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmp_save_modify_packet_stream_caps:
*May 22 04:12:18.079: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmp_leg_voice_elog_write:
*May 22 04:12:18.079: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmp_leg_voice_elog_write:
   CALL_ERROR; DSMP Session Is NULL
*May 22 04:12:18.079: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmp_leg_voice_elog_write:exit@2908
*May 22 04:12:18.079: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmp_leg_voice_elog_write:
*May 22 04:12:18.079: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmp_leg_voice_elog_write:
   CALL_ERROR; DSMP Session Is NULL
*May 22 04:12:18.079: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmp_leg_voice_elog_write:exit@2908
*May 22 04:12:18.079: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmp_leg_voice_elog_write:
*May 22 04:12:18.079: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmp_leg_voice_elog_write:
   CALL_ERROR; DSMP Session Is NULL
*May 22 04:12:18.079: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmp_leg_voice_elog_write:exit@2908
*May 22 04:12:18.083: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmp_leg_voice_elog_write:
*May 22 04:12:18.083: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmp_leg_voice_elog_write:
   CALL_ERROR; DSMP Session Is NULL
*May 22 04:12:18.083: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmp_leg_voice_elog_write:exit@2908
*May 22 04:12:18.083: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmp_save_modify_packet_stream_caps:
*May 22 04:12:18.083: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmp_save_modify_tdm_stream_caps:
*May 22 04:12:18.083: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmp_save_modify_tdm_stream_caps:exit@385
*May 22 04:12:18.087: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmp_save_modify_tdm_stream_caps:
*May 22 04:12:18.087: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmp_save_modify_tdm_stream_caps:exit@385
*May 22 04:12:18.087: //-1/F401BFC88006/DSMP:():-1/dsmp_enlist_stream: Stream 1 is enlisted,
 total = 3
*May 22 04:12:18.087: //-1/F401BFC88006/DSMP:():-1/dsmp_enlist_stream: Stream 4 is enlisted,
 total = 4
*May 22 04:12:18.087: //-1/F401BFC88006/DSMP:():-1/dsmp_exec:
   [state:S_DSMP_RSC_ALLOCATED, event:E_DSMP_CC_CONNECT_REQ]
*May 22 04:12:18.091: CNFSM: cur_container:rsc_allocating_container,
cur_state:S_DSMP_RSC_ALLOCATED, event:E_DSMP_CC_CONNECT_REQ
*May 22 04:12:18.091: //-1/F401BFC88006/DSMP:():-1/dsmp_is_req_not_in_proc_dsp_ready:
*May 22 04:12:18.091: //-1/F401BFC88006/DSMP:():-1/dsmp_gw_act_rsc_init:
*May 22 04:12:18.091: //-1/F401BFC88006/DSMP:():-1/dsmp_set_state_var: Message to DSP is
successful
*May 22 04:12:18.091: CNFSM: history stored state: S_DSMP_RSC_ALLOCATED, container:
rsc_allocating_container
*May 22 04:12:18.091: CNFSM: new_container:rsc_main_container
*May 22 04:12:18.095: CNFSM: next_state:S_DSMP_RSC_INITING
*May 22 04:12:18.095: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmp_decide_pgm_based_stm:
*May 22 04:12:18.095: //-1/F401BFC88006/DSMP:():-1/dsmp_gw_act_rsc_init_succ: pgm_base_stm_id
 = 1, service_id = 26, codec =4
*May 22 04:12:18.095: //-1/F401BFC88006/DSMP:():-1/dsmp_set_state_var: Message to DSP is
successful
*May 22 04:12:18.099: CNFSM: next_state:S_DSMP_RSC_OPENING
*May 22 04:12:18.099: //-1/F401BFC88006/DSMP:():-1/dsmp_gw_act_rsc_open_succ:
*May 22 04:12:18.099: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmp_leg_voice_elog_write:
*May 22 04:12:18.099: //-1/F401BFC88006/DSMP:():-1/dsmp_conf_static_params:
*May 22 04:12:18.099: //-1/F401BFC88006/DSMP:():-1/dsmp_conf_static_params: servic_type=1
base stream=1 tdm_caps=0x6557E454 pkt_caps=0x6557EC3C
*May 22 04:12:18.099: //-1/F401BFC88006/DSMP:():-1/dsmp_conf_static_params:
*May 22 04:12:18.103:  ip_tones = 0, pstn_tones = 0
*May 22 04:12:18.103: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmp_conf_static_params: voice detection
 disabled
*May 22 04:12:18.103: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmp_conf_static_params: silence detection
 disabled
*May 22 04:12:18.103: //-1/F401BFC88006/DSMP:():-1/dsmp_set_state_var: Message to DSP is
successful
*May 22 04:12:18.107: CNFSM: next_state:S_DSMP_RSC_STATIC_CONF
*May 22 04:12:18.107: //-1/F401BFC88006/DSMP:():-1/dsmp_gw_act_start_service:
*May 22 04:12:18.107: //-1/F401BFC88006/DSMP:():-1/dsmp_set_state_var: Message to DSP is
successful
*May 22 04:12:18.107: CNFSM: next_state:S_DSMP_RSC_STARTING
```

```
*May 22 04:12:18.107: //-1/F401BFC88006/DSMP:():-1/dsmp_act_rsc_start_succ:
*May 22 04:12:18.107: //-1/F401BFC88006/DSMP:():-1/dsmp_conf_dynamic_params:
*May 22 04:12:18.111: //-1/F401BFC88006/DSMP:():-1/dsmp_conf_dynamic_params: servic_type=1
 base stream=1
*May 22 04:12:18.111: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmp_leg_voice_elog_write:
*May 22 04:12:18.111: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmp_leg_voice_elog_write:
*May 22 04:12:18.111: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmp_leg_voice_elog_write:
*May 22 04:12:18.111: //-1/F401BFC88006/DSMP:():-1/dsmp_conf_detector_params:
*May 22 04:12:18.115: //-1/F401BFC88006/DSMP:():-1/dsmp_set_state_var: Message to DSP is
successful
*May 22 04:12:18.115: CNFSM: next_state:S_DSMP_RSC_DYNAMIC_CONF
*May 22 04:12:18.115: CNFSM: new_container:rsc_running_container
*May 22 04:12:18.115: CNFSM: next_state:S_DSMP_MCAST_CHECK
*May 22 04:12:18.115: //-1/F401BFC88006/DSMP:():-1/dsmp_is_mcast:
*May 22 04:12:18.119: //-1/F401BFC88006/DSMP:():-1/dsmp_is_mcast:exit@1487
*May 22 04:12:18.119: //-1/F401BFC88006/DSMP:():-1/dsmp_is_not_mcast:
*May 22 04:12:18.119: //-1/F401BFC88006/DSMP:():-1/dsmp_is_mcast:
*May 22 04:12:18.119: //-1/F401BFC88006/DSMP:():-1/dsmp_is_mcast:exit@1487
*May 22 04:12:18.119: //-1/F401BFC88006/DSMP:():-1/dsmp_is_not_mcast:exit@1500
*May 22 04:12:18.119: CNFSM: next_state:S_DSMP_DTMF_FSK_MODE_CHECK
*May 22 04:12:18.119: //-1/F401BFC88006/DSMP:():-1/dsmp_is_fsk_or_dtmf:
*May 22 04:12:18.123: //-1/F401BFC88006/DSMP:():-1/dsmp_is_diagnostic:
*May 22 04:12:18.123: //-1/F401BFC88006/DSMP:():-1/dsmp_is_diagnostic:exit@1587
*May 22 04:12:18.123: //-1/F401BFC88006/DSMP:():-1/dsmp_is_not_dtmf_fsk:
*May 22 04:12:18.123: //-1/F401BFC88006/DSMP:():-1/dsmp_is_fsk_or_dtmf:
*May 22 04:12:18.123: //-1/F401BFC88006/DSMP:():-1/dsmp_is_diagnostic:
*May 22 04:12:18.123: //-1/F401BFC88006/DSMP:():-1/dsmp_is_diagnostic:exit@1587
*May 22 04:12:18.123: //-1/F401BFC88006/DSMP:():-1/dsmp_is_not_dtmf_fsk:exit@1600
*May 22 04:12:18.127: CNFSM: next_state:S_DSMP_NOT_DTMF_FSK
*May 22 04:12:18.127: //-1/F401BFC88006/DSMP:():-1/dsmp_is_simple_voice:
*May 22 04:12:18.127: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmp_get_stm_service_type:
*May 22 04:12:18.127: //-1/F401BFC88006/DSMP:():-1/dsmp_gw_act_simple_voice_start:
*May 22 04:12:18.127: //-1/F401BFC88006/DSMP:():-1/dsmp_connect_req_done:
*May 22 04:12:18.127: //-1/F401BFC88006/DSMP:():-1/dsmp_enlist_connection: Connection is
added, enlisted LegIn: 2, enlisted LegOut: 4
*May 22 04:12:18.131: //-1/F401BFC88006/DSMP:():-1/dsmp_enlist_connection: Connection is
added, enlisted LegIn: 1, enlisted LegOut: 3
*May 22 04:12:18.131: CNFSM: new_container:simple_voice_container
*May 22 04:12:18.131: CNFSM: next_state:S_DSMP_VC_RUNNING
*May 22 04:12:18.131: //-1/F401BFC88006/DSMP:():-1/dsmp_exec:
   [state:S_DSMP_VC_RUNNING, event:E_DSMP_CC_MODIFY_REQ]
*May 22 04:12:18.135: CNFSM: cur_container:simple_voice_container,
cur_state:S_DSMP_VC_RUNNING, event:E_DSMP_CC_MODIFY_REQ
*May 22 04:12:18.135: CNFSM: new_container:rsc_running_container
*May 22 04:12:18.135: //-1/F401BFC88006/DSMP:():-1/dsmp_gw_act_save_modify_caps:
*May 22 04:12:18.135: //-1/F401BFC88006/DSMP:():-1/dsmp_save_modify_caps:
*May 22 04:12:18.135: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmp_save_modify_packet_stream_caps:
*May 22 04:12:18.139: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmp_save_modify_packet_stream_caps:
*May 22 04:12:18.139: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmp_save_modify_tdm_stream_caps:
*May 22 04:12:18.139: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmp_save_modify_tdm_stream_caps:
*May 22 04:12:18.139: //-1/F401BFC88006/DSMP:():-1/dsmp_save_stream_detectors:
*May 22 04:12:18.139: CNFSM: history stored state: S_DSMP_VC_RUNNING, container:
simple_voice_container
*May 22 04:12:18.139: CNFSM: updated current container: rsc_running_container
*May 22 04:12:18.143: CNFSM: next_state:S_DSMP_MODIFY_CAPS_SAVED
*May 22 04:12:18.143: //-1/F401BFC88006/DSMP:():-1/dsmp_is_modify_to_modem_passthru:
*May 22 04:12:18.143: //-1/F401BFC88006/DSMP:():-1/dsmp_is_static_params_changed:
*May 22 04:12:18.143: //-1/F401BFC88006/DSMP:():-1/dsmp_is_modify_to_modem_passthru:
*May 22 04:12:18.143: //-1/F401BFC88006/DSMP:():-1/dsmp_is_dynamic_params_changed:
*May 22 04:12:18.143: //-1/F401BFC88006/DSMP:():-1/dsmp_is_modify_to_modem_passthru:
*May 22 04:12:18.143: //-1/F401BFC88006/DSMP:():-1/dsmp_gw_act_dynamic_params_changed:
*May 22 04:12:18.147: //-1/F401BFC88006/DSMP:():-1/dsmp_conf_dynamic_params:
*May 22 04:12:18.147: //-1/F401BFC88006/DSMP:():-1/dsmp_conf_dynamic_params: servic_type=1
 base stream=1
*May 22 04:12:18.147: //-1/F401BFC88006/DSMP:():-1/dsmp_conf_detector_params:
*May 22 04:12:18.147: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmp_conf_detector_params: fax: 1
*May 22 04:12:18.147: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmp_conf_detector_params: modem: 1
*May 22 04:12:18.151: CNFSM: restoring history state: S_DSMP_VC_RUNNING
*May 22 04:12:18.151: CNFSM: restoring history container: simple_voice_container
*May 22 04:12:18.151: CNFSM: new_container:simple_voice_container
*May 22 04:12:18.151: CNFSM: next_state:S_DSMP_VC_RUNNING
*May 22 04:12:18.163: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmpUpdateRtpMode: sId = 1, callID = 24,
 mode = 3
```

```
*May 22 04:12:18.163: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmpUpdateRtpMode: sId = 1, callID = 24,
 mode = 3
*May 22 04:12:30.947: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmpUpdateRtpMode: sId = 1, callID = 24,
 mode = 3
```

The following is sample output from the **debug voip dsmp** command, with Cisco IOS Release 12.3(14)T software, when a VoIP call is in transition from connected to the disconnected state:

```
Router# debug voip dsmp


*May 22 04:12:30.951: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmpDisconnectGwBinder: sIdLegInputTdm1
 = 2, sIdLegOutputPak1 = 4, sIdLegInputPak2 = 1, sIdLegOutputTdm2 = 3 requesterCallID = 24
*May 22 04:12:30.951: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmpDeleteStream: streamID = 1, callID
= 23, ownerContext = 0x00000000
*May 22 04:12:30.951: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmpDeleteStream: streamID = 4, callID
= 23, ownerContext = 0x00000000
*May 22 04:12:30.951: //-1/F401BFC88006/DSMP:():-1/dsmp_exec:
   [state:S_DSMP_VC_RUNNING, event:E_DSMP_CC_DISCONNECT]
*May 22 04:12:30.955: CNFSM: cur_container:simple_voice_container,
cur_state:S_DSMP_VC_RUNNING, event:E_DSMP_CC_DISCONNECT
*May 22 04:12:30.955: CNFSM: new_container:rsc_running_container
*May 22 04:12:30.955: //-1/F401BFC88006/DSMP:():-1/dsmp_gw_act_cc_disconnect:
*May 22 04:12:30.955: //-1/F401BFC88006/DSMP:():-1/dsmp_delist_connection: Connection is
delisted, delisted LegIn: 2, delisted LegOut: 4, total conn_count is = 1
*May 22 04:12:30.955: //-1/F401BFC88006/DSMP:():-1/dsmp_delist_connection: Connection is
delisted, delisted LegIn: 1, delisted LegOut: 3, total conn_count is = 0
*May 22 04:12:30.955: CNFSM: next_state:CNFSM_NO_STATE_CHANGE
*May 22 04:12:30.955: //-1/F401BFC88006/DSMP:():-1/dsmp_exec:
   [state:S_DSMP_VC_RUNNING, event:E_DSMP_DELETE_STREAM]
*May 22 04:12:30.959: CNFSM: cur_container:simple_voice_container,
cur_state:S_DSMP_VC_RUNNING, event:E_DSMP_DELETE_STREAM
*May 22 04:12:30.959: CNFSM: new_container:rsc_running_container
*May 22 04:12:30.959: CNFSM: new_container:rsc_main_container
*May 22 04:12:30.959: //-1/F401BFC88006/DSMP:():-1/dsmp_not_last_stream:
*May 22 04:12:30.959: //-1/F401BFC88006/DSMP:():-1/dsmp_is_last_stream:
*May 22 04:12:30.959: //-1/F401BFC88006/DSMP:():-1/dsmp_gw_act_not_last_stream:
*May 22 04:12:30.959: //-1/F401BFC88006/DSMP:():-1/remove_stream_from_DB:
*May 22 04:12:30.959: //-1/F401BFC88006/DSMP:():-1/process_software_multicast_streams:
*May 22 04:12:30.959: CNFSM: next_state:CNFSM_NO_STATE_CHANGE
*May 22 04:12:30.959: //-1/F401BFC88006/DSMP:():-1/dsmp_exec:
   [state:S_DSMP_VC_RUNNING, event:E_DSMP_DELETE_STREAM]
*May 22 04:12:30.959: CNFSM: cur_container:simple_voice_container,
cur_state:S_DSMP_VC_RUNNING, event:E_DSMP_DELETE_STREAM
*May 22 04:12:30.963: CNFSM: new_container:rsc_running_container
*May 22 04:12:30.963: CNFSM: new_container:rsc_main_container
*May 22 04:12:30.963: //-1/F401BFC88006/DSMP:():-1/dsmp_not_last_stream:
*May 22 04:12:30.963: //-1/F401BFC88006/DSMP:():-1/dsmp_is_last_stream:
*May 22 04:12:30.963: //-1/F401BFC88006/DSMP:():-1/dsmp_gw_act_not_last_stream:
*May 22 04:12:30.963: //-1/F401BFC88006/DSMP:():-1/remove_stream_from_DB:
*May 22 04:12:30.963: //-1/F401BFC88006/DSMP:():-1/process_software_multicast_streams:
*May 22 04:12:30.963: CNFSM: next_state:CNFSM_NO_STATE_CHANGE
*May 22 04:12:30.967: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmpStopPlay: sIdLegOutput = 3
*May 22 04:12:30.975: //-1/F401BFC88006/DSMP:():-1/dsmp_exec:
   [state:S_DSMP_VC_RUNNING, event:E_DSMP_CC_STOP_PLAY_REQ]
*May 22 04:12:30.975: CNFSM: cur_container:simple_voice_container,
cur_state:S_DSMP_VC_RUNNING, event:E_DSMP_CC_STOP_PLAY_REQ
*May 22 04:12:30.975: CNFSM: new_container:rsc_running_container
*May 22 04:12:30.975: CNFSM: new_container:rsc_main_container
*May 22 04:12:30.975: //-1/F401BFC88006/DSMP:():-1/dsmp_gw_act_ignore:
*May 22 04:12:30.975: CNFSM: next_state:CNFSM_NO_STATE_CHANGE
*May 22 04:12:31.011: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmpStopPlay: sIdLegOutput = 3
*May 22 04:12:31.011: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmpDeleteStream: streamID = 2, callID
= 24, ownerContext = 0x0000000C
*May 22 04:12:31.011: //-1/xxxxxxxxxxxx/DSMP:():-1/dsmpDeleteStream: streamID = 3, callID
= 24, ownerContext = 0x0000000C
*May 22 04:12:31.015: //-1/F401BFC88006/DSMP:():-1/dsmp_exec:
   [state:S_DSMP_VC_RUNNING, event:E_DSMP_CC_STOP_PLAY_REQ]
*May 22 04:12:31.015: CNFSM: cur_container:simple_voice_container,
cur_state:S_DSMP_VC_RUNNING, event:E_DSMP_CC_STOP_PLAY_REQ
*May 22 04:12:31.015: CNFSM: new_container:rsc_running_container
*May 22 04:12:31.015: CNFSM: new_container:rsc_main_container
```

```
*May 22 04:12:31.015: //-1/F401BFC88006/DSMP:():-1/dsmp_gw_act_ignore:
*May 22 04:12:31.015: CNFSM: next_state:CNFSM_NO_STATE_CHANGE
*May 22 04:12:31.015: //-1/F401BFC88006/DSMP:():-1/dsmp_exec:
   [state:S_DSMP_VC_RUNNING, event:E_DSMP_DELETE_STREAM]
*May 22 04:12:31.015: CNFSM: cur_container:simple_voice_container,
cur_state:S_DSMP_VC_RUNNING, event:E_DSMP_DELETE_STREAM
*May 22 04:12:31.019: CNFSM: new_container:rsc_running_container
*May 22 04:12:31.019: CNFSM: new_container:rsc_main_container
*May 22 04:12:31.019: //-1/F401BFC88006/DSMP:():-1/dsmp_not_last_stream:
*May 22 04:12:31.019: //-1/F401BFC88006/DSMP:():-1/dsmp_is_last_stream:
*May 22 04:12:31.019: //-1/F401BFC88006/DSMP:():-1/dsmp_gw_act_not_last_stream:
*May 22 04:12:31.019: //-1/F401BFC88006/DSMP:():-1/remove_stream_from_DB:
*May 22 04:12:31.019: //-1/F401BFC88006/DSMP:():-1/process_software_multicast_streams:
*May 22 04:12:31.019: CNFSM: next_state:CNFSM_NO_STATE_CHANGE
*May 22 04:12:31.019: //-1/F401BFC88006/DSMP:():-1/dsmp_exec:
   [state:S_DSMP_VC_RUNNING, event:E_DSMP_DELETE_STREAM]
*May 22 04:12:31.019: CNFSM: cur_container:simple_voice_container,
cur_state:S_DSMP_VC_RUNNING, event:E_DSMP_DELETE_STREAM
*May 22 04:12:31.019: CNFSM: new_container:rsc_running_container
*May 22 04:12:31.023: CNFSM: new_container:rsc_main_container
*May 22 04:12:31.023: //-1/F401BFC88006/DSMP:():-1/dsmp_not_last_stream:
*May 22 04:12:31.023: //-1/F401BFC88006/DSMP:():-1/dsmp_is_last_stream:
*May 22 04:12:31.023: //-1/F401BFC88006/DSMP:():-1/dsmp_is_last_stream:
*May 22 04:12:31.023: //-1/F401BFC88006/DSMP:():-1/dsmp_gw_act_rsc_stop:
*May 22 04:12:31.023: //-1/F401BFC88006/DSMP:():-1/remove_stream_from_DB:
*May 22 04:12:31.023: //-1/F401BFC88006/DSMP:():-1/process_software_multicast_streams:
*May 22 04:12:31.023: //-1/F401BFC88006/DSMP:():-1/dsmp_set_state_var: Message to DSP is
successful
*May 22 04:12:31.023: CNFSM: history stored state: S_DSMP_VC_RUNNING, container:
simple_voice_container
*May 22 04:12:31.023: CNFSM: updated current container: rsc_main_container
*May 22 04:12:31.027: CNFSM: new_container:rsc_closing_container
*May 22 04:12:31.027: CNFSM: next_state:S_DSMP_RSC_STOPPING
*May 22 04:12:31.027: //-1/F401BFC88006/DSMP:():-1/dsmp_gw_act_rsc_stopping_close:
*May 22 04:12:31.027: //-1/F401BFC88006/DSMP:():-1/dsmp_set_state_var: Message to DSP is
successful
*May 22 04:12:31.027: CNFSM: next_state:S_DSMP_DSPMGR_CLOSING
*May 22 04:12:31.027: //-1/F401BFC88006/DSMP:():-1/dsmp_exec:
   [state:S_DSMP_DSPMGR_CLOSING, event:E_DSMP_DSPRM_CLOSE_COMPLETE]
*May 22 04:12:31.027: CNFSM: cur_container:rsc_closing_container,
cur_state:S_DSMP_DSPMGR_CLOSING, event:E_DSMP_DSPRM_CLOSE_COMPLETE
*May 22 04:12:31.027: //-1/F401BFC88006/DSMP:():-1/dsmp_gw_act_dspmgr_closing_complete:
*May 22 04:12:31.031: //-1/F401BFC88006/DSMP:():-1/remove_stream_from_DB:
*May 22 04:12:31.031: //-1/F401BFC88006/DSMP:():-1/process_software_multicast_streams:
*May 22 04:12:31.031: //-1/F401BFC88006/DSMP:():-1/remove_stream_from_DB:
*May 22 04:12:31.031: //-1/F401BFC88006/DSMP:():-1/process_software_multicast_streams:
*May 22 04:12:31.031: CNFSM: next_state:CNFSM_NULL_STATE
*May 22 04:12:31.031: //-1/F401BFC88006/DSMP:():-1/dsmp_free_session:
```

Related Commands

Command	Description
debug voip dsm	Displays debugging information from the DSM subsystem.
debug voip vtsp	Displays information about the VTSP.

debug voip dspapi

To troubleshoot the digital signal processor (DSP) application programming interface (API), use the **debug voip dspapi** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voip dspapi [**all**| **command**| **default**| **detail**| **error** [**call [informational]**| **software [informational]**]| **function**| **inout**| **notification**| **response**]

no debug voip dspapi

Syntax Description

all	(Optional) Displays all DSP API debugging messages.
command	(Optional) Displays DSP API commands.
default	(Optional) Displays DSP API detail, error, and inout debugging messages. This option also runs if no keywords are added.
detail	(Optional) Displays detailed information about commands sent to the DSP. This command is used in conjunction with other **debug voip dspapi** commands to show additional details when you use the **command**, **notification**, and **response** keywords.
error	(Optional) Displays DSP API errors.
call	(Optional) Displays call processing errors.
informational	(Optional) Displays minor errors and major errors. Without the **informational** keyword, only major errors are displayed.
software	(Optional) Displays software processing errors.
function	(Optional) Displays DSP API functions.
inout	(Optional) Displays output for the **command**, **notification**, and **response** keywords.
notification	(Optional) Displays DSP API notification messages.
response	(Optional) Displays DSP API response messages.

Command Default

Debugging is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(8)T	This command replaces the **debug dspapi**command.
12.3(14)T	T.38 fax relay call statistics were made available to Call Detail Records (CDRs) through Vendor-Specific Attributes (VSAs) and added to the call log.

Usage Guidelines

DSP API message events used to communicate with DSPs are intended for use with Connexant (NextPort) and Texas Instruments (54x) DSPs.

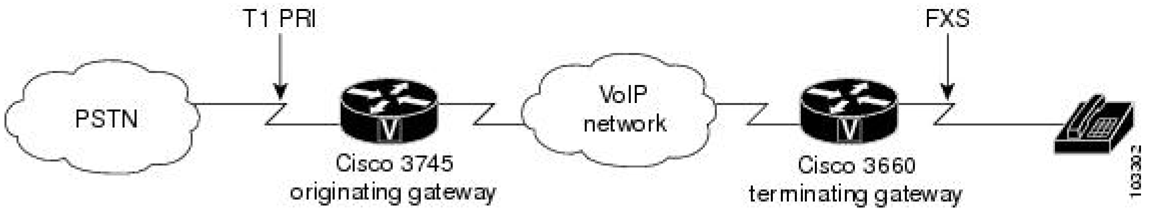
Caution This command severely impacts performance and should be used only for single-call debug capture.

Examples

The following examples show output for variations of the **debug voip dspapi** command:

For these examples, the topology shown in the figure below is used.

Figure 4: Network Topology for debug voip dspapi Examples


T1 PRI
FXS
PSTN
VoIP network
Cisco 3745 originating gateway
Cisco 3660 terminating gateway
103302

Examples

```
Router# debug voip dspapi command
voip dspapi command debugging is ON
Router#
*Apr 18 21:33:48.347: //-1/CD89F6A78020/DSPAPI/[2/1:23]/dsp_init:
*Apr 18 21:33:48.347: //-1/CD89F6A78020/DSPAPI/[2/1:23]/dsp_voice_config_params:
*Apr 18 21:33:48.347:     9 parameters
```

The following lines show encapsulation settings, jitter, inband signaling, echo cancellation, gain, and other quality of service (QoS) settings:

```
   [0] ENCAP RTP: Tx SSRC=0, Rx SSRC=0, Tx VPXCC=0, Rx VPXCC=0
    IFP payload type=122, SID support=1, TSE payload=101, Sequence number start=0
    Redundancy=0, ClearChannel payload type=0, Fax payload type=0
    Alaw PCM switchover=0, MuLAW PCM switchover=0
    DTMF payload type=0, NTE receive payload type=101
    Dynamic payload=0, Codec=g711ulaw
    [1] PO_JITTER: mode=2 initial=60(ms) max=200(ms) min=40(ms) fax_nom=300(ms)
    [2] INBAND_SIG: mode=0x1 enable
    [3] ECHO_CANCEL: Flags=0x37, Echo length=64(ms)
    [4] IDLE_CODE_DET: Enable=0, Code=0x0, Duration=6000(ms)
    [5] GAIN: Input=0.0(dB), Output=0.0(dB)
    [6] CNG: 1
    [7] INFO_FIELD_SIZE: 160 bytes
    [8] DIGIT_RELAY: 2
*Apr 18 21:33:48.359: //66/CD89F6A78020/DSPAPI/[2/1:23:66]/dsp_voice_get_capabilities:
*Apr 18 21:33:48.359: //66/CD89F6A78020/DSPAPI/[2/1:23:66]/dsp_stop_service:
    NONE (0)
*Apr 18 21:33:48.359: //66/CD89F6A78020/DSPAPI/[2/1:23:66]/dsp_open_service:
    G729IETF (25)
*Apr 18 21:33:48.359: //66/CD89F6A78020/DSPAPI/[2/1:23:66]/dsp_voice_config_params:
```

The following lines show settings for encapsulation, packet suppression, and voice activity detection (VAD):

```
*Apr 18 21:33:48.359:     3 parameters
    [0] ENCAP RTP: Tx SSRC=64, Rx SSRC=0, Tx VPXCC=0, Rx VPXCC=0
    IFP payload type=122, SID support=19, TSE payload=101, Sequence number start=3238
    Redundancy=0, ClearChannel payload type=125, Fax payload type=122
    Alaw PCM switchover=8, MuLAW PCM switchover=0

 DTMF payload type=121, NTE receive payload type=101
    Dynamic payload=0, Codec=g729r8
    [1] PAK_SUPPRESS: 0
    [2] VAD: Enable=1, Threshold=-38(dBm)
.
.
.
*Apr 18 21:33:48.363: //66/CD89F6A78020/DSPAPI/[2/1:23:66]/dsp_start_service:
    G729IETF (25)
*Apr 18 21:33:48.363: //66/CD89F6A78020/DSPAPI/[2/1:23:66]/dsp_voice_config_params:
*Apr 18 21:33:48.363:     1 parameter
    [0] VAD: Enable=1, Threshold=-38(dBm)
*Apr 18 21:33:50.867: //66/CD89F6A78020/DSPAPI/[2/1:23:66]/dsp_query_info:
    Request ID=1, Reset Flag=FALSE Q:PO_Delay PO_Error TX RX
.
.
.
*Apr 18 21:34:15.031: //66/CD89F6A78020/DSPAPI/[2/1:23:66]/dsp_query_info:
    Request ID=5, Reset Flag=TRUE Q:Error
*Apr 18 21:34:15.031: //66/CD89F6A78020/DSPAPI/[2/1:23:66]/dsp_query_info:
    Request ID=5, Reset Flag=TRUE Q:Levels
*Apr 18 21:34:15.031: //66/CD89F6A78020/DSPAPI/[2/1:23:66]/dsp_inband_tone_off:
*Apr 18 21:34:15.031: //66/CD89F6A78020/DSPAPI/[2/1:23:66]/dsp_stop_service:
    G729IETF (25)
*Apr 18 21:34:15.031: //66/CD89F6A78020/DSPAPI/[2/1:23:66]/dsp_close_service:
    G729IETF (25)
```

Examples

```
Router# **debug voip dspapi inout**

voip dspapi inout debugging is ON
*May  1 19:59:15.579: //67/2BA0E0758024/DSPAPI/[4/0/0 (67)]/dsp_init:
*May  1 19:59:15.579: //67/2BA0E0758024/DSPAPI/[4/0/0 (67)]/dsp_voice_config_params:
*May  1 19:59:15.579:     9 parameters
```

The following lines show encapsulation settings, jitter, inband signalling, echo cancellation, gain, and other quality of service (QoS) settings:

```
    [0] ENCAP RTP: Tx SSRC=0, Rx SSRC=0, Tx VPXCC=0, Rx VPXCC=0
    IFP payload type=122, SID support=1, TSE payload=101, Sequence number start=0
    Redundancy=0, ClearChannel payload type=0, Fax payload type=0
    Alaw PCM switchover=0, MuLAW PCM switchover=0
    DTMF payload type=0, NTE receive payload type=101
    Dynamic payload=0, Codec=g711ulaw
    [1] PO_JITTER: mode=2 initial=60(ms) max=200(ms) min=40(ms) fax_nom=300(ms)
    [2] INBAND_SIG: mode=0x1 enable
    [3] ECHO_CANCEL: Flags=0x17, Echo length=8(ms)
    [4] IDLE_CODE_DET: Enable=0, Code=0x0, Duration=6000(ms)
    [5] GAIN: Input=0.0(dB), Output=-6550.6(dB)
    [6] CNG: 1
    [7] INFO_FIELD_SIZE: 160 bytes
    [8] DIGIT_RELAY: 2
.
.
*May  1 19:59:15.587: //67/2BA0E0758024/DSPAPI/[4/0/0 (67)]/dsp_voice_get_capabilities:
*May  1 19:59:15.587: //67/2BA0E0758024/DSPAPI/[4/0/0 (67)]/dsp_stop_service:
    NONE (0)
*May  1 19:59:15.587: //67/2BA0E0758024/DSPAPI/[4/0/0 (67)]/dsp_open_service:
    G729IETF (25)
*May  1 19:59:15.587: //67/2BA0E0758024/DSPAPI/[4/0/0 (67)]/dsp_voice_config_params:
*May  1 19:59:15.587:     3 parameters
    [0] ENCAP RTP: Tx SSRC=0, Rx SSRC=0, Tx VPXCC=0, Rx VPXCC=0
    IFP payload type=122, SID support=19, TSE payload=101, Sequence number start=2977
    Redundancy=0, ClearChannel payload type=125, Fax payload type=122
    Alaw PCM switchover=8, MuLAW PCM switchover=0
    DTMF payload type=121, NTE receive payload type=101
    Dynamic payload=0, Codec=g729r8
    [1] PAK_SUPPRESS: 0
    [2] VAD: Enable=1, Threshold=-38(dBm)
*May  1 19:59:15.587: //67/2BA0E0758024/DSPAPI/[4/0/0 (67)]/dsp_idle_service:
    G729IETF (25)
*May  1 19:59:15.587: //67/2BA0E0758024/DSPAPI/[4/0/0 (67)]/dsp_voice_config_params:
*May  1 19:59:15.587:     3 parameters
    [0] INFO_FIELD_SIZE: 20 bytes
    [1] ENCAP RTP: Tx SSRC=64, Rx SSRC=0, Tx VPXCC=0, Rx VPXCC=0
    IFP payload type=122, SID support=19, TSE payload=101, Sequence number start=2977
    Redundancy=0, ClearChannel payload type=125, Fax payload type=122
    Alaw PCM switchover=8, MuLAW PCM switchover=0
    DTMF payload type=121, NTE receive payload type=101
    Dynamic payload=0, Codec=g729r8
    [2] DIGIT_RELAY: 2
*May  1 19:59:15.587: //67/2BA0E0758024/DSPAPI/[4/0/0 (67)]/dsp_start_service:
    G729IETF (25)
*May  1 19:59:15.587: //67/2BA0E0758024/DSPAPI/[4/0/0 (67)]/dsp_voice_config_params:
*May  1 19:59:15.587:     1 parameter
    [0] VAD: Enable=1, Threshold=-38(dBm)
*May  1 19:59:15.591: //67/2BA0E0758024/DSPAPI/[4/0/0 (67)]/dsp_inband_tone_on:
    Tone ID=1, Direction=2, Num frequencies=2
    Frequency(hz): a=440 b=480, Amplitude(dB): a=-16.0 b=-16.0
    Cadence1(ms): ON=2000 OFF=4000
    Cadence2(ms): ON=0 OFF=0
    Cadence3(ms): ON=0 OFF=0
    Cadence4(ms): ON=0 OFF=0
    Frequency(hz): a2=25667 b2=51816 a3=0 b3=1 a4=24596 b4=52484
    ITO option group: 1
*May  1 19:59:17.195: //67/2BA0E0758024/DSPAPI/[4/0/0 (67)]/dsp_inband_tone_off:
01:19:04: %PS-3-MULTFAIL: There is more than one failure with the  Power System 1; please
resolve problems immediately
```

The following statistics repeat for each DSP query. The transmit (tx) and receive (rx) statistics show number of packets, comfort noise settings, duration, and packet status.

```
*May  1 19:59:18.051: //67/2BA0E0758024/DSPAPI/[4/0/0 (67)]/dsp_query_info:
    Request ID=1, Reset Flag=FALSE Q:PO_Delay PO_Error TX RX
*May  1 19:59:18.051: //67/2BA0E0758024/DSPAPI/[4/0/0 (67)]/dsp_tx_stats:
    Request ID=1, Packets: Voice=113, Signaling=0, ComfortNoise=1
    TX duration=2460(ms): Voice=2260(ms), FAX=0(ms)
*May  1 19:59:18.051: //67/2BA0E0758024/DSPAPI/[4/0/0 (67)]/dsp_rx_stats:
    Request ID=1, Packets: Voice=33, Signalling=0, ComfortNoise=1
    RX duration=840(ms): Voice=640(ms), FAX=0(ms)
    Packets: Bad Sequence=0, Bad Protocol=0, Late=0, Early=1
*May  1 19:59:18.051: //67/2BA0E0758024/DSPAPI/[4/0/0 (67)]/dsp_playout_delay_stats:
    Request ID=1, Current=70(ms), MIN=70(ms), MAX=70(ms)
    Clock offset=80(ms), Inter arrival jitter=67082420(ms)
*May  1 19:59:18.051: //67/2BA0E0758024/DSPAPI/[4/0/0 (67)]/dsp_playout_error_stats:
    Request ID=1
    Concealment: Predictive=0(ms), Interpolative=0(ms), Silence=0(ms)
    Retro Memory update=0(ms), Buffer overflow=10(ms)
    Talkspurt endpoint detection errors=0
*May  1 19:59:19.827: //67/2BA0E0758024/DSPAPI/[4/0/0 (67)]/dsp_query_info:
    Request ID=1, Reset Flag=FALSE Q:PO_Delay PO_Error TX RX
```

```
.
.
.
```

Examples

This output shows the fax relay statistics.

```
Router# debug voip dspapi
voip dspapi debugging is ON
.
.
.
May  7 21:32:16.472 UTC: //8/D6635DD58005/DSPAPI/[1/0:3]/dsp_playout_error_stats:
    Request ID=1
    Concealment: Predictive=0(ms), Interpolative=0(ms), Silence=0(ms)
    Retro Memory update=0(na)(ms), Buffer overflow=0(ms)
    Talkspurt endpoint detection errors=0
May  7 21:32:18.996 UTC: //8/D6635DD58005/DSPAPI/[1/0:3]/dsp_query_info:
    Request ID=1, Reset Flag=FALSE Q:PO_Delay PO_Error TX RX
May  7 21:32:18.996 UTC: np_vsmgr_dispatch_voice_rsp(1/3): VOICE_LINK_INFO_RSP_NTF Received

May  7 21:32:18.996 UTC: request_id = 0x01, request_type = 0x0F
May  7 21:32:18.996 UTC: VOICE_TRANSMIT_STATS(1/3): num_voice_packets 36 num_sig_packets 0
 num_cn_packets 1 transmit_duration AD2 end_point_detection 0
May  7 21:32:18.996 UTC: //8/D6635DD58005/DSPAPI/[1/0:3]/dsp_tx_stats:
    Request ID=1, Packets: Voice=54, Signaling=0, ComfortNoise=1
    TX duration=2770(ms): Voice=0(ms), FAX=0(na)(ms)
May  7 21:32:18.996 UTC: VOICE_RECEIVE_STATS(1/3): num_voice_packets 20 num_sig_packets 0
num_cn_packets 2 receive_duration AD2  voice_receive_duration 0 num_pos_packets 0
num_bph_packets 0 num_late_packets 0 num_early_packets 0
May  7 21:32:18.996 UTC: //8/D6635DD58005/DSPAPI/[1/0:3]/dsp_rx_stats:
    Request ID=1, Packets: Voice=32, Signalling=0, ComfortNoise=2
    RX duration=2770(ms): Voice=0(ms), FAX=0(na)(ms)
    Packets: Bad Sequence=0, Bad Protocol=0, Late=0, Early=0
May  7 21:32:18.996 UTC: VOICE_PLAYOUT_DELAY_STATS(1/3): curr_playout_delay 5A
min_playout_delay 5A max_playout_delay 5A clock offset 2F07E72
May  7 21:32:19.000 UTC: //8/D6635DD58005/DSPAPI/[1/0:3]/dsp_playout_delay_stats:
    Request ID=1, Current=90(ms), MIN=90(ms), MAX=90(ms)
    Clock offset=49315442(ms), Inter arrival jitter=0(na)(ms)
May  7 21:32:19.000 UTC: VOICE_PLAYOUT_ERROR(1/3): pred_conceal 0x0 inter_conceal 0x0
silence_conceal 0x0 buffer_overflow 0x0 endpt_det_error 0x0
May  7 21:32:19.000 UTC: //8/D6635DD58005/DSPAPI/[1/0:3]/dsp_playout_error_stats:
    Request ID=1
    Concealment: Predictive=0(ms), Interpolative=0(ms), Silence=0(ms)
    Retro Memory update=0(na)(ms), Buffer overflow=0(ms)
    Talkspurt endpoint detection errors=0
May  7 21:32:21.456 UTC: VOICE_DET_STATUS_CHANGE_NTF(1/3): detector mask: 1 timestamp
51709BF8
May  7 21:32:21.456 UTC: //8/D6635DD58005/DSPAPI/[1/0:3]/dsp_voice_det_status_change:
    Status=1, Timestamp=1366334456, Tone ID=0, Trigger=TRUE
May  7 21:32:21.464 UTC: //8/D6635DD58005/DSPAPI/[1/0:3]/dsp_voice_config_params:
May  7 21:32:21.464 UTC:     1 parameter
    [0] PAK_SUPPRESS: 1
May  7 21:32:21.464 UTC: //8/D6635DD58005/DSPAPI/[1/0:3]/dsp_stop_service:
    G729IETF (26)
May  7 21:32:21.464 UTC: //8/D6635DD58005/DSPAPI/[1/0:3]/dsp_fax_config_params:
    1 parameters
May  7 21:32:21.464 UTC: //8/D6635DD58005/DSPAPI/[1/0:3]/dsp_open_service:
    FAX_RELAY (27)
May  7 21:32:21.464 UTC: //8/D6635DD58005/DSPAPI/[1/0:3]/dsp_is_call_pending:
    Call is not PENDING
May  7 21:32:21.464 UTC: //8/D6635DD58005/DSPAPI/[1/0:3]/dsp_set_call_pending:
    Set PENDING state
May  7 21:32:21.504 UTC: //8/D6635DD58005/DSPAPI/[1/0:3]/dsp_fax_query_link_info:
    Request ID=0, Group ID=1
May  7 21:32:21.504 UTC: vsm(1/3): np_vsmgr_voice_state_change() - state IDLE
May  7 21:32:21.504 UTC: //8/D6635DD58005/DSPAPI/[1/0:3]/dsp_is_call_pending:
    Call is PENDING
May  7 21:32:21.504 UTC: //8/D6635DD58005/DSPAPI/[1/0:3]/dsp_reset_call_pending:
    Reset PENDING state
May  7 21:32:21.504 UTC: //8/D6635DD58005/DSPAPI/[1/0:3]/dsp_call_status:
    Status=PENDING_SUCCESS
May  7 21:32:21.504 UTC: //8/D6635DD58005/DSPAPI/[1/0:3]/dsp_fax_config_params:
    11 parameters
May  7 21:32:21.504 UTC: //8/D6635DD58005/DSPAPI/[1/0:3]/dsp_start_service:
    FAX_RELAY (27)
May  7 21:32:21.504 UTC: //8/D6635DD58005/DSPAPI/[1/0:3]/dsp_set_call_pending:
    Set PENDING state
May  7 21:32:22.556 UTC: vsm(1/3): np_vsmgr_voice_state_change() - state ACTIVE
May  7 21:32:22.556 UTC: //8/D6635DD58005/DSPAPI/[1/0:3]/dsp_is_call_pending:
    Call is PENDING
May  7 21:32:22.556 UTC: //8/D6635DD58005/DSPAPI/[1/0:3]/dsp_reset_call_pending:
    Reset PENDING state
May  7 21:32:22.556 UTC: //8/D6635DD58005/DSPAPI/[1/0:3]/dsp_call_status:
    Status=PENDING_SUCCESS
May  7 21:32:22.564 UTC: FAX_RELAY_LINK_INFO_RSP_NTF: slot 1 port 3 timestamp 76082770
fr-entered (20ms)
May  7 21:32:22.564 UTC: chan_id [3/1:D (8)] np_vsmgr_fax_relay_link_info_response:
May  7 21:32:29.712 UTC: //8/D6635DD58005/DSPAPI/[1/0:3]/dsp_fax_relay_t30_decode:
    T30 msg : 0x4
May  7 21:32:30.436 UTC: //8/D6635DD58005/DSPAPI/[1/0:3]/dsp_fax_relay_t30_decode:
    T30 msg : 0x2
May  7 21:32:30.784 UTC: //8/D6635DD58005/DSPAPI/[1/0:3]/dsp_fax_relay_t30_decode:
    T30 msg : 0x1
May  7 21:32:33.936 UTC: //8/D6635DD58005/DSPAPI/[1/0:3]/dsp_fax_relay_t30_decode:
    T30 msg : 0x42
May  7 21:32:34.280 UTC: //8/D6635DD58005/DSPAPI/[1/0:3]/dsp_fax_relay_t30_decode:
    T30 msg : 0x41
May  7 21:32:39.676 UTC: //8/D6635DD58005/DSPAPI/[1/0:3]/dsp_fax_relay_t30_decode:
    T30 msg : 0x21
May  7 21:32:39.676 UTC: np_fax_relay_t30_decode : Rx Direction
May  7 21:32:39.736 UTC: FARELAY_INIT_HS_MOD : 0x8
May  7 21:33:10.385 UTC: //8/D6635DD58005/DSPAPI/[1/0:3]/dsp_fax_relay_t30_decode:
    T30 msg : 0x7D
May  7 21:33:13.073 UTC: //8/D6635DD58005/DSPAPI/[1/0:3]/dsp_fax_relay_t30_decode:
    T30 msg : 0x31
May  7 21:33:15.217 UTC: //8/D6635DD58005/DSPAPI/[1/0:3]/dsp_fax_relay_t30_decode:
    T30 msg : 0x5F
May  7 21:33:16.073 UTC: FAX_RELAY_DET_STATUS_CHANGE: slot: 1 port: 3 detector mask 0x2
May  7 21:33:16.073 UTC: //8/D6635DD58005/DSPAPI/[1/0:3]/dsp_fax_det_status:
    Status=2, Timestamp=716372818
May  7 21:33:16.073 UTC: //8/D6635DD58005/DSPAPI/[1/0:3]/dsp_query_info:
    Request ID=5, Reset Flag=FALSE Q:FaxRelay
May  7 21:33:16.097 UTC: FAX_RELAY_DATA_PUMP_STATS(1/3) - valid:0x3FFC1F55 state_code:0x1
level:0x18 phase_jitter:0x0 freq_offset:0x0 eqm:0x7FFE jit_depth:0x38B jit_buf_ov:0x0
tx_paks:0x5A rx_pkts:0x62C inv_pkts:0x0 oos_pkts:0x0 hs_mod:0x8 init_hs_mod:0x8 tx_pgs:0x0
 rx_pgs:0x1 ecm:0x1 nsf_country:0x0 nsf_manuf_len:0x20
nsf_manuf:0031B8EE80C48511DD0D0000DDDD0000DDDD00000000000000022ED00B0A400 encap:0x1
pkt_loss_con:0x0
May  7 21:33:16.097 UTC: //8/D6635DD58005/DSPAPI/[1/0:3]/dsp_fax_relay_stats:
    Request ID=5, MAX jitter depth=907, MAX net RX qdepth=0(na)
    Jitter buffer overflow=0, Net RX qoverflow=0(na)
    Packets: TX=90 TX drops=0(na)
    Packets: RX=1580 RX loss=0(na), RX invalid=0, RX OOSequence=0
    HS modulation=8, Pages: TX=0 RX=1
    MAX TX In qdepth=0(na), MAX RX Out qdepth=0(na)
    MAX HS buffer usage=0(na), TX In qoverflow=0(na), RX Out qoverflow=0(na)
    FAX: State=1, level=24, Phase jitter=0, Frequency offset=0, EQM=32766
    Initial HS modulation=8, Fax Direction=2, ECM Enabled=1
    NSF Countery Code=0, nsf_manuf_code[32]=0031
    Encapsulation Protocol=1, Pkt Loss Conceal=0
May  7 21:33:16.101 UTC: //8/D6635DD58005/DSPAPI/[1/0:3]/dsp_stop_service:
    FAX_RELAY (27)
May  7 21:33:16.101 UTC: //8/D6635DD58005/DSPAPI/[1/0:3]/dsp_open_service:
    G729IETF (26)
May  7 21:33:16.101 UTC: //8/D6635DD58005/DSPAPI/[1/0:3]/dsp_is_call_pending:
    Call is not PENDING
May  7 21:33:16.101 UTC: //8/D6635DD58005/DSPAPI/[1/0:3]/dsp_set_call_pending:
    Set PENDING state
May  7 21:33:16.985 UTC: FAX_RELAY_LINK_INFO_RSP_NTF: slot 1 port 3 timestamp 76518179
fr-end
May  7 21:33:17.001 UTC: vsm(1/3): np_vsmgr_voice_state_change() - state IDLE
May  7 21:33:17.001 UTC: //8/D6635DD58005/DSPAPI/[1/0:3]/dsp_is_call_pending:
    Call is PENDING
```

```
May  7 21:33:17.001 UTC: //8/D6635DD58005/DSPAPI/[1/0:3]/dsp_reset_call_pending:
    Reset PENDING state
May  7 21:33:17.001 UTC: //8/D6635DD58005/DSPAPI/[1/0:3]/dsp_call_status:
    Status=PENDING_SUCCESS
May  7 21:33:17.001 UTC: //8/D6635DD58005/DSPAPI/[1/0:3]/dsp_voice_config_params:
May  7 21:33:17.001 UTC:     4 parameters
.
.
.
```

Related Commands

Command	Description
debug voip hpi	Enables debugging for HPI message events.

debug voip dump-file-acct

To display debugging messages related to file accounting flushing processes, use the **debug voip dump-file-acct** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voip dump-file-acct

no debug voip dump-file-acct

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging of file accounting processes is not enabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.4(15)XY	This command was introduced.
12.4(20)T	This command was integrated into Cisco IOS Release 12.4(20)T.

Usage Guidelines

This command displays event and error information about the file accounting processes for flushing the buffer and writing the call detail records (CDRs) to the file.

Examples

The following example displays output from the **debug voip dump-file-acct**command:

```
Router# debug voip dump-file-acct

*May 10 06:31:10.187: voice_file_acct_write:
*May 10 06:31:10.187: file_acct_write_local: file accounting buffer overflow,dumping to
file
*May 10 06:31:10.187: voice_file_acct_initiate_dump_to_file: ctx_id=2, url=flash:cdr
*May 10 06:31:10.187: voice_file_acct_lock, ctx_id=2, refcnt=2
*May 10 06:31:10.187: create_file_acct_buffer: buffer of 63k created from chunk 0x46B5F474
*May 10 06:31:10.187: file_acct_write_local: message (len=640) written to file_acct:
30080 bytes left
*May 10 06:31:10.191: handle_file_acct_dump_request
*May 10 06:31:10.191: handle_file_acct_dump_request: pick up dump request (ctx_id=2)
*May 10 06:31:10.191: open_file_acct_dump_file:
url=flash:cdr_ragdenCME1_05_10_2007_06_30_28.191  < == shows url.
*May 10 06:31:10.215:  Secondary mode file acct is successful
*May 10 06:31:10.215: handle_file_acct_dump_request :to_write is 29748  <== shows how much
 is written to.
*May 10 06:31:10.219: : File accounting,write successful to file
*May 10 06:31:10.219: handle_file_acct_dump_request :to_write is 640
*May 10 06:31:10.219: : File accounting,write successful to file
*May 10 06:31:10.323: voice_file_acct_unlock, ctx_id=2  refcnt=1
```

Related Commands

Command	Description
debug voip fileacct	Displays debugging messages related to generating attributes for file accounting.
gw-accounting	Enables an accounting method for collecting CDRs.
primary	Sets the primary location for storing the CDRs generated for file accounting.

debug voip eddri

To turn on debugging for the event dispatcher and data repository interface (EDDRI), use the **debug voip eddri**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voip eddri {**event**| **timers**| **prefix**| **all**}

no debug voip eddri {**event**| **timers**| **prefix**| **all**}

Syntax Description

event	Turns on debugging for EDDRI events.
timers	Turns on debugging for EDDRI timers.
prefix	Turns on debugging for the prefix database.
all	Turns on debugging all EDDRI activities.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(1)	This command was introduced.

Usage Guidelines

There is always a performance penalty when using **debug** commands.

The EDDRI notifies TGREP when an attribute changes on some subsystems. EDDRI interacts with the dial peer subsystem, the trunk group subsystems, call control API (CCAPI) subsystem and the customer relationship management (CRM) subsystem to notify changes in particular attributes. EDDRI is responsible for creating the prefix database.

Examples

The following example shows sample output from the **debug voip eddri**command:

```
21:00:53: eddri_interesting_ac_pt: new AC_curr 22 FD_curr -5 SD_curr -5
21:00:53: eddri_interesting_ac_pt: percent trigger diff 4
21:00:53: eddri_interesting_ac_pt: Interesting Point
21:00:53: eddri_send_prefix_event_to_clients : reason 0x40 num_prefix 1
```

With the send prefix event the available circuits value and the triggers for reporting are updated.

```
21:00:53: eddri_send_prefix_event_to_clients attr 0xFF ev_id 1 qid 0x64209230 reason 0x40
eddri_dequeue_event : dequeue event
21:00:53: eddri_interesting_ac_pt : tc 23 IAC 22 lwm 5 hwm 50 pct_trigger 2 oneMinusW 933
```

```
21:00:53: eddri_interesting_ac_pt: old AC_curr 23 FD_curr 0 SD_curr 0
21:00:53: eddri_interesting_ac_pt: new AC_curr 22 FD_curr -5 SD_curr -5
21:00:53: eddri_interesting_ac_pt: percent trigger diff 4
21:00:53: eddri_interesting_ac_pt: Interesting Point
21:00:53: eddri_send_prefix_event_to_clients : reason 0x40 num_prefix 1
```

Related Commands

Command	Description
debug tgrep error	Turns on debugging for any errors in functioning.
debug tgrep events	Turns on debugging for main events occurring throughout the subsystem.
debug tgrep fsm	Turns on debugging for FSM activity.
debug tgrep io	Turns on debugging for detailed socket level activities.
debug tgrep messages	Turns on debugging for the movement of TGREP messages.
debug tgrep msgdump	Turns on debugging for the dump of the details of TGREP messages.
debug tgrep timer-event	Turns on debugging for events that are related to the timer.
debug tgrep timers	Turns on debugging for timer activity.
debug tgrep tripr	Turns on debugging for the TRIP Reporter.
show voice eddri prefix	Shows applicable prefixes for the EDDRI.

debug voip enum

To view Voice over IP (VoIP) telephone number mapping (ENUM) information, use the **debug voip enum** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voip enum {**detail**| **summary**}

no debug voip enum {**detail**| **summary**}

Syntax Description

detail	Displays detailed output.
summary	Displays summary output.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(11)T	This command was introduced.

Usage Guidelines

Disable console logging and use buffered logging before using the **debug voip enum** command. Using the **debug voip enum** command generates a large volume of debugs, which can affect router performance.

Examples

The following shows sample output from the **debug voip enum detail** command:

The output shows the match number as 5108891234, enum table as 10. Rule 1 in table 10 matched the pattern and after applying the replacement rule, the resulting string is 5108891234. The enum query is sent out for the domain 4.3.2.1.9.8.8.0.1.5.e164.cisco.com. The output then shows the matching Naming Authority Pointer (NAPTR) records obtained in the response. The records are then processed and the final URLs (contact lists) are shown toward the end.

```
Router# debug voip enum detail
enum_resolve_domain:match_num 5108891234 table_indx 10
enum_resolve_domain:rule 1 result string 5108891234
generate_enum_search_string :search string 4.3.2.1.9.8.8.0.1.5.e164.cisco.com
enum_dns_query:name = 4.3.2.1.9.8.8.0.1.5.e164.cisco.com type = 35, ns_server = 0
order 100 pref 10 service sip+E2U flag U
regexp /^.*$/sip:5108891234@1.8.50.14/ replacement
order 200 pref 10 service h323+E2U flag U
regexp /^.*$/h323:5555@1.5.1.1/ replacement
num_elem = 2
NAPTR Record :order 100 pref 10 service sip+E2U
              flags U regexp /^.*$/sip:5108891234@1.8.50.14/
              replacement
```

```
NAPTR Record :order 200 pref 10 service h323+E2U
               flags U regexp /^.*$/h323:5555@1.5.1.1/
               replacement
decode_naptr_record :re_string ^.*$
decode_naptr_record :re_substitution_string sip:5108891234@1.8.50.14
decode_naptr_record :re_flags_string
U_FLAG case, stopping query
new_e164_user sip:5108891234@1.8.50.14
decode_naptr_record :re_string ^.*$
decode_naptr_re
tahoe13#cord :re_substitution_string h323:5555@1.5.1.1
decode_naptr_record :re_flags_string
U_FLAG case, stopping query
new_e164_user h323:5555@1.5.1.1
contact_list :
              sip:5108891234@1.8.50.14
contact_list :
              h323:5555@1.5.1.1
enum_resolve_domain:contact_list 64558450
```

A sample output of the **debug voip enum summary** command is shown below.

The output shows the matching number, the enum table used and the rule in the table that matched the number along with the resulting string. Note that this output is a subset of the output from **debug voip enum detail** command.

```
Router# debug voip enum summary
enum_resolve_domain:match_num 5108891234 table_indx 10
enum_resolve_domain:rule 1 result string 5108891234
```

The table below provides an alphabetical listing of the **debug voip enum** command fields and a description of each field.

Table 61: debug voip enum Field Descriptions

Field	Description
contact_list	Final list of URLs that the gateway will try to contact as an attempt to place the call.
flag	Flag value of a NAPTR record as defined in RFC 2915.
match_num	Number to be used for matching against the enum match table.
name	Fully qualified domain name sent out to Domain Name System (DNS) server
ns_server	Address of the DNS server. If 0, the DNS server configured on the gateway is used.
num_elem	Number of records received in the response.
order	Order in the record, as defined in RFC 2915.
pref	Preference of the record, as defined in RFC 2915.
regexp	Regular expression of the record, as defined in RFC 2915.

Field	Description
replacement	Replacement string of the record, as defined in RFC 2915.
re_flags_string	Flag indicating whether matching and replacement should be case sensitive: • i = Case insensitive • otherwise = Case sensitive
re_string	The first part of the regexp, delimited by "/". This is used to match the incoming string. Refer to RFC 2915.
re_substitution_string	The second part of regexp, delimited by "/".
result string	String that results when match_num is taken through the enum match table for a match. This string will be used to form a FQDN.
rule	Rule number that matched match_num in the enum match table.
search string	String sent out to the DNS server.
service	Service field of the NAPTR record. Refer to RFC 2915.
table_indx	Index of the enum match table picked for this call.
type	Type of record requested in the query: 35 = NAPTR 33 = DNS Service (SRV)

Related Commands

Command	Description
rule (ENUM configuration)	Defines the rule pattern for an ENUM match table.
show voice enum-match-table	Displays the ENUM match table rules.
test enum	Tests the ENUM match table rules.
voice enum-match-table	Initiates the ENUM match table definition.

debug voip event-log

To enable debugging of the event log module, use the **debug voip event-log**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voip event-log

no debug voip event-log

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(8)T	This command was introduced.

Examples

The following is sample output from the **debug voip event-log** command:

```
Router# debug voip event-log

voip event-log debugging is on
Router#
*Jul 18 22:22:45.417: voice_elog_open: ctx_id=1F, size=4, url=
*Jul 18 22:22:45.417: lock_elog, ctx_id=1F, refcnt=1
*Jul 18 22:22:45.417: voice_elog_write:
*Jul 18 22:22:45.417: elog_write_local: message (len=143) written to elog:
1F:1058566965:584:INFO: Call setup indication received, called = 4085550198,  calling =
52927, echo canceller = enable, direct inward dialing 3953 bytes left
*Jul 18 22:22:45.417: voice_elog_write:
*Jul 18 22:22:45.417: elog_write_local: message (len=38) written to elog:
1F:1058566965:585:INFO: Dialpeer = 1 3915 bytes left
*Jul 18 22:22:45.421: voice_elog_open: ctx_id=B, size=4, url=
*Jul 18 22:22:45.421: lock_elog, ctx_id=B, refcnt=1
*Jul 18 22:22:45.421: voice_elog_write:
*Jul 18 22:22:45.421: elog_write_local: message (len=114) written to elog:
B:1058566965:586:INFO: Session started for App-type = generic, URL =
tftp://demo/scripts/master/generic.vxml 3982 bytes left
*Jul 18 22:22:45.421: voice_elog_write:
*Jul 18 22:22:45.421: elog_write_local: message (len=69) written to elog:
B:1058566965:587:INFO: Incoming Telephony call received, LegID = 1F 3913 bytes left
*Jul 18 22:22:45.421: voice_elog_write:
*Jul 18 22:22:45.421: elog_write_local: message (len=89) written to elog:
B:1058566965:588:INFO: LegID = 1F: Calling = 4085550198, called = 52927, dial peer = 1 3824
 bytes left
*Jul 18 22:22:45.421: voice_elog_write:
*Jul 18 22:22:45.421: elog_write_local: message (len=66) written to elog:
B:1058566965:589:INFO: LegID = 1F: Leg State = LEG_INCCONNECTED 3758 bytes left
*Jul 18 22:22:45.433: voice_elog_write:
*Jul 18 22:22:45.437: elog_write_local: message (len=42) written to elog:
1F:1058566965:590:INFO: Digit collection 3873 bytes left
*Jul 18 22:22:45.437: voice_elog_write:
*Jul 18 22:22:45.437: elog_write_local: message (len=57) written to elog:
1F:1058566965:591:INFO: Call connected using codec None 3816 bytes left
*Jul 18 22:22:45.437: voice_elog_write:
```

```
*Jul 18 22:22:45.437: elog_write_local: message (len=85) written to elog:
B:1058566965:592:INFO: Playing prompt #1: tftp://172.19.139.245/audio/ch_welcome.au 3673
bytes left
Router#
*Jul 18 22:22:55.942: voice_elog_write:
*Jul 18 22:22:55.942: elog_write_local: message (len=51) written to elog:
B:1058566975:593:ERR : Prompt play setup failure.  3622 bytes left
*Jul 18 22:22:55.942: voice_elog_write:
*Jul 18 22:22:55.942: elog_write_local: message (len=65) written to elog:
B:1058566975:594:INFO: Script received event = "error.badfetch" 3557 bytes left
*Jul 18 22:22:56.918: voice_elog_write:
*Jul 18 22:22:56.918: elog_write_local: message (len=98) written to elog: 1F:1058
Router#
566976:595:INFO: Inform application call disconnected (cause = normal call clearing (16))
3718 bytes left
*Jul 18 22:22:56.918: voice_elog_write:
*Jul 18 22:22:56.918: elog_write_local: message (len=78) written to elog:
B:1058566976:596:INFO: Script received event = "telephone.disconnect.hangup"
 3479 bytes left
*Jul 18 22:22:56.922: voice_elog_write:
*Jul 18 22:22:56.922: elog_write_local: message (len=89) written to elog:
B:1058566976:597:INFO: LegID = 1F: Call disconnected, cause = normal call clearing (16)
3390 bytes left
*Jul 18 22:22:56.922: voice_elog_write:
*Jul 18 22:22:56.922: elog_write_local: message (len=79) written to elog:
1F:1058566976:598:INFO: Call disconnected (cause = normal call clearing (16))
 3639 bytes left
*Jul 18 22:22:56.930: voice_elog_write:
*Jul 18 22:22:56.930: elog_write_local: message (len=39) written to elog:
1F:1058566976:599:INFO: Call released
 3600 bytes left
*Jul 18 22:22:56.930: voice_elog_close, ctx_id=1F voice_elog_close, ctx_id=19
*Jul 18 22:22:56.930: unlock_elog, ctx_id=19, refcnt=0
*Jul 18 22:22:56.930: delete_elog, ctx_id=19
*Jul 18 22:22:56.930: voice_elog_write:
*Jul 18 22:22:56.930: elog_write_local: message (len=59) written to elog:
B:1058566976:600:INFO: Session done, terminating cause = 3331 bytes left
*Jul 18 22:22:56.930: voice_elog_close, ctx_id=B
```

Related Commands

Command	Description
call application event-log	Enables event logging for voice application instances.
debug voip ais	Enables debugging of the AIS database.

debug voip fastpath

To turn on debugging to monitor VoIP fastpath activity, use the **debug voip fastpath**command in privileged EXEC mode. To turn off VoIP fastpath debugging, use the **no** form of this command.

debug voip fastpath[**invalidate**][*slot*/*port*]

no debug voip fastpath[**invalidate**][*slot*/*port*]

Syntax Description

invalidate	(Optional) Turns on debugging for VoIP fastpath cache invalidation.
slot / *port*	(Optional) Slot and port to be debugged. Slash mark is required.

Command Default

VoIP fastpath debugging does not occur.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.4(21)	This command was introduced on the Cisco AS5400XM and AS5350XM.

Usage Guidelines

The **debug voip fastpath** command displays the details on every packet that is being switched via fastpath. The **debug voip fastpath invalidate** command displays the details of cache invalidation and cache update. The **debug voip fastpath** command and its options are interchangeable with the **debug voice fastpath** command.

VoIP fastpath is enabled by default. In order to disable it, issue the **no voip-fastpath enable** command in global configuration mode.

When VoIP fastpath is enabled, the IP address and User Datagram Protocol (UDP) port number information for the logical channel that is opened for a specific call are cached. VoIP fastpath prevents the RTP stream from reaching the application layer. Instead, the packets are forwarded at a lower layer to help reduce CPU utilization in high call-volume scenarios.

When supplementary services such as hold or transfer are used, VoIP fastpath causes the router to stream the audio to the cached IP address and UDP port. The new logical channel information (generated after a call on hold is resumed or after a transfer is completed) is disregarded. Traffic must go to the application layer constantly so that redefinition of the logical channel is taken into account and audio is streamed to the new IP address and UDP port pair. Therefore, be sure to disable VoIP fastpath in order to support supplementary services.

Note The **debug voip fastpath** command should be enabled only when there is light traffic on the gateway. Enabling this command can affect the functionality of the gateway.

Examples

The following example shows how to turn on VoIP fastpath debugging, shows how to use the **show debug** command to display what debugging functions are enabled, and provides sample output for the debugging function:

```
Router# debug voip fastpath
Fastpath related debugging is on
Router# show debug
 fastpath:
  Fastpath related debugging is on
Router#
*Nov 14 08:22:35.971: NP VPD(2/01): pak sent via fastpath,part=0x652DEE80 ret=0x000003
len=32
*Nov 14 08:22:35.987: NP VPD(2/01): pak sent via fastpath,part=0x652DEEC0 ret=0x000003
len=32
*Nov 14 08:22:36.011: NP VPD(2/01): pak sent via fastpath,part=0x652DEF00 ret=0x000003
len=32
*Nov 14 08:22:36.031: NP VPD(2/01): pak sent via fastpath,part=0x652DEF40 ret=0x000003
len=32
*Nov 14 08:22:36.051: NP VPD(2/01): pak sent via fastpath,part=0x652DEF80 ret=0x000003
len=32
*Nov 14 08:22:36.071: NP VPD(2/01): pak sent via fastpath,part=0x652DEFC0 ret=0x000003
len=32
*Nov 14 08:22:36.095: NP VPD(2/01): pak sent via fastpath,part=0x652DF000 ret=0x000003
len=32
*Nov 14 08:22:36.111: NP VPD(2/01): pak sent via fastpath,part=0x652DF040 ret=0x000003
len=32
*Nov 14 08:22:36.131: NP VPD(2/01): pak sent via fastpath,part=0x652DF080 ret=0x000003
len=32
*Nov 14 08:22:36.151: NP VPD(2/01): pak sent via fastpath,part=0x652DF0C0 ret=0x000003
len=32
*Nov 14 08:22:36.171: NP VPD(2/01): pak sent via fastpath,part=0x652DF100 ret=0x000003
len=32
*Nov 14 08:22:36.195: NP VPD(2/01): pak sent via fastpath,part=0x652DF140 ret=0x000003
len=32
*Nov 14 08:22:36.207: NP VPD(2/01): pak sent via fastpath,part=0x652DF180 ret=0x000003
len=32
*Nov 14 08:22:36.231: NP VPD(2/01): pak sent via fastpath,part=0x652DF1C0 ret=0x000003
len=32
*Nov 14 08:22:36.251: NP VPD(2/01): pak sent via fastpath,part=0x652DF200 ret=0x000003
len=32
*Nov 14 08:22:36.271: NP VPD(2/01): pak sent via fastpath,part=0x652DF240 ret=0x000003
len=32
*Nov 14 08:22:36.291: NP VPD(2/01): pak sent via fastpath,part=0x652DF280 ret=0x000003
len=32
*Nov 14 08:22:36.315: NP VPD(2/01): pak sent via fastpath,part=0x652DF2C0 ret=0x000003
len=32
*Nov 14 08:22:36.331: NP VPD(2/01): pak sent via fastpath,part=0x652DF300 ret=0x000003
len=32
*Nov 14 08:22:36.351: NP VPD(2/01): pak sent via fastpath,part=0x652DF340 ret=0x000003
len=32
*Nov 14 08:22:36.371: NP VPD(2/01): pak sent via fastpath,part=0x652DF380 ret=0x000003
len=32
*Nov 14 08:22:36.391: NP VPD(2/01): pak sent via fastpath,part=0x652DF3C0 ret=0x000003
len=32
```

The following example shows how to use the **debug voip fastpath***slot*/*port* command to debug slot 2, port 13 on the router:

```
Router# debug voip fastpath 2/013
Fastpath related debugging is on
```

```
*Nov 14 08:28:00.623: NP VPD(2/13): pak sent via fastpath,part=0x652DFFC0 ret=0x000003
len=32
*Nov 14 08:28:00.643: NP VPD(2/13): pak sent via fastpath,part=0x652E0000 ret=0x000003
len=32
*Nov 14 08:28:00.659: NP VPD(2/13): pak sent via fastpath,part=0x652E0080 ret=0x000003
len=32
*Nov 14 08:28:00.831: NP VPD(2/13): pak sent via fastpath,part=0x652E0280 ret=0x000003
len=32
*Nov 14 08:28:00.855: NP VPD(2/13): pak sent via fastpath,part=0x652E0300 ret=0x000003
len=32
*Nov 14 08:28:00.867: NP VPD(2/13): pak sent via fastpath,part=0x652E0380 ret=0x000003
len=32
*Nov 14 08:28:01.031: NP VPD(2/13): pak sent via fastpath,part=0x652E0540 ret=0x000003
len=32
*Nov 14 08:28:01.051: NP VPD(2/13): pak sent via fastpath,part=0x652E0580 ret=0x000003
len=32
*Nov 14 08:28:01.075: NP VPD(2/13): pak sent via fastpath,part=0x652E0640 ret=0x000003
len=32
*Nov 14 08:28:01.231: NP VPD(2/13): pak sent via fastpath,part=0x652E0840 ret=0x000003
len=32
*Nov 14 08:28:01.251: NP VPD(2/13): pak sent via fastpath,part=0x652E07C0 ret=0x000003
len=32
*Nov 14 08:28:01.271: NP VPD(2/13): pak sent via fastpath,part=0x652E0900 ret=0x000003
len=32
*Nov 14 08:28:01.439: NP VPD(2/13): pak sent via fastpath,part=0x652E0AC0 ret=0x000003
len=32
*Nov 14 08:28:01.463: NP VPD(2/13): pak sent via fastpath,part=0x652E0B40 ret=0x000003
len=32
*Nov 14 08:28:01.483: NP VPD(2/13): pak sent via fastpath,part=0x652E0BC0 ret=0x000003
len=32
```

The following example shows how to enable debugging for fastpath cache invalidation on slot 2, port 17, and shows how to display sample output for the debugging function:

```
Router# debug voip fastpath invalidate 2/17

 Fastpath cache invalidation related  debugging is on
Router# show voip call summary
PORT           CODEC    VAD VTSP STATE            VPM STATE
============== ======== === ==================== ======================
6/4:0.20        g729r8    y  S_CONNECT             CSM_OC6_CONNECTED
6/4:0.21        g729r8    y  S_CONNECT             CSM_OC6_CONNECTED
Router# show spe | i a
Country code config : default T1 (u Law)
Country code setting: e1-default
Port state: (s)shutdown   (r)recovery  (t)test  (a)active call
            (b)busiedout  (d)download  (B)bad   (p)busyout pending
Call type : (m)modem      (d)digital   (v)voice (f)fax-relay       (_)not in use
Summary   :
Ports     : Total  540  In-use     2  Free   514  Disabled    24
Calls     : Modem    0  Digital    0  Voice    2  Fax-relay    0
                     SPE          SPE     SPE  SPE   Port         Call
SPE#    Port #       State        Busyout Shut Crash State        Type
2/02    0012-0017    ACTIVE             0    0     0 _____a       _____v
2/03    0018-0023    ACTIVE             0    0     0 a_____       v_____
Router# show logging

Syslog logging: enabled (274 messages dropped, 20 messages rate-limited,
                0 flushes, 0 overruns, xml disabled, filtering disabled)
    Console logging: disabled
    Monitor logging: level debugging, 0 messages logged, xml disabled,
                     filtering disabled
    Buffer logging: level debugging, 1018 messages logged, xml disabled,
                    filtering disabled
    Logging Exception size (8192 bytes)
    Count and timestamp logging messages: disabled
    Trap logging: level informational, 133 message lines logged

Log Buffer (1000000 bytes):
*Nov 14 08:40:36.499:  NP VPD (2/17): Cached header parameter values: header size : 28,
payload size : 13, ssrc : 0x24DB1F03, udp chksum : 0x0
*Nov 14 08:40:36.499:  NP VPD (2/17): Cached IP/UDP pkt details: dest-ip: 31.31.31.1
```

```
src-ip: 31.31.31.3   dport: 0x4070  sport: 0x43A6
*Nov 14 08:40:40.851:  NP VPD (2/17): Cached header parameter values: header size : 28,
payload size : 32, ssrc : 0x24DB1F03, udp chksum : 0x0
*Nov 14 08:40:40.851:  NP VPD (2/17): Cached IP/UDP pkt details: dest-ip: 31.31.31.1
src-ip: 31.31.31.3   dport: 0x4070  sport: 0x43A6
*Nov 14 08:40:40.939:  NP VPD (2/17): Cache being cleared due to change in payload size old
 payload size : 32  new rx payload size : 13 cached ssrc : 24DB1F03
```

Related Commands

Command	Description
debug voice fastpath	Turns on debugging to monitor voice fastpath packets.
show voice call	Displays the call status information for voice ports.
voice fastpath enable	Turns on voice fastpath.

debug voip fileacct

To display debugging messages related to voice attributes for file accounting, use the **debug voip fileacct** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voip fileacct

no debug voip fileacct

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging of file accounting is not enabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.4(15)XY	This command was introduced.
12.4(20)T	This command was integrated into Cisco IOS Release 12.4(20)T.

Usage Guidelines

This command displays details about the attributes captured in call detail records (CDRs) and their values for the file accounting feature.

Examples

The following example displays output from the **debug voip fileacct**command:

```
Router# debug voip fileacct

*May 10 06:27:43.719: : add attr:47A815E4 clid(21) 4 5000
*May 10 06:27:43.719: new list: 0x4792614C prev list: 0x47A815D0
*May 10 06:27:43.719: : add attr:47926160 dnis(22) 0
*May 10 06:27:43.719: new list: 0x47C3A2C0 prev list: 0x4792614C
*May 10 06:27:43.719: : add attr:47C3A2D4 subscriber(106) 11 RegularLine
*May 10 06:27:43.719: new list: 0x4517FC04 prev list: 0x47C3A2C0
*May 10 06:27:43.719: : add attr:4517FC18 override_session_time(67) 4 0(0) Telephony Leg
*May 10 06:27:43.719: new list: 0x478C0CA4 prev list: 0x4517FC04
*May 10 06:27:43.719: : add attr:478C0CB8 h323-ivr-out(68) 14 Tariff:Unknown
*May 10 06:27:43.719: new list: 0x477EAFFC prev list: 0x478C0CA4
*May 10 06:27:43.719: : add attr:477EB010 h323-voice-quality(70) 1 0
*May 10 06:27:43.719: new list: 0x4783EF80 prev list: 0x477EAFFC
*May 10 06:27:43.719: : add attr:4783EF94 gw-rxd-cgn(94) 28 ton:0,npi:0,pi:0,si:0,#:5000
*May 10 06:27:43.719: list is 466C17A8, list->featurename is 0,feat id is 11205
*May 10 06:27:43.719: fcur is 466C17A8, attr is "TWC","05/10/2007
06:27:43.695","5000","",0,11205,6510EBF8 FDF611DB A527DA52 74E8B890,2BD8,"","","",""
*May 10 06:27:43.719: : del attr47B8E814 callID(1) 4 11224(2BD8)
*May 10 06:27:43.719: : del attr45250054 cdr type(2) 4 0(0)
*May 10 06:27:43.719: : del attr452C52F0 leg type(3) 4 1(1)
*May 10 06:27:43.719: : del attr47914064 h323-conf-id(4) 35 6510EBF8 FDF611DB A527DA52
74E8B890
```

Related Commands

Command	Description
debug voip dump-file-acct	Displays debugging messages related to file accounting flushing processes.
gw-accounting	Enables an accounting method for collecting CDRs.
primary	Sets the primary location for storing the CDRs generated for file accounting.

debug voip fpi call-rate

To enable the call-rate computation, use the **debug voip fpi call-rate**. To disable the debugging output, use the **no** form of this command

debug voip fpi call-rate

no debug voip fpi call-rate

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS XE Release 3.9S	This command was introduced.

Usage Guidelines

Use the **debug voip fpi call-rate** command in conjunction with the **show voip fpi call-rate**command.

debug voip h221

To debug telephony call control information, use the **debug voip h221**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voip h221 [**all**| **default**| **error** [**call [informational]**| **software [informational]**]| **function**| **individual**| **inout**| **raw [decode]**]

no debug voip h221

Syntax Description

all	(Optional) Enables all H.221 debugging, except the raw option.
default	(Optional) Activates function, inout, error call, and software debugging.
error	(Optional) Enables H.221 call error and software error debugging.
error [call]	(Optional) Enables H.221 major call processing error debugs related to the H.221 subsystem.
error [call [informational]]	(Optional) Enables H.221 major and informational call processing error debugs related to the H.221 subsystem.
error [software]	(Optional) Enables H.221 major software error debugs related to the H.221 subsystem.
error [software [informational]]	(Optional) Enables H.221 major and informational software error debugs related to the H.221 subsystem.
function	(Optional) Enables procedure tracing.
individual	(Optional) Activates individual H.221 debugging.
inout	(Optional) Enables subsystem inout debugging.
raw	(Optional) Displays raw BAS messages.
raw [decode]	(Optional) Decodes raw BAS data.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.4(11)T	This command was introduced.

Usage Guidelines

This command enables debugging for H.221 message events (voice telephony call control information).

Note This command provides the same results as the **debug voice h221** command.

Caution We recommend that you log the output from the **debug voip h221 all** command to a buffer, rather than sending the output to the console; otherwise, the size of the output could severely impact the performance of the gateway.

Use the **debug voip h221 individual** *x* command, (where *x* is an index number for a debug category), to activate a single debug, selected by index number instead of entering a group of debug commands. See the table below for a list of debug categories and corresponding index numbers.

Table 62: Indexes and Categories for the debug voip h221 individual command

Index Number	Debug Category
1, 2, 30, 31, 32	Secondary number exchange
5, 6, 14, 15, 16, 22	Audio mode/caps
7, 10, 12, 13, 17, 28	Video mode/caps
8, 9, 23	B-channel mode/caps
11, 24, 33	Miscellaneous command exchange
18	Bandwidth calculations
19, 20, 21	DSP configuration
3, 4, 25, 27, 42, 43	General caps/internal
26	Non-standard caps/command
29	Loop request
34, 35, 36, 37, 38, 39, 40, 41	BAS squelch

Examples

The raw keyword displays the raw BAS information coming from or to the DSP. It is displayed in a hexadecimal octet format. The **decode** option decodes the BAS information into a readable English format.

The following is sample output from the **debug voip h221 raw** decode command:

```
BAS=81:1 0 0 0 0 0 0 1: AUDIO CAPS=g711 a-law
BAS=82:1 0 0 0 0 0 1 0: AUDIO CAPS=g711 u-law
BAS=84:1 0 0 0 0 1 0 0: AUDIO CAPS=g722 48k
BAS=85:1 0 0 0 0 1 0 1: AUDIO CAPS=g728
BAS=F9:1 1 1 1 1 0 0 1: H.242 MBE start indication
BAS=02:0 0 0 0 0 0 1 0: H.242 MBE length=2
BAS=0A:0 0 0 0 1 0 1 0: H.242 MBE type=H.263 caps
BAS=8A:1 - - - - - - -: Always 1
BAS=8A:- 0 0 0 1 - - -: H.263 MPI=1
BAS=8A:- - - - - 0 1 -: H.263 FORMAT=h.263_cif
BAS=8A:- - - - - - - 0: No additional options
```

Related Commands

Command	Description
debug voip ccapi	Enables debugging for the call control application programming interface (CCAPI) contents.
debug voip rtp	Enables debugging for Real-Time Transport Protocol (RTP) named event packets.

debug voip h324

To debug video call control information, use the **debug voip h324**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voip h324 [**all**| **function**| **inout**| **default**| **individual** [*number*]| **message**| **error** [**software** [**informational**]| **call** [**informational**]]]

Syntax Description

all	(Optional) Enables all H.324 debugging except raw and raw decode.
default	(Optional) Activates function, inout, error call, and software debugging.
error	(Optional) Enables H.324 call error and software error debugging.
error [call]	(Optional) Enables H.324 major call processing error debugs related to the H.324 subsystem.
error [call [informational]]	(Optional) Enables H.324 major and informational call processing error debugs related to the H.324 subsystem.
error [software]	(Optional) Enables H.324 major software error debugs related to the H.324 subsystem.
error [software [informational]]	(Optional) Enables H.324 major and informational software error debugs related to the H.324 subsystem.
function	(Optional) Enables procedure tracing.
individual	(Optional) Activates individual H.324 debugging.
inout	(Optional) Enables subsystem inout debugging.
message	(Optional) Enables H.245 message display to/from H.324. Only displays message types, for message detail, use debug h245 asn1.
number	Index number. Number of debug category. See

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.4(22)T	This command was introduced.

Usage Guidelines

This command enables debugging for H.324 message events (video call control information).

Note This command is the same as the **debug voice h324**command**.**

Caution We recommend that you log the output from the **debug voip h324 all** command to a buffer, rather than sending the output to the console; otherwise, the size of the output could severely impact the performance of the gateway.

Use the **debug voip h324 individual** *index-number* command, where *index number is*a debug category, to activate a single debug.

This is helpful when trying to see a specific problem, without having a large number of debug output being generated. For example, the user could select the command **debug voip h324 individual 4** to see calls where no video caps arrived from the IP side of the call (SIP to H.324 direction). Multiple debug output can be activated using this command, one at a time. These are not additional debug output to the ones enabled by the command **debug voip h324 all**, just another way to selectively see specific information, without generating large amounts of debug output.

Table 63: Index Numbers and Descriptions for the debug voip h324 Command

Index Number	Description
1	Shows incoming H.245 message type
2	Shows MSD master/slave determination upon receiving MSD from peer
3	Warns that no audio caps were found from IP leg (not necessarily an error).
4	Warns that no video caps were found from IP leg (not necessarily an error).
5	Shows MSD master/slave determination when sending MSDack.
6	Displays media type being sent (audio/video), when sending MES message.
7	Displays H.223 parameters when sending TCS.

Index Number	Description
8	Displays OLC information, when sending audio OLC.
9	Displays OLC information, when sending video OLC.
10	Displays OLCack information, when sending OLCack.
11	Displays OLCrej information, when sending OLCrej.
12	Displays digit begin sent, when sending USER INPUT message.
13-15	Displays internal status bits of h245 messages sent/received in the h324 subsystem. No user data is provided.
16	Displays master/slave determination when MSDack is received.
17	Displays media type when MESack is received.
18	Displays media type when MESrej is received.
19	Displays OLC information, when receiving audio OLC.
20	Displays OLC information, when receiving video OLC.
21	Displays media type when OLCack is received.
22	Displays media type when OLCrej is received.
23	Displays message type, when an H.245 miscellaneous message is received (for example FastVideoUpdate).
24	Displays digit being received, when receiving USER INPUT message.
25	Displays message type, when an H.245 miscellaneous message is sent (for example FastVideoUpdate).
26	Displays outgoing message command type. No user data is provided with this debug.
27	Displays the initial H.223 mux level received from the peer, reported by the DSP.

Index Number	Description
28	Displays information about either OLCack or OLCrej being sent in response to an OLC request.
29	Displays the audio codec being opened with the IP leg.
30	Displays the video codec being opened with the IP leg. Should always be the same as the video codec with the H.324 leg.
31	Displays when Cisco IOS is sending the DSP either the H.223 multiplex table, or AL information. No user data is provided.
32	Indicates the digit being sent to the IP leg, through the RFC 2833 procedure.
33-34	Displays the parameters being sent to the DSP to configure either audio or video.
35	Displays information about the H.223 multiplex table being sent to the DSP.
36	Displays information about the H.223 AL configuration being sent to the DSP.
37-38	Indicates message arriving from IP leg. No user data is provided.
39	Displays information when receiving VENDOR ID message. This may show the type of equipment being connected to on the H.324 leg, if the peer adds the information to the message.
40	Displays the new H.223 multiplex level being configured.
41	Displays the new H.223 maximum PDU size being configured.
42	Indicates when the internal video capability memory has been released. No user data is provided.
43	Indicates when an empty capability set (ECS) has arrived from the IP leg of the call.
44	Indicates when a new capability set has arrived from the IP leg after an ECS has arrived.

Index Number	Description
45	Displays the dynamic payload number from the IP leg (H.324 to IP direction).

debug voip hpi

To enable debugging for Host Port Interface (HPI) message events, use the **debug voip hpi**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voip hpi [**all**| **command**| **default**| **detail**| **error** [**call [informational]**| **software [informational]**]| **function**| **inout**| **notification**| **response**| **stats**| **checker**]

no debug voip hpi

Syntax Description

all	(Optional) Displays all HPI debugging messages.
command	(Optional) Displays commands that are being sent to the 54x DSP.
default	(Optional) Displays HPI detail, error, and inout debugging messages and also runs if no keywords are added.
detail	(Optional) Displays detailed information about commands for the HPI. This command is used in conjunction with other **debug voip hpi** commands to show additional details when you use the **command**, **notification**, and **response** keywords.
error	(Optional) Displays HPI error messages.
call	(Optional) Displays call processing errors.
informational	(Optional) Displays minor and major errors. Without the **informational** keyword, only major errors are displayed.
software	(Optional) Displays software processing errors.
function	(Optional) Displays HPI functions.
inout	(Optional) Displays the output for the **command**, **notification**, **response**, and **stats** keywords.
notification	(Optional) Displays notification messages that are sent from the 54x DSP (for example, tone detection notification).
response	(Optional) Displays responses to commands that are sent by the 54x DSP (for example, responses to statistic requests).

stats	(Optional) Displays HPI statistics.
checker	(Optional) Displays HPI checker operations.

Command Default

Debugging is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(8)T	This command replaces the **debug hpi** command.
12.3(14)T	The **checker** keyword was added.

Usage Guidelines

This command enables debugging for HPI message events, which are used to communicate with digital signal processors (DSPs).

Use the **debug voip hpi all** command to view gateway DSP modem relay termination codes. The DSP-to-host messages for the modem relay termination indicate to the host the modem relay session termination time, physical or link layer, and other probable causes for disconnection. On receiving this indication from the DSP, the host can disconnect the call or place the channel in the modem passthrough state.

Examples

The following is sample output from the **debug voip hpi all** command for an incoming ISDN call:

```
Router# debug voip hpi all
```

```
01:28:44: //-1/xxxxxxxxxxxx/HPI/[]/hpi_dspmgr_open:
```

The following event shows that the HPI has identified the call, as shown by the GUID, but the call leg has not been specified, as shown by the -1 value in the CallEntry ID:

```
01:28:44: //-1/3FE022AC8009/HPI/[2/0:23]/hpi_dspmgr_open:
    Allocated DSP resource: dsp_intf=0x64AF0EEC hpi_cdb=0x64ACED34 ret=1
01:28:44: //-1/3FE022AC8009/HPI/[2/0:23]/hpi_dspmgr_open:
    Exit Line # 9411
01:28:44: //-1/3FE022AC8009/HPI/[2/0:23]/hpi_init:
01:28:44: //-1/3FE022AC8009/HPI/[2/0:23]/hpi_init:
    Open channel
01:28:44: //-1/3FE022AC8009/HPI/[2/0:23]/hpi_init:
    Packet details:
      Packet Length (16), Channel Id (1), Packet Id (74)
      ALawULawSelect=A Law Associated SignalingChannel (128)
      Timeslot=0 SerialPort=0
01:28:44: //-1/3FE022AC8009/HPI/[2/0:23]/caplog_hpi_msg_log:
01:28:44: //-1/3FE022AC8009/HPI/[2/0:23]/hpi_voice_config_params:
```

For each packet-related event, information about the packet is shown following the event. The following two events show the Real-Time Protocol (RTP) packet:

```
01:28:44: //-1/3FE022AC8009/HPI/[2/0:23]/hpi_encap_config:
    RTP information
```

```
01:28:44: //-1/3FE022AC8009/HPI/[2/0:23]/hpi_encap_config:
    Packet details:
      Packet Length (38), Channel Id (1), Packet Id (92) TransportProtocol=2
      t_ssrc=0x00 r_ssrc=0x00 t_vpxcc=0x0 r_vpxcc=0x0
      sid_support=0 tse_payload=101 seq_num=0x0 redundancy=0
      cc_payload_type=0 fax_payload_type=0 alaw_pcm_switchover=0
      mulaw_pcm_switchover=0 dtmf_payload_type=0 nte_rcv_payload_type=101
01:28:44: //-1/3FE022AC8009/HPI/[2/0:23]/caplog_hpi_msg_log:
01:28:44: //-1/3FE022AC8009/HPI/[2/0:23]/hpi_set_playout_config:
    Packet details:
      Packet Length (18), Channel Id (1), Packet Id (76)
      Mode=1, Initial=60, Min=40, Max=200, fax_nom=300
01:28:44: //-1/3FE022AC8009/HPI/[2/0:23]/caplog_hpi_msg_log:
01:28:44: //-1/3FE022AC8009/HPI/[2/0:23]/hpi_inband_sig:
01:28:44: //-1/3FE022AC8009/HPI/[2/0:23]/hpi_fax_enable:
01:28:44: //-1/3FE022AC8009/HPI/[2/0:23]/hpi_fax_enable:
    Enable FAX
01:28:44: //-1/3FE022AC8009/HPI/[2/0:23]/hpi_fax_enable:
    Packet details:
      Packet Length (8), Channel Id (1), Packet Id (67)
01:28:44: //-1/3FE022AC8009/HPI/[2/0:23]/caplog_hpi_msg_log:
01:28:44: //-1/3FE022AC8009/HPI/[2/0:23]/hpi_echo_cancel:
    Packet details:
      Packet Length (14), Channel Id (1), Packet Id (66)
      flags=0x00000B00, Threshold=-21, SuppressorCoverage=7
01:28:44: //-1/3FE022AC8009/HPI/[2/0:23]/caplog_hpi_msg_log:
01:28:44: //-1/3FE022AC8009/HPI/[2/0:23]/hpi_idle_code_det:
    Packet details:
      Packet Length (14), Channel Id (1), Packet Id (116)
      Enable (FALSE), Code=0x00000000, Duration (6000 ms)
01:28:44: //-1/3FE022AC8009/HPI/[2/0:23]/caplog_hpi_msg_log:
01:28:44: //-1/3FE022AC8009/HPI/[2/0:23]/hpi_set_gain:
    Packet details:
      Packet Length (12), Channel Id (1), Packet Id (91)
      Gain: In=0, Out=0
01:28:44: //-1/3FE022AC8009/HPI/[2/0:23]/caplog_hpi_msg_log:
01:28:44: //-1/3FE022AC8009/HPI/[2/0:23]/hpi_cng_config:
01:28:44: //-1/3FE022AC8009/HPI/[2/0:23]/hpi_info_field_size_config:
01:28:44: //-1/3FE022AC8009/HPI/[2/0:23]/hpi_digit_relay_config:
    Exit Line # 4162
01:28:44: //-1/3FE022AC8009/HPI/[2/0:23]/hpi_dspmgr_update_callid:
```

At this point, the HPI identifies the call leg, as shown by the CallEntry ID changing from -1 to 11.

```
01:28:44: //11/3FE022AC8009/HPI/[2/0:23:11]/hpi_voice_get_capabilities:
    Exit Line # 5073
01:28:44: //11/3FE022AC8009/HPI/[2/0:23:11]/hpi_stop_service:
01:28:44: //11/3FE022AC8009/HPI/[2/0:23:11]/hpi_idle_service:
    Packet details:
      Packet Length (8), Channel Id (1), Packet Id (68)
01:28:44: //11/3FE022AC8009/HPI/[2/0:23:11]/caplog_hpi_msg_log:
01:28:44: //11/3FE022AC8009/HPI/[2/0:23:11]/hpi_open_service:
    Setting codec g729r8
01:28:44: //11/3FE022AC8009/HPI/[2/0:23:11]/hpi_set_codec:
01:28:44: //11/3FE022AC8009/HPI/[2/0:23:11]/hpi_dsprm_callback:
01:28:44: //11/3FE022AC8009/HPI/[2/0:23:11]/hpi_voice_config_params:
01:28:44: //11/3FE022AC8009/HPI/[2/0:23:11]/hpi_digit_relay_config:
    Exit Line # 4162
```

The RTP packet is shown again, but now more information is available, such as payload types. The packet ID identifies this as the same RTP packet shown earlier.

```
01:28:44: //11/3FE022AC8009/HPI/[2/0:23:11]/hpi_encap_config:
    RTP information
01:28:44: //11/3FE022AC8009/HPI/[2/0:23:11]/hpi_encap_config:
    Packet details:
      Packet Length (38), Channel Id (1), Packet Id (92) TransportProtocol=2
      t_ssrc=0x040 r_ssrc=0x00 t_vpxcc=0x0 r_vpxcc=0x0
      sid_support=1 tse_payload=101 seq_num=0x13D3 redundancy=0
      cc_payload_type=125 fax_payload_type=122 alaw_pcm_switchover=8
      mulaw_pcm_switchover=0 dtmf_payload_type=121 nte_rcv_payload_type=101
01:28:44: //11/3FE022AC8009/HPI/[2/0:23:11]/caplog_hpi_msg_log:
```

```
01:28:44: //11/3FE022AC8009/HPI/[2/0:23:11]/hpi_pak_suppress:
    Stop packet suppression
01:28:44: //11/3FE022AC8009/HPI/[2/0:23:11]/hpi_pak_suppress:
    Packet details:
      Packet Length (10), Channel Id (1), Packet Id (106)
      Mode=1
01:28:44: //11/3FE022AC8009/HPI/[2/0:23:11]/caplog_hpi_msg_log:
01:28:44: //11/3FE022AC8009/HPI/[2/0:23:11]/hpi_vad_enable:
    Packet details:
      Packet Length (18), Channel Id (1), Packet Id (78)
      VAD=1 (ON): Threshold=-38, VADTime=250 Aggressive=0, Noise=-62
01:28:44: //11/3FE022AC8009/HPI/[2/0:23:11]/caplog_hpi_msg_log:
01:28:44: //11/3FE022AC8009/HPI/[2/0:23:11]/hpi_init:
    Open channel
01:28:44: //11/3FE022AC8009/HPI/[2/0:23:11]/hpi_init:
    Packet details:
      Packet Length (16), Channel Id (1), Packet Id (74)
      ALawULawSelect=A Law Associated SignalingChannel (128)
      Timeslot=0 SerialPort=0
01:28:44: //11/3FE022AC8009/HPI/[2/0:23:11]/caplog_hpi_msg_log:
01:28:44: //11/3FE022AC8009/HPI/[2/0:23:11]/hpi_voice_config_params:
01:28:44: //11/3FE022AC8009/HPI/[2/0:23:11]/hpi_encap_config:
    RTP information
01:28:44: //11/3FE022AC8009/HPI/[2/0:23:11]/hpi_encap_config:
    Packet details:
      Packet Length (38), Channel Id (1), Packet Id (92) TransportProtocol=2
      t_ssrc=0x040 r_ssrc=0x00 t_vpxcc=0x0 r_vpxcc=0x0
      sid_support=1 tse_payload=101 seq_num=0x13D3 redundancy=0
      cc_payload_type=125 fax_payload_type=122 alaw_pcm_switchover=8
      mulaw_pcm_switchover=0 dtmf_payload_type=121 nte_rcv_payload_type=101
01:28:44: //11/3FE022AC8009/HPI/[2/0:23:11]/caplog_hpi_msg_log:
01:28:44: //11/3FE022AC8009/HPI/[2/0:23:11]/hpi_set_playout_config:
    Packet details:
      Packet Length (18), Channel Id (1), Packet Id (76)
      Mode=1, Initial=60, Min=40, Max=200, fax_nom=300
01:28:44: //11/3FE022AC8009/HPI/[2/0:23:11]/caplog_hpi_msg_log:
01:28:44: //11/3FE022AC8009/HPI/[2/0:23:11]/hpi_inband_sig:
```

In the following several events, fax is enabled. Packets for echo cancellation, gain, voice activity detection (VAD), and other parameters appear.

```
01:28:44: //11/3FE022AC8009/HPI/[2/0:23:11]/hpi_fax_enable:
    Enable FAX
01:28:44: //11/3FE022AC8009/HPI/[2/0:23:11]/hpi_fax_enable:
    Packet details:
      Packet Length (8), Channel Id (1), Packet Id (67)
01:28:44: //11/3FE022AC8009/HPI/[2/0:23:11]/caplog_hpi_msg_log:
01:28:44: //11/3FE022AC8009/HPI/[2/0:23:11]/hpi_echo_cancel:
    Packet details:
      Packet Length (14), Channel Id (1), Packet Id (66)
      flags=0x00000B00, Threshold=-21, SuppressorCoverage=7
01:28:44: //11/3FE022AC8009/HPI/[2/0:23:11]/caplog_hpi_msg_log:
01:28:44: //11/3FE022AC8009/HPI/[2/0:23:11]/hpi_idle_code_det:
    Packet details:
      Packet Length (14), Channel Id (1), Packet Id (116)
      Enable (FALSE), Code=0x00000000, Duration (6000 ms)
01:28:44: //11/3FE022AC8009/HPI/[2/0:23:11]/caplog_hpi_msg_log:
01:28:44: //11/3FE022AC8009/HPI/[2/0:23:11]/hpi_set_gain:
    Packet details:
      Packet Length (12), Channel Id (1), Packet Id (91)
      Gain: In=0, Out=0
01:28:44: //11/3FE022AC8009/HPI/[2/0:23:11]/caplog_hpi_msg_log:
01:28:44: //11/3FE022AC8009/HPI/[2/0:23:11]/hpi_cng_config:
01:28:44: //11/3FE022AC8009/HPI/[2/0:23:11]/hpi_info_field_size_config:
01:28:44: //11/3FE022AC8009/HPI/[2/0:23:11]/hpi_digit_relay_config:
    Exit Line # 4162
01:28:44: //11/3FE022AC8009/HPI/[2/0:23:11]/hpi_start_service:
01:28:44: //11/3FE022AC8009/HPI/[2/0:23:11]/hpi_voice_mode:
    Packet details:
      Packet Length (28), Channel Id (1), Packet Id (73)
      CodingType=20, Voice FieldSize (20), VAD Flag (250),
      EchoLength=512, ComfortNoise=1, inband_detect=0x00000001,
      DigitRelay=2, AGC Flag=0, ECAN TestGroup=0,
      ECAN TestNumber=0, DynamicPayload=0
01:28:44: //11/3FE022AC8009/HPI/[2/0:23:11]/caplog_hpi_msg_log:
01:28:44: //11/3FE022AC8009/HPI/[2/0:23:11]/hpi_idle_code_det:
    Packet details:
      Packet Length (14), Channel Id (1), Packet Id (116)
      Enable (FALSE), Code=0x00000000, Duration (6000 ms)
01:28:44: //11/3FE022AC8009/HPI/[2/0:23:11]/caplog_hpi_msg_log:
01:28:44: //11/3FE022AC8009/HPI/[2/0:23:11]/hpi_start_service:
    Exit Line # 2816
01:28:44: //11/3FE022AC8009/HPI/[2/0:23:11]/hpi_send_data_to_dsp:
01:28:46: //11/3FE022AC8009/HPI/[2/0:23:11]/hpi_query_info:
```

Transmit and receive events are shown, along with packet information.

```
01:28:46: //11/3FE022AC8009/HPI/[2/0:23:11]/hpi_query_tx:
    Packet details:
      Packet Length (10), Channel Id (1), Packet Id (86)
      ResetFlag (0x00000000)
01:28:46: //11/3FE022AC8009/HPI/[2/0:23:11]/caplog_hpi_msg_log:
01:28:46: //11/3FE022AC8009/HPI/[2/0:23:11]/hpi_query_rx:
    Packet details:
      Packet Length (10), Channel Id (1), Packet Id (87)
      ResetFlag (0x00000000)
01:28:46: //11/3FE022AC8009/HPI/[2/0:23:11]/caplog_hpi_msg_log:
01:28:46: //11/3FE022AC8009/HPI/[2/0:23:11]/hpi_query_pd:
    Playout delay
01:28:46: //11/3FE022AC8009/HPI/[2/0:23:11]/hpi_query_pd:
    Packet details:
      Packet Length (10), Channel Id (1), Packet Id (83)
      ResetFlag (0x00000000)
01:28:46: //11/3FE022AC8009/HPI/[2/0:23:11]/caplog_hpi_msg_log:
01:28:46: //11/3FE022AC8009/HPI/[2/0:23:11]/hpi_query_pe:
    Playout error
01:28:46: //11/3FE022AC8009/HPI/[2/0:23:11]/hpi_query_pe:
    Packet details:
      Packet Length (10), Channel Id (1), Packet Id (84)
      ResetFlag (0x00000000)
01:28:46: //11/3FE022AC8009/HPI/[2/0:23:11]/caplog_hpi_msg_log:
01:28:46: //11/3FE022AC8009/HPI/[2/0:23:11]/hpi_query_info:
    Exit Line # 6578
01:28:46: //11/3FE022AC8009/HPI/[2/0:23:11]/hpi_receive_message:
01:28:46: //11/3FE022AC8009/HPI/[2/0:23:11]/caplog_hpi_msg_log:
```

Statistics for each of the events are displayed.

```
01:28:46: //11/3FE022AC8009/HPI/[2/0:23:11]/hpi_receive_query_tx:
    Got TX stats
    Packet details:
      Packet Length (36), Channel Id (1), Packet Id (199)
      TX Packets (87), Signaling Packets (0) ComfortNoise Packets (0)
      Transmit Duration (1750)
      Voice Transmit Duration (1750), FAX Transmit Duration (0)
01:28:46: //11/3FE022AC8009/HPI/[2/0:23:11]/hpi_receive_message:
01:28:46: //11/3FE022AC8009/HPI/[2/0:23:11]/caplog_hpi_msg_log:
01:28:46: //11/3FE022AC8009/HPI/[2/0:23:11]/hpi_receive_query_rx:
    Got RX stats
    Packet details:
      Packet Length (120), Channel Id (1), Packet Id (200)
      RX Packets (5): Signaling (0), ComfortNoise (1)
      Receive Duration (1750): Voice (70) FAX (0)
      Packet Counts: OOSequence (0), Bad header (0), Late (1), Early (0)
01:28:46: //11/3FE022AC8009/HPI/[2/0:23:11]/hpi_receive_message:
01:28:46: //11/3FE022AC8009/HPI/[2/0:23:11]/caplog_hpi_msg_log:
01:28:46: //11/3FE022AC8009/HPI/[2/0:23:11]/hpi_receive_query_pd:
    Got Playout Delay stats...
    Packet details:
      Packet Length (24), Channel Id (1), Packet Id (196)
      RX Delay: CurrentEstimate=69 Low WaterMark (69) High WaterMark (70)
      Clock Offset (-279863943)
01:28:46: //11/3FE022AC8009/HPI/[2/0:23:11]/hpi_receive_message:
01:28:46: //11/3FE022AC8009/HPI/[2/0:23:11]/caplog_hpi_msg_log:
01:28:46: //11/3FE022AC8009/HPI/[2/0:23:11]/hpi_receive_query_pe:
    Got Playout Error stats
    Packet details:
      Packet Length (32), Channel Id (1), Packet Id (197)
      Predictive Concealment Duration (0)
      Interpolative Concealment Duration (0)
      Silence Concealment Duration (0)
      Retroactive Memory Update (0)
      Buffer overflow discard duration (10)
      Talkspurt Detection Errors (0)
```

The following sample output from the **debug voip hpi checker** command helps verify the operations of the HPI checker:

```
Router# debug voip hpi checker
*May 19 06:30:53.532: hpi [] DSP [0/0x0] S_HPI_CLOSED(0) E_HPI_DSPRM_OPEN/SET_CODEC(54)
*May 19 06:30:53.532: hpi [] DSP [0/0x0] -> S_HPI_CLOSED(0)
*May 19 06:30:53.620: hpi [2/0:23 (22)] DSP [0/0x0] S_HPI_CLOSED(0)
E_HPI_DSP_OPEN_VOICE_CHANNEL(11)
*May 19 06:30:53.620: hpi [2/0:23 (22)] DSP [0/0x0] -> S_HPI_IDLE(1)
*May 19 06:30:53.620: hpi [2/0:23 (22)] DSP [0/0x0] S_HPI_IDLE(1)
E_HPI_DSPRM_OPEN/SET_CODEC(54)
*May 19 06:30:53.620: hpi [2/0:23 (22)] DSP [0/0x0] -> S_HPI_IDLE(1)
*May 19 06:30:53.624: hpi [2/0:23 (22)] DSP [0/0x0] S_HPI_IDLE(1) E_HPI_DSP_ENC_CONFIG(29)
*May 19 06:30:53.624: hpi [2/0:23 (22)] DSP [0/0x0] -> S_HPI_IDLE(1)
*May 19 06:30:53.624: hpi [2/0:23 (22)] DSP [0/0x0] S_HPI_IDLE(1)
E_HPI_DSP_SET_VOICE_PLAYOUT_DELAY(13)
*May 19 06:30:53.624: hpi [2/0:23 (22)] DSP [0/0x0] -> S_HPI_IDLE(1)
*May 19 06:30:53.624: hpi [2/0:23 (22)] DSP [0/0x0] S_HPI_IDLE(1) E_HPI_DSP_IDLE_CODE_CONTROL
 (50)
*May 19 06:30:53.624: hpi [2/0:23 (22)] DSP [0/0x0] -> S_HPI_IDLE(1)
*May 19 06:30:53.624: hpi [2/0:23 (22)] DSP [0/0x0] S_HPI_IDLE(1) E_HPI_DSP_VOICE_MODE(10)
*May 19 06:30:53.624: hpi [2/0:23 (22)] DSP [0/0x0] -> S_HPI_VOICE(2)
*May 19 06:30:53.624: hpi [2/0:23 (22)] DSP [0/0x0] S_HPI_VOICE(2)
E_HPI_DSP_GEN_PACKET_CONTROL(41)
*May 19 06:30:53.628: hpi [2/0:23 (22)] DSP [0/0x0] -> S_HPI_VOICE(2)
*May 19 06:30:53.628: hpi [2/0:23 (22)] DSP [0/0x0] S_HPI_VOICE(2)
E_HPI_DSP_ECHO_CANCELLER_CONTROL(3)
*May 19 06:30:53.628: hpi [2/0:23 (22)] DSP [0/0x0] -> S_HPI_VOICE(2)
*May 19 06:30:53.628: hpi [2/0:23 (22)] DSP [0/0x0] S_HPI_VOICE(2) E_HPI_DSP_SET_GAINS(28)
*May 19 06:30:53.628: hpi [2/0:23 (22)] DSP [0/0x0] -> S_HPI_VOICE(2)
*May 19 06:30:53.628: hpi [2/0:23 (22)] DSP [0/0x0] S_HPI_VOICE(2) E_HPI_DSP_DIGIT_RELAY(22)
*May 19 06:30:53.628: hpi [2/0:23 (22)] DSP [0/0x0] -> S_HPI_VOICE(2)
*May 19 06:30:53.628: hpi [2/0:23 (22)] DSP [0/0x0] S_HPI_VOICE(2) E_HPI_DSP_VAD_ENABLE(15)
*May 19 06:30:53.628: hpi [2/0:23 (22)] DSP [0/0x0] -> S_HPI_VOICE(2)
*May 19 06:30:53.632: hpi [2/0:23 (22)] DSP [0/0x0] S_HPI_VOICE(2)
E_HPI_DSP_GEN_PACKET_CONTROL(41)
*May 19 06:30:53.632: hpi [2/0:23 (22)] DSP [0/0x0] -> S_HPI_VOICE(2)
*May 19 06:30:53.632: hpi [2/0:23 (22)] DSP [0/0x0] S_HPI_VOICE(2)
E_HPI_DSP_INBAND_DETECTOR_CONTROL(45)
*May 19 06:30:53.632: hpi [2/0:23 (22)] DSP [0/0x0] -> S_HPI_VOICE(2)
*May 19 06:30:53.636: hpi [2/0:23 (22)] DSP [0/0x0] S_HPI_VOICE(2)
E_HPI_DSP_INBAND_DETECTOR_CONTROL(45)
*May 19 06:30:53.636: hpi [2/0:23 (22)] DSP [0/0x0] -> S_HPI_VOICE(2)
*May 19 06:30:53.624: hpi [2/0:23 (22)] DSP [0/0x0] S_HPI_IDLE(1) E_HPI_DSP_VOICE_MODE(10)
*May 19 06:30:53.624: hpi [2/0:23 (22)] DSP [0/0x0] -> S_HPI_VOICE(2)
*May 19 06:30:53.624: hpi [2/0:23 (22)] DSP [0/0x0] S_HPI_VOICE(2)
E_HPI_DSP_GEN_PACKET_CONTROL(41)
*May 19 06:30:53.628: hpi [2/0:23 (22)] DSP [0/0x0] -> S_HPI_VOICE(2)
*May 19 06:30:53.628: hpi [2/0:23 (22)] DSP [0/0x0] S_HPI_VOICE(2)
E_HPI_DSP_ECHO_CANCELLER_CONTROL(3)
*May 19 06:30:53.628: hpi [2/0:23 (22)] DSP [0/0x0] -> S_HPI_VOICE(2)
*May 19 06:30:53.628: hpi [2/0:23 (22)] DSP [0/0x0] S_HPI_VOICE(2) E_HPI_DSP_SET_GAINS(28)
*May 19 06:30:53.628: hpi [2/0:23 (22)] DSP [0/0x0] -> S_HPI_VOICE(2)
*May 19 06:30:53.628: hpi [2/0:23 (22)] DSP [0/0x0] S_HPI_VOICE(2) E_HPI_DSP_DIGIT_RELAY(22)
*May 19 06:30:53.628: hpi [2/0:23 (22)] DSP [0/0x0] -> S_HPI_VOICE(2)
*May 19 06:30:53.628: hpi [2/0:23 (22)] DSP [0/0x0] S_HPI_VOICE(2) E_HPI_DSP_VAD_ENABLE(15)
*May 19 06:30:53.628: hpi [2/0:23 (22)] DSP [0/0x0] -> S_HPI_VOICE(2)
*May 19 06:30:53.632: hpi [2/0:23 (22)] DSP [0/0x0] S_HPI_VOICE(2)
E_HPI_DSP_GEN_PACKET_CONTROL(41)
*May 19 06:30:53.632: hpi [2/0:23 (22)] DSP [0/0x0] -> S_HPI_VOICE(2)
*May 19 06:30:53.632: hpi [2/0:23 (22)] DSP [0/0x0] S_HPI_VOICE(2)
E_HPI_DSP_INBAND_DETECTOR_CONTROL(45)
```

```
*May 19 06:30:53.632: hpi [2/0:23 (22)] DSP [0/0x0] -> S_HPI_VOICE(2)
*May 19 06:30:53.636: hpi [2/0:23 (22)] DSP [0/0x0] S_HPI_VOICE(2)
E_HPI_DSP_INBAND_DETECTOR_CONTROL(45)
*May 19 06:30:53.636: hpi [2/0:23 (22)] DSP [0/0x0] -> S_HPI_VOICE(2)
*May 19 06:30:56.512: hpi [2/0:23 (22)] DSP [0/0x0] S_HPI_VOICE(2) E_HPI_DSP_GET_TX_STAT(23)
*May 19 06:30:56.512: hpi [2/0:23 (22)] DSP [0/0x0] -> S_HPI_VOICE(2)
*May 19 06:30:56.512: hpi [2/0:23 (22)] DSP [0/0x0] S_HPI_VOICE(2) E_HPI_DSP_GET_RX_STAT(24)
*May 19 06:30:56.512: hpi [2/0:23 (22)] DSP [0/0x0] -> S_HPI_VOICE(2)
*May 19 06:30:56.512: hpi [2/0:23 (22)] DSP [0/0x0] S_HPI_VOICE(2)
E_HPI_DSP_GET_VOICE_PLAYOUT_DELAY(20)
*May 19 06:30:56.512: hpi [2/0:23 (22)] DSP [0/0x0] -> S_HPI_VOICE(2)
*May 19 06:30:56.512: hpi [2/0:23 (22)] DSP [0/0x0] S_HPI_VOICE(2)
E_HPI_DSP_GET_VOICE_PLAYOUT_ERROR(21)
*May 19 06:30:56.512: hpi [2/0:23 (22)] DSP [0/0x0] -> S_HPI_VOICE(2)
*May 19 06:30:59.384: hpi [2/0:23 (22)] DSP [0/0x0] S_HPI_VOICE(2) E_HPI_DSP_GET_TX_STAT(23)
*May 19 06:30:59.384: hpi [2/0:23 (22)] DSP [0/0x0] -> S_HPI_VOICE(2)
*May 19 06:30:59.384: hpi [2/0:23 (22)] DSP [0/0x0] S_HPI_VOICE(2) E_HPI_DSP_GET_RX_STAT(24)
*May 19 06:30:59.384: hpi [2/0:23 (22)] DSP [0/0x0] -> S_HPI_VOICE(2)
*May 19 06:30:59.384: hpi [2/0:23 (22)] DSP [0/0x0] S_HPI_VOICE(2)
E_HPI_DSP_GET_VOICE_PLAYOUT_DELAY(20)
*May 19 06:30:59.384: hpi [2/0:23 (22)] DSP [0/0x0] -> S_HPI_VOICE(2)
*May 19 06:30:59.384: hpi [2/0:23 (22)] DSP [0/0x0] S_HPI_VOICE(2)
E_HPI_DSP_GET_VOICE_PLAYOUT_ERROR(21)
*May 19 06:30:59.384: hpi [2/0:23 (22)] DSP [0/0x0] -> S_HPI_VOICE(2)
*May 19 06:31:06.524: hpi [2/0:23 (22)] DSP [0/0x0] S_HPI_VOICE(2)
E_HPI_DSP_GEN_PACKET_CONTROL(41)
*May 19 06:31:06.524: hpi [2/0:23 (22)] DSP [0/0x0] -> S_HPI_VOICE(2)
*May 19 06:31:06.532: hpi [2/0:23 (22)] DSP [0/0x0] S_HPI_VOICE(2) E_HPI_DSP_GET_LEVELS(26)
*May 19 06:31:06.536: hpi [2/0:23 (22)] DSP [0/0x0] -> S_HPI_VOICE(2)
*May 19 06:31:06.536: hpi [2/0:23 (22)] DSP [0/0x0] S_HPI_VOICE(2) E_HPI_DSP_GET_ERROR_STAT(0)
*May 19 06:31:06.536: hpi [2/0:23 (22)] DSP [0/0x0] -> S_HPI_VOICE(2)
*May 19 06:31:06.572: hpi [2/0:23 (22)] DSP [0/0x0] S_HPI_VOICE(2) E_HPI_DSP_IDLE_MODE(5)
*May 19 06:31:06.572: hpi [2/0:23 (22)] DSP [0/0x0] -> S_HPI_IDLE(1)
*May 19 06:31:06.572: hpi [2/0:23 (22)] DSP [0/0x0] S_HPI_IDLE(1)
E_HPI_DSP_CLOSE_VOICE_CHANNEL(12)
*May 19 06:31:06.572: hpi [2/0:23 (22)] DSP [0/0x0] -> S_HPI_CLOSED(0)
*May 19 06:31:06.576: hpi [2/0:23 (22)] DSP [0/0x0] -> S_HPI_CLOSED(0)
*May 19 06:31:06.576: hpi [2/0:23 (22)] DSP [0/0x0] -> S_HPI_CLOSED(0)
```

Related Commands

Command	Description
show voice hpi capture	Verifies capture status and statistics.

debug voip ipipgw

To turn on debugging for the Cisco Multiservice IP-to-IP Gateway (IPIPGW), use the **debug voip ipipgw** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voip ipipgw

no debug voip ipipgw

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(13)T3	This command was introduced.

Examples

The following example shows debugging output from a Cisco IPIPGW:

```
Aug  8 15:24:30.626 EDT: cch323_build_early_fastStart_cap_response:
ccb-remote_fastStart=0x63C20630
Aug  8 15:24:30.626 EDT:
cch323_build_early_fastStart_cap_response:symm_mask=1,tempOtherCodec=5,templocalCodec=5,audioFastStartArray=0x63C1299C
The following line shows fast start response beginning for the inbound leg of the IP-to-IP
 call:
Aug  8 15:24:30.626 EDT: cch323_build_early_fastStart_cap_response: Assuming ipipgw on
inbound leg.
Aug  8 15:24:30.626 EDT: Function: cch323_build_olc_for_ccapi, Line: 1198,
audioFastStartArray=0x63C1299C
Aug  8 15:24:30.626 EDT: cch323_build_olc_for_ccapi: channel_info ptr=0x63C203F0, ccb
ptr=0x63C18580
The following lines indicate the inbound call leg will send the channel information to the
 outbound call leg (not yet created):
Aug  8 15:24:30.626 EDT: cch323_build_olc_for_ccapi: Channel Information:
        Logical Channel Number (fwd/rev): 1
        Channel address (fwd/rev):        0x10C0C27
        RTP  Channel (fwd/rev):           19362
        RTCP Channel (fwd/rev):           19363
        QoS Capability (fwd/rev):         0
        Symmetric Audio Cap Mask:         0x1
        Symmetric Audio Codec Bytes:      160
        Flow Mode:                        0
        Silence Suppression:              0
Aug  8 15:24:30.626 EDT: cch323_build_olc_for_ccapi:NumOfElements = 1 idx = 1
```

The following line indicates the inbound call leg is set to work in IP-to-IP mode (0xF0):

```
Aug  8 15:24:30.630 EDT: cch323_set_h245_state_mc_mode_incoming: h245 state m/c mode=0xF0
```

The following line indicates flow mode for incoming call leg is set to FLOW_THROUGH (incoming callid = 35). At this point Session Application opens the outbound leg. Some output is omitted here.

```
Aug  8 15:24:30.630 EDT: cch323_media_flow_mode: IPIPGW(35):Flow Mode=1
Aug  8 15:24:30.630 EDT: cch323_set_h245_state_mc_mode_outgoing:call_spi_mode = 1
```

The following line indicates the outbound call leg is set to work in IP-to-IP mode (0xF0):

```
Aug  8 15:24:30.630 EDT: cch323_set_h245_state_mc_mode_outgoing: h245 state m/c mode=0xF0
Aug  8 15:24:30.630 EDT: cch323_get_peer_info line 1022:
Aug  8 15:24:30.630 EDT: cch323_get_peer_info line 1026:
Aug  8 15:24:30.630 EDT: cch323_set_pref_codec_list:IPIPGW(36):peer channel present: dp
pref mask=FFFFFFFF
Aug  8 15:24:30.630 EDT: cch323_set_pref_codec_list:IPIPGW(36):first preferred
codec(bytes)=5(160)
```

The following line indicates the outbound call leg is set to FLOW_THROUGH (outbound callid = 36):

```
Aug  8 15:24:30.630 EDT: cch323_get_peer_info: Flow Mode set to FLOW_THROUGH for callId 36
Aug  8 15:24:30.642 EDT: cch323_build_local_encoded_fastStartOLCs: state_mc_mode=0xF0 on
outbound leg.
Aug  8 15:24:30.642 EDT: cch323_build_local_encoded_fastStartOLCs:srcAddress = 0x10C0C30,
h245_lport = 0, flow mode = 1, minimum_qos=0
Aug  8 15:24:30.642 EDT: cch323_generic_open_logical_channel: IPIPGW: current codec =
5:160:160.
```

The following line indicates the IPIPGW received fast start response from the remote (called party) entity of the outbound call leg:

```
Aug  8 15:24:30.658 EDT: Function: cch323_receive_fastStart_cap_response Line: 2800
Aug  8 15:24:30.658 EDT: Function: cch323_build_olc_for_ccapi, Line: 1198,
audioFastStartArray=0x63C1259C
Aug  8 15:24:30.658 EDT: cch323_build_olc_for_ccapi: channel_info ptr=0x63C12738, ccb
ptr=0x631A4D68
Aug  8 15:24:30.658 EDT: cch323_build_olc_for_ccapi: Channel Information:
        Logical Channel Number (fwd/rev): 1
        Channel address (fwd/rev):        0x10C0C28
        RTP  Channel (fwd/rev):           19128
        RTCP Channel (fwd/rev):           19129
        QoS Capability (fwd/rev):         0
        Symmetric Audio Cap Mask:         0x1
        Symmetric Audio Codec Bytes:      160
        Flow Mode:                        0
        Silence Suppression:              0
Aug  8 15:24:30.658 EDT: cch323_build_olc_for_ccapi:NumOfElements = 1 idx = 1
Aug  8 15:24:30.658 EDT: Function: cch323_do_open_channel_ind Line: 1080
Aug  8 15:24:30.658 EDT: Function: cch323_open_channel_ind Line: 1132
```

The following lines indicates the outbound call leg (36) sends the channel response back to the inbound call leg (35) via CCAPI:

```
Aug  8 15:24:30.658 EDT: cch323_receive_fastStart_cap_response: callID 0x24(36),
audioFastStartArray = 0x0.
Aug  8 15:24:30.658 EDT: cch323_peer_channel_ind: IPIPGW:### chn info coming in chn_ind()
Aug  8 15:24:30.658 EDT: cch323_peer_channel_ind: IPIPGW(35):giving event to Fast start
logic.
Aug  8 15:24:30.658 EDT: Function: cch323_do_open_channel Line: 5557
Aug  8 15:24:30.658 EDT: cch323_do_open_channel: line:5566, ccb->status=0x4000000
Aug  8 15:24:30.658 EDT: cch323_do_open_channel:srcAddress = 0x10C0C30, h245_lport = 18308,
 minimum_qos=0
Aug  8 15:24:30.658 EDT: cch323_build_fastStart_cap_response: Start...
Aug  8 15:24:30.658 EDT: cch323_build_fastStart_cap_response: selectCodec=5, codec_mask=1,
 configured_codecBytes=160
        forward_codecBytes=160, reverse_codecBytes=160, audioFastStartArray=0x63C1299C
Aug  8 15:24:30.658 EDT: cch323_prepare_fastStart_cap_response line 2138
Aug  8 15:24:30.658 EDT: cch323_prepare_fastStart_cap_response: callID 0x23(35),
audioFastStartArray = 0x0.
```

```
Aug  8 15:24:30.658 EDT: cch323_prepare_fastStart_cap_response,
ccb->local_fastStart=0x63C183C0, srcAddr=0x10C0C30, lport=18308, rport=19362, rc=1
Aug  8 15:24:30.658 EDT: cch323_build_fastStart_cap_response: local_fastStart=0x63C183C0,
negotiated_codec=5, negotiated_codec_bytes=160
Aug  8 15:24:30.658 EDT: cch323_build_fastStart_cap_response: Received peer cap info. Notify
 RAS state machine (possible BRQ).
```

Outbound leg, at this point, has prepared the fast start response to be sent to the originating (calling party). This is sent in the next outgoing call control message (such as ALERT or PROGRESS):

```
Aug  8 15:24:30.658 EDT: cch323_build_fastStart_cap_response: Done.
Aug  8 15:24:30.658 EDT: cch323_do_open_channel: line:5644, ccb->status=0x4004200
Aug  8 15:24:30.674 EDT: cch323_h245_connection_sm: state = 0 event=5 ccb=63C18580
Aug  8 15:24:30.674 EDT: cch323_h245_connection_sm: listen state=0
Aug  8 15:24:30.678 EDT: cch323_h245_cap_ind: IPIPGW(35): masks au=0x1 data=0xC uinp=0x32.
```

The following line indicates the inbound call leg (35) received capability set (CAPSET) message**:**

```
Aug  8 15:24:30.678 EDT: cch323_run_h245_cap_in_sm:IPIPGW(35): got incoming CAPSET msg.
Aug  8 15:24:30.678 EDT: cch323_do_transparent_cap_ind: IPIPGW(35):mask sent to other leg=1
```

The following lines show the inbound call leg (35) forwarding the TCS to the outbound leg and waiting for the response of the outbound call leg (TCSACK or TCSREJ):

```
Aug  8 15:24:30.678 EDT: cch323_run_h245_cap_in_sm:IPIPGW(35):suppressTCS: our TCS will be
 sent based on peer.
Aug  8 15:24:30.678 EDT: cch323_h245_cap_notify:IPIPGW(35):not xmiting CAPSACK: wait for
peer to ack.
Aug  8 15:24:30.678 EDT: cch323_caps_ind: IPIPGW(36):setting the mask to new : current
mask=0x4FFFF new mask=0x1.
Aug  8 15:24:30.678 EDT: cch323_caps_ind: IPIPGW(36): ExtendedCapsPresent
Aug  8 15:24:30.678 EDT: cch323_set_dtmf_relay_mask: IPIPGW(36): extract dtmf-caps from
caps struct
Aug  8 15:24:30.678 EDT: cch323_set_dtmf_relay_mask: IPIPGW(36): After extracting dtmf-caps
 from caps structccb->user_caps.user_input_bit_mask[0x1C]
```

The following line shows the outbound leg sending the TCS to the called party. No codec filter is configured on outbound dial-peer (FFFFFFFF):

```
Aug  8 15:24:30.678 EDT: cch323_prepare_preferred_codec_list: IPIPGW(36):munging
caps:5:1:FFFFFFFF
Aug  8 15:24:30.678 EDT: cch323_prepare_preferred_codec_list: IPIPGW(36):final mask=1
Aug  8 15:24:30.678 EDT: cch323_peer_caps_ind_common:IPIPGW(36):starting 245 via tunnel
Aug  8 15:24:30.678 EDT: cch323_h245_connection_sm: state = 0 event=1 ccb=631A4D68
Aug  8 15:24:30.678 EDT: cch323_h245_connection_sm: listen state=0
Aug  8 15:24:30.678 EDT: cch323_h245_start_cap_sm: IPIPGW(36): starting the cap/msd machine.
Aug  8 15:24:30.678 EDT: cch323_send_generic_caps: IPIPGW: audiomask raw =0x1.
Aug  8 15:24:30.678 EDT: cch323_set_pref_codec_list:IPIPGW(36):first preferred
codec(bytes)=21(0)
```

The following line shows the outbound leg forwarding the TCS over H.225 tunnel (starting H.245 via tunnel):

```
Aug  8 15:24:30.678 EDT: cch323_send_generic_caps: IPIPGW:[trans]audio mask after
operation=0x1.
```

The following lines show master-slave determination events passing from inbound to outbound and vice versa:

```
Aug  8 15:24:30.678 EDT: cch323_run_passthru_msd: IPIPGW(36):event = H245_EVENT_MSD
Aug  8 15:24:30.678 EDT: cch323_h245_connection_sm: state = 0 event=5 ccb=63C18580
Aug  8 15:24:30.678 EDT: cch323_h245_connection_sm: listen state=0
Aug  8 15:24:30.678 EDT: cch323_run_passthru_msd: IPIPGW(35):event = H245_EVENT_MS_IND
Aug  8 15:24:30.678 EDT: cch323_h245_connection_sm: state = 2 event=5 ccb=631A4D68
Aug  8 15:24:30.678 EDT: cch323_h245_connection_sm: listen state=0
Aug  8 15:24:30.678 EDT: cch323_h245_cap_ind: IPIPGW(36): masks au=0x1 data=0xC uinp=0x32.
Aug  8 15:24:30.678 EDT: cch323_run_h245_cap_in_sm:IPIPGW(36): got incoming CAPSET msg.
Aug  8 15:24:30.678 EDT: cch323_do_transparent_cap_ind: IPIPGW(36):mask sent to other leg=1
```

The following lines show the outbound leg forwarding the TCS to the other leg and waiting for its response (TCSACK or TCSREJ):

```
Aug  8 15:24:30.678 EDT: cch323_run_h245_cap_in_sm:IPIPGW(36):suppressTCS: our TCS will be
 sent based on peer.
Aug  8 15:24:30.678 EDT: cch323_h245_cap_notify:IPIPGW(36):not xmiting CAPSACK: wait for
peer to ack.
Aug  8 15:24:30.678 EDT: cch323_run_passthru_msd: IPIPGW(36):event = H245_EVENT_MSD
Aug  8 15:24:30.678 EDT: cch323_caps_ind: IPIPGW(35):setting the mask to new : current
mask=0x4FFFF new mask=0x1.
Aug  8 15:24:30.682 EDT: cch323_caps_ind: IPIPGW(35): ExtendedCapsPresent
Aug  8 15:24:30.682 EDT: cch323_set_dtmf_relay_mask: IPIPGW(35): extract dtmf-caps from
caps struct
Aug  8 15:24:30.682 EDT: cch323_set_dtmf_relay_mask: IPIPGW(35): After extracting dtmf-caps
 from caps structccb->user_caps.user_input_bit_mask[0x1C]
Aug  8 15:24:30.682 EDT: cch323_prepare_preferred_codec_list: IPIPGW(35):munging
caps:21:1:FFFFFFFF
Aug  8 15:24:30.682 EDT: cch323_prepare_preferred_codec_list: IPIPGW(35):final mask=1
Aug  8 15:24:30.682 EDT: cch323_peer_caps_ind_common:IPIPGW(35):starting 245 via tunnel
Aug  8 15:24:30.682 EDT: cch323_h245_connection_sm: state = 0 event=1 ccb=63C18580
Aug  8 15:24:30.682 EDT: cch323_h245_connection_sm: listen state=0
Aug  8 15:24:30.682 EDT: cch323_h245_start_cap_sm: IPIPGW(35): starting the cap/msd machine.
Aug  8 15:24:30.682 EDT: cch323_send_generic_caps: IPIPGW: audiomask raw =0x1.
Aug  8 15:24:30.682 EDT: cch323_set_pref_codec_list:IPIPGW(35):first preferred
codec(bytes)=21(0)
```

The following line shows the inbound call leg sending the TCS to the calling party:

```
Aug  8 15:24:30.682 EDT: cch323_send_generic_caps: IPIPGW:[trans]audio mask after
operation=0x1.
Aug  8 15:24:30.682 EDT: cch323_run_passthru_msd: IPIPGW(35):event = H245_EVENT_MSD
Aug  8 15:24:30.682 EDT: cch323_h245_connection_sm: state = 2 event=5 ccb=631A4D68
Aug  8 15:24:30.682 EDT: cch323_h245_connection_sm: listen state=0
Aug  8 15:24:30.682 EDT: cch323_run_passthru_msd: IPIPGW(36):event = H245_EVENT_MS_IND
Aug  8 15:24:30.682 EDT: cch323_h245_connection_sm: state = 2 event=5 ccb=631A4D68
Aug  8 15:24:30.682 EDT: cch323_h245_connection_sm: listen state=0
Aug  8 15:24:30.682 EDT: cch323_run_h245_cap_out_sm: IPIPGW(36): got caps ack.
Aug  8 15:24:30.682 EDT: cch323_run_h245_cap_out_sm:IPIPGW(36): sending caps ack to other
leg.
Aug  8 15:24:30.682 EDT: Function: cch323_do_caps_ack Line: 1116
Aug  8 15:24:30.682 EDT: cch323_run_passthru_msd: IPIPGW(35):event = H245_EVENT_MSD
Aug  8 15:24:30.682 EDT: cch323_peer_caps_ack: IPIPGW(35):sending caps resp event to CAP
state mc.
Aug  8 15:24:30.682 EDT: cch323_h245_connection_sm: state = 2 event=5 ccb=631A4D68
Aug  8 15:24:30.682 EDT: cch323_h245_connection_sm: listen state=0
Aug  8 15:24:30.682 EDT: cch323_run_passthru_msd: IPIPGW(36):event = H245_EVENT_MS_CFM
Aug  8 15:24:30.682 EDT: cch323_run_passthru_msd: IPIPGW(35):event = H245_EVENT_MS_DET_RSP
Aug  8 15:24:30.686 EDT: cch323_h245_connection_sm: state = 2 event=5 ccb=63C18580
Aug  8 15:24:30.686 EDT: cch323_h245_connection_sm: listen state=0
Aug  8 15:24:30.686 EDT: cch323_run_h245_cap_out_sm: IPIPGW(35): got caps ack.
```

The following line shows the inbound leg informing the outbound leg of the TCSACK**:**

```
Aug  8 15:24:30.686 EDT: cch323_run_h245_cap_out_sm:IPIPGW(35): sending caps ack to other
leg.
Aug  8 15:24:30.686 EDT: Function: cch323_do_caps_ack Line: 1116
Aug  8 15:24:30.686 EDT: cch323_peer_caps_ack: IPIPGW(36):sending caps resp event to CAP
state mc.
Aug  8 15:24:30.686 EDT: cch323_h245_connection_sm: state = 2 event=5 ccb=63C18580
Aug  8 15:24:30.686 EDT: cch323_h245_connection_sm: listen state=0
```

The following lines show that master-slave determination procedures are completed on both call legs:

```
Aug  8 15:24:30.686 EDT: cch323_run_passthru_msd: IPIPGW(35):event = H245_EVENT_MS_CFM
Aug  8 15:24:30.686 EDT: cch323_run_passthru_msd: IPIPGW(36):event = H245_EVENT_MS_DET_RSP
```

debug voip ivr

Note The **debug voip ivr**command is replaced by the **debug voip application**command. See the **debug voip application**command for more information.

debug voip ivr all

Note The **debug voip ivr all**command is replaced by the **debug voip application all**command. See the **debug voip application all**command for more information.

debug voip ivr applib

Note The **debug voip ivr applib** command is replaced by the **debug voip application core** command. See the **debug voip application core** command for more information.

debug voip ivr callsetup

Note The **debug voip ivr callset**command is replaced by the **debug voip application callset**command. See the **debug voip application callset**command for more information.

debug voip ivr digitcollect

Note The **debug voip ivr digitcollect**command is replaced by the **debug voip application digitcollect**command. See the **debug voip application digitcollect** command for more information.

debug voip ivr dynamic

Note The **debug voip ivr dynamic** command is replaced by the **debug voip application media state** command. See the **debug voip application media state** command for more information.

debug voip ivr error

Note

The **debug voip ivr error**command is replaced by the **debug voip application error**command. See the **debug voip application error**command for more information.

debug voip ivr redirect

Note The **debug voip ivr redirect**command is replaced by the **debug voip application redirect**command. See the **debug voip application redirect**command for more information.

debug voip ivr script

Note The **debug voip ivr script** command is replaced by the **debug voip application script** command. See the **debug voip application script** command for more information.

debug voip ivr settlement

Note

The **debug voip ivr settlement**command is replaced by the **debug voip application settlement** command. See the **debug voip application settlement**command for more information.

debug voip ivr states

Note The **debug voip ivr states**command is replaced by the **debug voip application media state**command. See the **debug voip application media state** command for more information.

debug voip ivr supplementary-service

Note The **debug voip ivr supplementary-service**command is replaced by the **debug voip application supplementary-service**command. See the **debug voip application supplementary-service** command for more information.

debug voip ivr tclcommands

Note The **debug voip ivr tclcommands**command is replaced by the **debug voip application tclcommands**command. See the **debug voip application tclcommands**command for more information.

debug voip lpcor

To display debugging information for the logical partitioning class of restriction (LPCOR) feature, use the **debug voip lpcor**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voip lpcor [**all**| **default**| **detail**| **error** [**call**| **software**]| **function**| **inout**]

no debug voip lpcor [**all**| **default**| **detail**| **error** [**call**| **software**]| **function**| **inout**]

Syntax Description

all	(Optional) Enables all LPCOR debugging.
default	(Optional) Enables error, function, and inout debugging. This is the default option if no keywords are used.
detail	(Optional) Enables detailed trace messages of the LPCOR subsystem.
error	(Optional) Enables LPCOR major call and software error debugging.
call	(Optional) Enables major call error debugging.
software	(Optional) Enables major software error debugging.
function	(Optional) Enables tracing of the functions called by the LPCOR subsystem.
inout	(Optional) Enables function in and out debugging.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
15.0(1)XA	This command was introduced.
15.1(1)T	This command was integrated into Cisco IOS Release 15.1(1)T.

Usage Guidelines

This command enables debugging for LPCOR events.

Examples

The following is sample output from the **debug voip lpcor**command for a call between two phones that was blocked by LPCOR policy validation:

```
*Jun 25 09:52:18.910: //-1/xxxxxxxxxxxx/LPCOR/lpcor_get_index_by_name:
   lpcor local_sccp_phone_1
*Jun 25 09:52:18.910: //-1/xxxxxxxxxxxx/LPCOR/lpcor_get_index_by_name:
   lpcor local_sccp_phone_1 index 1
*Jun 25 09:52:20.114: //-1/xxxxxxxxxxxx/LPCOR/lpcor_get_index_by_peer:
   peer tag 20003, direction 1
*Jun 25 09:52:20.114: //-1/xxxxxxxxxxxx/LPCOR/lpcor_get_index_by_name:
   lpcor local_sccp_phone_2
*Jun 25 09:52:20.114: //-1/xxxxxxxxxxxx/LPCOR/lpcor_get_index_by_name:
   lpcor local_sccp_phone_2 index 2
*Jun 25 09:52:20.114: //-1/xxxxxxxxxxxx/LPCOR/lpcor_get_index_by_peer:
   Return Lpcor Index 2 for Peer Tag 20003 *Jun 25 09:52:20.114:
//-1/xxxxxxxxxxxx/LPCOR/lpcor_index_is_valid:
   lpcor index 1 is valid
*Jun 25 09:52:20.114: //-1/xxxxxxxxxxxx/LPCOR/lpcor_policy_validate:
   Source LPCOR Index=1, Target LPCOR Policy=local_sccp_phone_2  -Traceback= 0x42949584
0x4219C430 0x4219CCDC 0x421A6B60 0x421A6D1C 0x421A75EC 0x421AB328 0x421ACA14 0x421B2518
0x421B2FFC 0x421B7614 0x4217F910 0x421F522C 0x421F89CC 0x421D84FC 0x422033BC *Jun 25
09:52:20.118: //-1/xxxxxxxxxxxx/LPCOR/lpcor_policy_validate:
   Validate Fail; lpcor (source[1] target[2])
```

Related Commands

Command	Description
debug ephone lpcor	Displays debugging information for calls using the LPCOR feature.
debug voip application lpcor	Enables debugging of the LPCOR application system.
show voice lpcor policy	Displays the LPCOR policy for the specified resource group.
voice lpcor enable	Enables LPCOR functionality on the Cisco Unified CME router.
voice lpcor policy	Creates a LPCOR policy for a resource group.

debug voip profile fax

To enable a set of debug commands for fax applications, use the **debug voip profile fax** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voip profile fax [**mail**| **relay** [**application**| **signaling**]]

no debug voip profile fax

Syntax Description

mail	Enables the following set of debugs for an onramp or offramp fax mail call: • **debug csm voice** • **debug fax dmsp all** • **debug fax fmsp all** • **debug fax foip all** • **debug fax mmoip aaa all** • **debug fax mspi all** • **debug fax mta all** • **debug isdn q931** • **debug voip application all** • **debug voip application vxml all** • **debug voip ccapi all** • **debug voip dsm all** • **debug voip dspapi all** • **debug voip hpi all** • **debug voip ivr all** • **debug voip vtsp all**

	The following debug commands are enabled for access servers with MICA modem cards: • **debug fax fmsp all** • **debug fax mmoip aaa** • **debug fax mta all** • **debug isdn q931** • **debug voip application all** • **debug voip application vxml all** • **debug voip ccapi all** • **debug voip dsm all** • **debug voip dspapi all** • **debug voip hpi all** • **debug voip ivr all** • **debug voip vtsp all** The following debug options are enabled for access servers with universal port dial feature cards: • **debug fax dmsp all** • **debug fax fmsp all** • **debug fax foip all** • **debug fax mspi all** • **debug voip application vxml all** • **debug voip ivr all**
relay	Enables the **debug fax relay t30 all-level-1** and the sets specified by either the **application** or **signaling** keyword.
application	Enables the following set of debugs for fax relay applications: • **debug voip application all** • **debug voip application vxml all** • **debug voip ccapi all** • **debug voip dialpeer all** • **debug voip ivr all**

signaling	Enables the following set of debugs for fax relay signaling: • **debug cch323 all** • **debug ccsip error** • **debug ccsip messages** • **debug cdapi detail** • **debug cdapi events** • **debug csm voice** • **debug gtd error** • **debug gtd events** • **debug h225 asn1** • **debug h225 events** • **debug h225 q931** • **debug h245 events** • **debug h245 asn1** • **debug isdn q931** • **debug mgcp errors** • **debug mgcp events** • **debug mgcp media** • **debug mgcp packets** • **debug mgcp voipcac** • **debug rtpspi all** • **debug voip ccapi all** • **debug voip dsm all** • **debug voip dspapi all** • **debug voip hpi all** • **debug voip rawmsg** • **debug voip tsp all** • **debug voip vtsp all**

Command Default

Debugging is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(8)T	This command was introduced.

Usage Guidelines

The **debug voip profile fax** command can be used to use a set of debug commands at one time. Because this command generates a large amount of messages, router performance can be affected.

Caution The **debug voip profile fax** command generates debug messages from many VoIP components. The number of messages can impact the performance of your router. This command should only be used during low traffic periods.

Examples

Output has been omitted due to its large volume.

Related Commands

Command	Description
debug voip profile help	Displays the sets of commands supported by the **debug voip profile** commands.
debug voip profile modem	Enables a set of debug commands for modem applications.
debug voip profile voice	Enables a set of debug commands for voice.

debug voip profile help

To display the sets of debug commands supported by the **debug voip profile**commands, use the **debug voip profile help** command in privileged EXEC mode.

debug voip profile help

Command Default

Debugging is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(8)T	This command was introduced.

Usage Guidelines

The **debug voip profile help** command displays the sets of debug commands supported by the **debug voip profile** commands. It does not display any debug output.

Examples

The following is sample output from the **debug voip profile help** command:

```
Router# debug voip profile help

"debug voip profile modem relay signaling" includes:
   debug csm voice
   debug isdn q931
   debug cdapi detail
   debug cdapi events
   debug voip dspapi all
   debug voip hpi all
   debug voip vtsp all
   debug voip tsp all
   debug voip ccapi all
   debug cch323 all
   debug ccsip error
   debug ccsip messages
   debug mgcp errors
   debug mgcp events
   debug mgcp media
   debug mgcp packets
   debug mgcp voipcac
   debug voip dsm all

"debug voip profile voice application" includes:
   debug voip dialpeer all
   debug voip ccapi all
   debug voip ivr all
   debug voip application all
   debug voip application vxml all

"debug voip profile [ voice | modem pass-through ] signaling" includes:
   debug csm voice
   debug isdn q931
   debug cdapi detail
   debug cdapi events
   debug h225 asn1
   debug h225 events
   debug h225 q931
   debug h245 events
   debug h245 asn1
   debug voip dspapi all
   debug voip hpi all
   debug voip vtsp all
   debug voip tsp all
   debug voip ccapi all
   debug cch323 all
   debug rtpspi all
   debug ccsip error
   debug ccsip messages
   debug mgcp errors
   debug mgcp events
   debug mgcp media
   debug mgcp packets
   debug mgcp voipcac
   debug voip rawmsg
   debug gtd error
   debug gtd events
   debug voip dsm all

"debug voip profile fax mail" includes:
   debug csm voice
   debug isdn q931
   debug voip dspapi all
   debug voip hpi all
   debug voip vtsp all
   debug voip ccapi all
   debug voip ivr all
   debug voip application all
   debug voip application vxml all
   debug fmail client
   debug fmail server
   debug fax mta all
   debug fax receive all
   debug fax send all
   debug text-to-fax
   debug tiff reader
   debug tiff writer
   debug fax mmoip aaa
   debug voip dsm all

"debug voip profile fax relay application" includes:
   debug voip dialpeer all
   debug voip ccapi all
   debug voip ivr all
   debug voip application all
   debug voip application vxml all

"debug voip profile fax relay signaling" includes:
   debug csm voice
   debug isdn q931
   debug cdapi detail
   debug cdapi events
   debug h225 asn1
   debug h225 events
   debug h225 q931
   debug h245 events
   debug h245 asn1
   debug voip dspapi all
   debug voip hpi all
   debug voip vtsp all
   debug voip tsp all
   debug voip ccapi all
   debug cch323 all
   debug rtpspi all
   debug ccsip error
   debug ccsip messages
   debug mgcp errors
   debug mgcp events
   debug mgcp media
   debug mgcp packets
   debug mgcp voipcac
   debug voip rawmsg
   debug gtd error
   debug gtd events
```

Related Commands

Command	Description
debug voip profile fax	Enables a set of debug commands for fax applications.
debug voip profile modem	Enables a set of debug commands for modem applications.
debug voip profile voice	Enables a set of debug commands for voice.

debug voip profile modem

To enable a set of debug commands for modem applications, use the **debug voip profile modem** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voip profile modem [**pass-through signaling**| **relay signaling**]

no debug voip profile modem

Syntax Description

pass-through signaling	Enables the following set of debugs for modem pass-through signaling: • **debug cch323 all** • **debug ccsip error all** • **debug ccsip messages** • **debug cdapi detail** • **debug cdapi events** • **debug csm voice** • **debug gtd error** • **debug gtd events** • **debug h225 asn1** • **debug h225 events** • **debug h225 q931** • **debug isdn q931** • **debug mgsp errors all** • **debug mgcp events** • **debug mgcp media** • **debug mgcp packets** • **debug mgcp voipcac** • **debug rtpspi all** • **debug voip ccapi all** • **debug voip dsm all** • **debug voip rawmsg** • **debug voip tsp all** • **debug voip vtsp all** • **debug vpm all**

relay signaling	Enables the following set of debugs for modem relay signaling: • **debug voip ccapi all** • **debug voip vtsp all** • **debug cch323 all** • **debug ccsip error** • **debug ccsip messages all** • **debug mgcp all** • **debug mgcp events** • **debug mgcp media** • **debug mgcp packets** • **debug mgcp voipcac all** • **debug isdn q931**

Command Default

Debugging is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(8)T	This command was introduced.

Usage Guidelines

The **debug voip profile modem** command can be used to use a set of debug commands at one time. Because this command causes a large amount of messages to be generated, router performance can be affected.

Caution The **debug voip profile modem** command generates debug messages from many VoIP components, which generates a large number of debug messages. The number of messages can affect the performance of your router. This command should only be used during low traffic periods.

Examples

Output has been omitted due to its large volume.

Related Commands

Command	Description
debug voip profile fax	Enables a set of debug commands for fax applications.
debug voip profile help	Displays the sets of commands supported by the **debug voip profile** commands.
debug voip profile voice	Enables a set of debug commands for voice.

debug voip profile voice

To enable a set of debug commands for voice, use the **debug voip profile voice** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voip profile voice [**application**| **signaling**]

no debug voip profile voice

Syntax Description

application	Enables the following set of debugs for voice applications: • **debug voip applib** • **debug voip application vxml all** • **debug voip ccapi all** • **debug voip ivr all**

signaling	Enables the following set of debugs for voice signaling: • **debug cch323 all** • **debug ccsip error all** • **debug ccsip messages** • **debug cdapi detail** • **debug cdapi events** • **debug csm voice** • **debug gtd error** • **debug gtd events** • **debug h225 asn1** • **debug h225 events** • **debug h225 q931** • **debug isdn q931** • **debug mgsp errors all** • **debug mgcp events** • **debug mgcp media** • **debug mgcp packets** • **debug mgcp voipcac** • **debug rtpspi all** • **debug voip ccapi all** • **debug voip dsm all** • **debug voip rawmsg** • **debug voip tsp all** • **debug voip vtsp all** • **debug vpm all**

Command Default

Debugging is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(8)T	This command was introduced.

Usage Guidelines

The **debug voip profile voice** command can be used to use a set of debug commands at one time. Because this command causes a large amount of messages to be generated, router performance can be affected.

Caution The **debug voip profile voice** command generates debug messages from many VoIP components, which generates a large number of debug messages. The number of messages can impact the performance of your router. This command should only be used during low traffic periods.

Examples

Output has been omitted due to its large volume.

Related Commands

Command	Description
debug voip profile fax	Enables a set of debug commands for fax applications.
debug voip profile help	Displays the sets of commands supported by the **debug voip profile** commands.
debug voip profile modem	Enables a set of debug commands for modem applications.

debug voip rawmsg

To display the raw message owner, length, and pointer, use the **debug voip rawmsg** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voip rawmsg [detail]

no debug voip rawmsg [detail]

Syntax Description

detail	(Optional) Prints the contents of the raw message in hexadecimal.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(6)T	This command was introduced.
12.2(11)T	The new debug header was added to the following Cisco routers: Cisco 2600 series, Cisco 3620, Cisco 3640, and Cisco 3660; on the following universal gateways: Cisco AS5350, Cisco AS5400, and Cisco AS5850; on the following universal access servers: Cisco AS5300 and Cisco AS5800; and on the Cisco MC3810 multiservice access concentrators.

Usage Guidelines

We recommend that you log output from the **debug voip rawmsg**command to a buffer rather than sending the output to the console; otherwise, the size of the output could severely impact the performance of the gateway.

Examples

The following is sample output from the **debug voip rawmsg**command:

```
Router# debug voip rawmsg
voip rawmsg debugging is on
Router#
*Mar  1 01:16:25.155: //-1/xxxxxxxxxxxx/CCAPI/ccAllocRawMsgInfo: VoIP Raw Msg Al
loc from 1, Length 18 Body 638E0C5
```

These debug messages show that a raw message is allocated for this call. The pointer to the memory location for this raw message is 63075164.

```
*Mar  1 01:16:25.155: //-1/xxxxxxxxxxxx/CCAPI/ccAllocRawMsgInfo: Raw Message ALL
OCATED: ptr is 63075164, owner is 1, length is 18, msg is 638E0C54, type is 0, p
rotocol id is 0
```

The call control API (CCAPI) gets a setup indicator. It has no information about the callid (-1) and GUID *(*xxxxxxxxxxxx*)*.

```
*Mar  1 01:16:25.159: //-1/xxxxxxxxxxxx/CCAPI/cc_api_call_setup_ind:
*Mar  1 01:16:25.159: Raw Message MaMa is TSP owner is CCAPI, length is 77, ptr
is 63075164, type is 0, protocol id is 2
```

The SSAPP at this stage knows about the CallEntry ID (30) but not about GUID (xxxxxxxxx) or the dial-peer (-1).

```
*Mar  1 01:16:25.163: //30/xxxxxxxxxxxx/SSAPP:-1:-1/ssaCallSetupInd:
*Mar  1 01:16:25.163: Raw Message MaMa is TSP owner is SSAPP, length is 77, ptr
is 63075164, type is 0, protocol id is 2
```

The SSAPP learns about the GUID (34C457CD802F) and also learns the incoming dial peer (10002).

```
*Mar  1 01:16:25.163: //30/34C457CD802F/SSAPP:10002:-1/ssaSetupPeer:
*Mar  1 01:16:25.163: Raw Message MaMa is TSP owner is SSAPP, length is 77, ptr
is 63075164, type is 0, protocol id is 2
```

The CCAPI gets the call proceeding for CallEntry ID 30. CCAPI still does not have a GUID (xxxxxxxxxxx).

```
*Mar  1 01:16:25.163: //30/xxxxxxxxxxxx/CCAPI/ccCallProceeding:
```

A new raw message buffer is created and the previous one is released.

```
*Mar  1 01:16:25.163: //-1/xxxxxxxxxxxx/CCAPI/ccAllocRawMsgInfo: VoIP Raw Msg Al
loc from 10, Length 77 Body 0
*Mar  1 01:16:25.167: //-1/xxxxxxxxxxxx/CCAPI/ccAllocRawMsgInfo: Raw Message ALL
OCATED: ptr is 630751EC, owner is 10, length is 77, msg is 638E0F0C, type is 0,
protocol id is 0
*Mar  1 01:16:25.167: //30/34C457CD802F/SSAPP:10002:-1/ssaSetupPeer:
*Mar  1 01:16:25.167: ssaSetupPeer: Saved rawmsgpp 630751EC len 77
IAM,
GCI,34c457cd14f911cc802f95f5fabb6b0f?)??p?34999
*Mar  1 01:16:25.167: //30/xxxxxxxxxxxx/CCAPI/ccCallSetupRequest:
*Mar  1 01:16:25.167: Raw Message MaMa is TSP owner is SSAPP, length is 77, ptr
is 63075164, type is 0, protocol id is 2
*Mar  1 01:16:25.167: //-1/xxxxxxxxxxxx/CCAPI/ccIFCallSetupRequestPrivate:
*Mar  1 01:16:25.167: Raw Message MaMa is TSP owner is SSAPP, length is 77, ptr
is 63075164, type is 0, protocol id is 2
```

The SSAPP gets a message indicating the digits were passed along the VoIP call leg to the terminating gateway. The CallEntry ID is 30, GUID is 34C457CD802F and the incoming dial peer is 10002.

```
*Mar  1 01:16:25.167: //30/34C457CD802F/SSAPP:10002:-1/ssaReportDigitsDone:
The old raw message 63075164 was freed. The new one is 630751EC.
*Mar  1 01:16:25.179: //-1/xxxxxxxxxxxx/CCAPI/ccFreeRawMsgInfo:
Router#Raw Message FREED: ptr is 63075164, owner is 3, length is 4D, msg is 638E
0DB0, type is 0, protocol id is 2
```

CCAPI got a call proceeding on the second call leg (31); it has no information about the GUID (xxxxxxxxx).

```
*Mar  1 01:16:25.223: //31/xxxxxxxxxxxx/CCAPI/cc_api_call_proceeding:
```

CCAPI got a call alert on the second call leg (31); still no information about the GUID (xxxxxxxxx).

```
*Mar  1 01:16:25.227: //31/xxxxxxxxxxxx/CCAPI/cc_api_call_alert:
```

The alert is sent to the first call leg (30), GUID 34C457CD802F.

```
*Mar  1 01:16:25.227: //30/34C457CD802F/SSAPP:10002:-1/ssaAlert:
*Mar  1 01:16:25.227: //30/xxxxxxxxxxxx/CCAPI/ccCallAlert:
The call is answered at this point and the CCAPI gets a call connect for the second call
leg (CallEntry ID is 31; GUID is xxxxxxxxx).
*Mar  1 01:16:40.975: //31/xxxxxxxxxxxx/CCAPI/cc_api_call_connected:
```

The call connect is sent to the first call leg (30), GUID 34C457CD802F.

```
*Mar  1 01:16:40.975: //30/34C457CD802F/SSAPP:10002:-1/ssaConnect:
*Mar  1 01:16:40.975: //30/xxxxxxxxxxxx/CCAPI/ccCallConnect:
```

The current raw message (ptr 630751EC) is released; a new one will be proclaimed when needed.

```
*Mar  1 01:16:40.975: //-1/xxxxxxxxxxxx/CCAPI/ccFreeRawMsgInfo: Raw Message FREE
D: ptr is 630751EC, owner is 10, length is 4D, msg is 638E0F0C, type is 0,
protocol id is 2
```

A new raw message (ptr 63075274) is proclaimed.

```
*Mar  1 01:17:04.007: //-1/xxxxxxxxxxxx/CCAPI/ccAllocRawMsgInfo: VoIP Raw Msg Al
loc from 1, Length 4 Body 638E1068
*Mar  1 01:17:04.007: //-1/xxxxxxxxxxxx/CCAPI/ccAllocRawMsgInfo: Raw Message ALL
OCATED: ptr is 63075274, owner is 1, length is 4, msg is 638E1068, type is 0, protocol id
is 0
```

The call terminates now. CCAPI detects a call disconnect from the first call leg (30) with no GUID (xxxxxxxxx).

```
*Mar  1 01:17:04.007: //30/xxxxxxxxxxxx/CCAPI/cc_api_call_disconnected:
*Mar  1 01:17:04.007: Raw Message MaMa is TSP owner is CCAPI, length is 4, ptr i
s 63075274, type is 0, protocol id is 2
```

The disconnect is sent to the first call leg (30), GUID (34C457CD802F).

```
*Mar  1 01:17:04.011: //30/34C457CD802F/SSAPP:10002:14/ssaDisconnected:
*Mar  1 01:17:04.011: Raw Message MaMa is TSP owner is SSAPP, length is 4, ptr i
s 63075274, type is 0, protocol id is 2
```

The CCAPI disconnects both call legs (incoming 30 and outgoing 31).

```
*Mar  1 01:17:04.011: //30/xxxxxxxxxxxx/CCAPI/ccCallDisconnect:
*Mar  1 01:17:04.011: //31/xxxxxxxxxxxx/CCAPI/ccCallDisconnect:
*Mar  1 01:17:04.011: Raw Message MaMa is TSP owner is SSAPP, length is 4, ptr i
s 63075274, type is 0, protocol id is 2
```

The raw message is released.

```
*Mar  1 01:17:04.015: //-1/xxxxxxxxxxxx/CCAPI/ccFreeRawMsgInfo: Raw Message FREE
D: ptr is 63075274, owner is 3, length is 4, msg is 638E1068, type is 0, protocol id is 2
```

The following example shows output when you use the **debug voip rawmsg detail** command. This example shows that the CCAPI layer received an indication for call setup. The detailed raw message dumps the hex of the message. This output is used to track down data pointing to different variables within the software modules.

```
Router# debug voip rawmsg detail
*Mar  6 17:03:24.169://-1/xxxxxxxxxxxx/CCAPI/ccAllocRawMsgInfo:VoIP Raw Msg Al
loc from 5, Length 0 Body 0
*Mar  6 17:03:24.173://-1/xxxxxxxxxxxx/CCAPI/cc_api_call_setup_ind:
*Mar  6 17:03:24.173:Raw Message MaMa is CCAPI owner is CCAPI, length is 59, ptr is 63045C14,
 type is 0, protocol id is 18
*Mar  6 17:03:24.173:Raw Message is :1C 39 9E 01 00 03 67 74 64 00 00 00 2E 49
 41 4D 2C 0D 0A 47 43 49 2C 35 33 39 61 30 35 39 64 31 39 36 62 31 31 63 63 38 3
9 30 63 39 30 37 65 31 65 31 37 35 61 34 37 0D 0A 0D 0A
```

Related Commands

Command	Description
debug cdapi	Displays information about the call distributor application programming interface.
debug tsp	Displays information about the telephony service provider.
voice call debug	Allows configuration of the voice call debug output.

debug voip rtcp

To enable debugging for Real-Time Transport Control Protocol (RTCP) packets, use the **debug voip rtcp** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voip rtcp {**error**| **packet**| **session**}

no debug voip rtcp

Syntax Description

error	Prints out a trace for error cases.
packet	Provides debug output for RTCP packets.
session	Provides all session debug information.

Command Default

Debugging for RTCP packets is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(2)T	This command was introduced.
12.2(11)T	This command was implemented on the Cisco AS5300, Cisco AS5400, and Cisco AS5850.
12.2(15)T	This command was implemented on the Cisco 1751 and Cisco 1760.

Usage Guidelines

When used without a keyword, this command turns on debugging for all events. This command severely impacts performance; use with caution.

Examples

The following is sample output from the **debug voip rtcp** command:

```
Router# debug voip rtcp
1w0d: voip_rtcp_create_session: callID=37, dstCallID=36 laddr=172.19.169.85, lp0
1w0d: voip_rtcp_get_cname: cname=0.0.0@172.19.169.85
1w0d: voip_rtcp_send_event: event=EV_NEW
1w0d: voip_rtcp_new: rtcp_interval=1893
Router#
1w0d: voip_rtcp_send_event: event=EV_STATS
1w0d: voip_rtcp_stats_req: rtcp_interval=3448
1w0d: voip_rtcp_stats_req:delay=45 lost_packets=0 rtt=0
Router#
1w0d: recv:
```

```
1w0d: SR: ssrc=0x1272A94D sr_ntp_h=0xAF44E045 sr_ntp_l=0xA6CE39C sr_timestamp=02
1w0d: SDES: ssrc=0x1272A94D name=1 len=19 data=0.0.0@172.19.169.77
1w0d: rtcp_round_trip_delay: ssrc=0x1D86A955
Router#
1w0d: voip_rtcp_send_event: event=EV_STATS
1w0d: voip_rtcp_stats_req: rtcp_interval=6394
1w0d: voip_rtcp_stats_req:delay=40 lost_packets=0 rtt=0
1w0d: recv:
1w0d: SR: ssrc=0x1272A94D sr_ntp_h=0xAF44E047 sr_ntp_l=0xFFB007F6 sr_timestamp=6
1w0d: SDES: ssrc=0x1272A94D name=1 len=19 data=0.0.0@172.19.169.77
1w0d: rtcp_round_trip_delay: ssrc=0x1D86A955
Router#
1w0d: voip_rtcp_remove_ccb
1w0d: voip_rtcp_send_event: event=EV_DESTROY
1w0d: voip_rtcp_destroy_idle
1w0d: voip_rtcp_close_session
1w0d: Cleaning up sess=62F95F58, sp=19544, dp=17130
```

debug voip rtp

To enable debugging for Real-Time Transport Protocol (RTP) named event packets, use the **debug voip rtp** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voip rtp {**error**| **session** [**conference**| **dtmf-relay**| **event**| **multicast**| **named-event** [*payload-type*]| **nse**| **text-relay**]| **packet** [**callid** *id-number packet-number*| **remote-ip** *ip-address* **remote-port** *port-number packet-number*]}

no debug voip rtp

Syntax Description

error	Prints out a trace for error cases.
session	Provides all session debug information. If used with a keyword, supplies more specific debug information according to the keywords used.
conference	(Optional) Provides debug information for conference packets.
dtmf-relay	(Optional) Provides debug information for dual-tone multifrequency (DTMF) packets.
event	(Optional) Enables VoIP RTP session generic event debugging trace.
multicast	(Optional) Provides debug information for multicast packets.
named-event	(Optional) Provides debug information for named telephony event (NTE) packets.
nse	(Optional) Provides debug information for named signaling events (NSEs).
text-relay	(Optional) Provides debug information for text-relay packets.
packet	Enables VoIP RTP packet debugging trace.
callid *id-number packet-number*	(Optional) Provides debug information for a specific call ID number (obtained by using the **show voip rtp connections** command). The *packet-number*argument specifies the number of packets to trace so that the display is not flooded.

remote-ip *ip-address* **remote-port** *port-number* *packet-number*	(Optional) Provides debug information for a remote IP address and RTP port number. The *packet-number*argument specifies the number of packets to trace so that the display is not flooded.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(2)XB	This command was introduced.
12.2(8)T	This command was integrated into Cisco IOS Release 12.2(8)T.
12.4(4)XC	This command was implemented on the Cisco AS5300, Cisco AS5400, and Cisco AS5850.
12.2(15)T	This command was implemented on the Cisco 1751 and Cisco 1760.
12.4(6)T	The **text-relay** keyword was added.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.

Usage Guidelines

This command severely impacts performance and should be used only for single-call debug capture.

Examples

The following example shows debugging output for the **debug voip rtp session named-event**command. The example is for a gateway that sends digits 1, 2, 3, then receives digits 9,8,7. The payload type, event ID, and additional packet payload are shown in each log.

The first three packets indicate the start of the tone (initial packet and two redundant). The last three packets indicate the end of the tone (initial packet and two redundant). The packets in between are refresh packets that are sent every 50 milliseconds (without redundancy).

```
Router# debug voip rtp session named-event
00:09:29:          Pt:99     Evt:1       Pkt:03 00 00  <<<Rcv>
00:09:29:          Pt:99     Evt:1       Pkt:03 00 00  <<<Rcv>
00:09:29:          Pt:99     Evt:1       Pkt:03 00 00  <<<Rcv>
00:09:29:          Pt:99     Evt:1       Pkt:03 01 90  <<<Rcv>
00:09:29:          Pt:99     Evt:1       Pkt:03 03 20  <<<Rcv>
00:09:29:          Pt:99     Evt:1       Pkt:03 04 B0  <<<Rcv>
00:09:29:          Pt:99     Evt:1       Pkt:83 04 C8  <<<Rcv>
00:09:29:          Pt:99     Evt:1       Pkt:83 04 C8  <<<Rcv>
00:09:29:          Pt:99     Evt:1       Pkt:83 04 C8  <<<Rcv>
00:09:29:          Pt:99     Evt:2       Pkt:03 00 00  <<<Rcv>
00:09:29:          Pt:99     Evt:2       Pkt:03 00 00  <<<Rcv>
00:09:29:          Pt:99     Evt:2       Pkt:03 00 00  <<<Rcv>
00:09:29:          Pt:99     Evt:2       Pkt:03 01 90  <<<Rcv>
00:09:29:          Pt:99     Evt:2       Pkt:03 03 20  <<<Rcv>
```

```
00:09:29:          Pt:99     Evt:2       Pkt:03 04 B0  <<<Rcv>
00:09:29:          Pt:99     Evt:2       Pkt:83 05 18  <<<Rcv>
00:09:29:          Pt:99     Evt:2       Pkt:83 05 18  <<<Rcv>
00:09:29:          Pt:99     Evt:2       Pkt:83 05 18  <<<Rcv>
00:09:29:          Pt:99     Evt:3       Pkt:03 00 00  <<<Rcv>
00:09:29:          Pt:99     Evt:3       Pkt:03 00 00  <<<Rcv>
00:09:29:          Pt:99     Evt:3       Pkt:03 00 00  <<<Rcv>
00:09:30:          Pt:99     Evt:3       Pkt:03 01 90  <<<Rcv>
00:09:30:          Pt:99     Evt:3       Pkt:03 03 20  <<<Rcv>
00:09:30:          Pt:99     Evt:3       Pkt:03 04 B0  <<<Rcv>
00:09:30:          Pt:99     Evt:3       Pkt:03 06 40  <<<Rcv>
00:09:30:          Pt:99     Evt:3       Pkt:83 06 80  <<<Rcv>
00:09:30:          Pt:99     Evt:3       Pkt:83 06 80  <<<Rcv>
00:09:30:          Pt:99     Evt:3       Pkt:83 06 80  <<<Rcv>
00:09:31:  <Snd>>> Pt:99     Evt:9       Pkt:02 00 00
00:09:31:  <Snd>>> Pt:99     Evt:9       Pkt:02 00 00
00:09:31:  <Snd>>> Pt:99     Evt:9       Pkt:02 00 00
00:09:31:  <Snd>>> Pt:99     Evt:9       Pkt:02 01 90
00:09:31:  <Snd>>> Pt:99     Evt:9       Pkt:02 03 20
00:09:31:  <Snd>>> Pt:99     Evt:9       Pkt:02 04 B0
00:09:31:  <Snd>>> Pt:99     Evt:9       Pkt:02 06 40
00:09:31:  <Snd>>> Pt:99     Evt:9       Pkt:82 06 58
00:09:31:  <Snd>>> Pt:99     Evt:9       Pkt:82 06 58
00:09:31:  <Snd>>> Pt:99     Evt:9       Pkt:82 06 58
00:09:31:  <Snd>>> Pt:99     Evt:8       Pkt:02 00 00
00:09:31:  <Snd>>> Pt:99     Evt:8       Pkt:02 00 00
00:09:31:  <Snd>>> Pt:99     Evt:8       Pkt:02 00 00
00:09:31:  <Snd>>> Pt:99     Evt:8       Pkt:02 01 90
00:09:31:  <Snd>>> Pt:99     Evt:8       Pkt:02 03 20
00:09:31:  <Snd>>> Pt:99     Evt:8       Pkt:02 04 B0
00:09:31:  <Snd>>> Pt:99     Evt:8       Pkt:02 06 40
00:09:31:  <Snd>>> Pt:99     Evt:8       Pkt:82 06 90
00:09:31:  <Snd>>> Pt:99     Evt:8       Pkt:82 06 90
00:09:31:  <Snd>>> Pt:99     Evt:8       Pkt:82 06 90
00:09:31:  <Snd>>> Pt:99     Evt:7       Pkt:02 00 00
00:09:31:  <Snd>>> Pt:99     Evt:7       Pkt:02 00 00
00:09:31:  <Snd>>> Pt:99     Evt:7       Pkt:02 00 00
00:09:31:  <Snd>>> Pt:99     Evt:7       Pkt:02 01 90
00:09:31:  <Snd>>> Pt:99     Evt:7       Pkt:02 03 20
00:09:31:  <Snd>>> Pt:99     Evt:7       Pkt:02 04 B0
00:09:32:  <Snd>>> Pt:99     Evt:7       Pkt:02 06 40
00:09:32:  <Snd>>> Pt:99     Evt:7       Pkt:82 06 58
00:09:32:  <Snd>>> Pt:99     Evt:7       Pkt:82 06 58
00:09:32:  <Snd>>> Pt:99     Evt:7       Pkt:82 06 58
```

The following example shows debugging output for the **debug voip rtp session text-relay**command:

```
Router# debug voip rtp session text-relay
Pt:119    Evt:0     4     247   37    128    Cnt:F7 4B  <Snd>>>
```

Related Commands

Command	Description
text relay protocol	Configures the system-wide protocol type for text packets transmitted between gateways.
text relay rtp	Configures the RTP payload type and redundancy level.

debug voip settlement all

To enable debugging in all settlement areas, use the **debug voip settlement all** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voip settlement all [**enter**| **error**| **exit**| **misc**| **network**| **security**| **transaction**]

no debug voip settlement all [**enter**| **error**| **exit**| **misc**| **network**| **security**| **transaction**]

Syntax Description

enter	(Optional) Displays all entrances.
error	(Optional) Displays information only if an error occurs.
exit	(Optional) Displays all exits.
misc	(Optional) Displays the details on the code flow of each transaction.
network	(Optional) Displays network connectivity data.
security	(Optional) Displays security and encryption errors.
transaction	(Optional) Displays transaction information.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(4)XH1	This command was introduced.

Usage Guidelines

The **debug voip settlement all** command enables the following debug settlement commands:

- **debug voip settlement enter**
- **debug voip settlement error**
- **debug voip settlement exit**
- **debug voip settlement misc**
- **debug voip settlement network**

- **debug voip settlement security**
- **debug voip settlement transaction**

debug voip settlement enter

To show all the settlement function entrances, enter the **debug voip settlement enter** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voip settlement enter

no debug voip settlement enter

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(4)XH1	This command was introduced.

Examples

The following is sample output from the **debug voip settlement enter** command:

```
Router# debug voip settlement enter

00:43:40:OSP:ENTER:OSPPMimeMessageCreate()
00:43:40:OSP:ENTER:OSPPMimeMessageInit()
00:43:40:OSP:ENTER:OSPPMimeMessageSetContentAndLength()
00:43:40:OSP:ENTER:OSPPMimeMessageBuild()
00:43:40:OSP:ENTER:OSPPMimeDataFree()
00:43:40:OSP:ENTER:OSPPMimePartFree()
00:43:40:OSP:ENTER:OSPPMimePartFree()
00:43:40:OSP:ENTER:OSPPMsgInfoAssignRequestMsg()
00:43:40:OSP:ENTER:osppHttpSelectConnection
00:43:40:OSP:ENTER:OSPPSockCheckServicePoint() ospvConnected = <1>
00:43:40:OSP:ENTER:OSPPSockWaitTillReady()
00:43:40:OSP:ENTER:osppHttpBuildMsg()
00:43:40:OSP:ENTER:OSPPSSLSessionWrite()
00:43:40:OSP:ENTER:OSPPSockWrite()
00:43:40:OSP:ENTER:OSPPSockWaitTillReady()
```

debug voip settlement error

To show all the settlement errors, enter the **debug voip settlement error** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voip settlement error

no debug voip settlement error

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(4)XH1	This command was introduced.

```
-1:OSP internal software error.
16:A bad service was chosen.
17:An invalid parameter was passed to OSP.
9010:Attempted to access an invalid pointer.
9020:A time related error occurred.
10010:OSP provider module failed initialization.
10020:OSP provider tried to access a NULL pointer.
10030:OSP provider could not fine transaction collection.
10040:OSP provider failed to obtain provider space.
10050:OSP provider tried to access an invalid handle.
10060:OSP provider has reached the maximum number of providers.
11010:OSP transaction tried to delete a transaction which was not allowed.
11020:OSP transaction tried a transaction which does not exist.
11030:OSP transaction tried to start a transaction, but data had already been delivered.
11040:OSP transaction could not identify the response given.
11050:OSP transaction failed to obtain transaction space.
11060:OSP transaction failed (possibly ran out) to allocate memory.
11070:OSP transaction tried to perform a transaction which is not allowed.
11080:OSP transaction found no more responses.
11090:OSP transaction could not find a specified value.
11100:OSP transaction did not have enough space to copy.
11110:OSP transaction - call id did not match destination.
11120:OSP transaction encountered an invalid entry.
11130:OSP transaction tried to use a token too soon.
11140:OSP transaction tried to use a token too late.
11150:OSP transaction - source is invalid.
11160:OSP transaction - destination is invalid.
11170:OSP transaction - calling number is invalid.
11180:OSP transaction - called number is invalid.
11190:OSP transaction - call id is invalid.
```

```
11200:OSP transaction - authentication id is invalid.
11210:OSP transaction - call id was not found
11220:OSP transaction - The IDS of the called number was invalid.
11230:OSP transaction - function not implemented.
11240:OSP transaction tried to access an invalid handle.
11250:OSP transaction returned an invalid return code.
11260:OSP transaction reported an invalid status code.
11270:OSP transaction encountered an invalid token.
11280:OSP transaction reported a status which could not be identified.
11290:OSP transaction in now valid after it was not found.
11300:OSP transaction could not find the specified destination.
11310:OSP transaction is valid until not found.
11320:OSP transaction - invalid signaling address.
11330:OSP transaction could not find the ID of the transmitter.
11340:OSP transaction could not find the source number.
11350:OSP transaction could not find the destination number.
11360:OSP transaction could not find the token.
11370:OSP transaction could not find the list.
11380:OSP transaction was not allowed to accumulate.
11390:OSP transaction - transaction usage was already reported.
11400:OSP transaction could not find statistics.
11410:OSP transaction failed to create new statistics.
11420:OSP transaction made an invalid calculation.
11430:OSP transaction was not allowed to get the destination.
11440:OSP transaction could not fine the authorization request.
11450:OSP transaction - invalid transmitter ID.
11460:OSP transaction could not find any data.
11470:OSP transaction found no new authorization requests.
12010:OSP security did not have enough space to copy.
12020:OSP security received and invalid argument.
12030:OSP security could not find the private key.
12040:OSP security encountered an un-implemented function.
12050:OSP security ran out of memory.
12060:OSP security received an invalid signal.
12065:OSP security could not initialize the SSL database.
12070:OSP security could not find space for the certificate.
12080:OSP security has no local certificate info defined.
12090:OSP security encountered a zero length certificate.
12100:OSP security encountered a certificate that is too big.
12110:OSP security encountered an invalid certificate.
12120:OSP security encountered a NULL certificate.
12130:OSP security has too many certificates.
12140:OSP security has no storage provided.
12150:OSP security has no private key.
12160:OSP security encountered an invalid context.
12170:OSP security was unable to allocate space.
12180:OSP security - CA certificates do not match.
12190:OSP security found no authority certificates
12200:OSP security - CA certificate index overflow.
13010:OSP error message - failed to allocate memory.
13110:OSP MIME error - buffer is too small.
13115:OSP MIME error - failed to allocate memory.
13120:OSP MIME error - could not find variable.
13125:OSP MIME error - no input was found.
13130:OSP MIME error - invalid argument.
13135:OSP MIME error - no more space.
13140:OSP MIME error - received an invalid type.
13145:OSP MIME error - received an invalid subtype.
13150:OSP MIME error - could not find the specified protocol.
13155:OSP MIME error - could not find MICALG.
13160:OSP MIME error - boundary was not found.
13165:OSP MIME error - content type was not found.
13170:OSP MIME error - message parts were not found.
13301:OSP XML error - received incomplete XML data.
13302:OSP XML error - bad encoding of XML data.
13303:OSP XML error - bad entity in XML data.
13304:OSP XML error - bad name in XML data.
13305:OSP XML error - bad tag in XML data.
13306:OSP XML error - bad attribute in XML data.
13307:OSP XML error - bad CID encoding in XML data.
13308:OSP XML error - bad element found in XML data.
13309:OSP XML error - no element found in XML data.
13310:OSP XML error - no attribute found in XML data.
```

```
13311:OSP XML error - OSP received invalid arguments.
13312:OSP XML error - failed to create a new buffer.
13313:OSP XML error - failed to get the size of a buffer.
13314:OSP XML error - failed to send the buffer.
13315:OSP XML error - failed to read a block from the buffer.
13316:OSP XML error - failed to allocate memory.
13317:OSP XML error - could not find the parent.
13318:OSP XML error - could not find the child.
13319:OSP XML error - data type not found in XML data.
13320:OSP XML error - failed to write a clock to the buffer.
13410:OSP data error - no call id preset.
13415:OSP data error - no token present.
13420:OSP data error - bad number presented.
13425:OSP data error - no destination found.
13430:OSP data error - no usage indicator present.
13435:OSP data error - no status present.
13440:OSP data error - no usage configured.
13445:OSP data error - no authentication indicator.
13450:OSP data error - no authentication request.
13455:OSP data error - no authentication response.
13460:OSP data error - no authentication configuration.
13465:OSP data error - no re-authentication request.
13470:OSP data error - no re-authentication response.
13475:OSP data error - invalid data type present.
13480:OSP data error - no usage information available.
13485:OSP data error - no token info present.
13490:OSP data error - invalid data present.
13500:OSP data error - no alternative info present.
13510:OSP data error - no statistics available.
13520:OSP data error - no delay present.
13610:OSP certificate error - memory allocation failed.
14010:OSP communications error - invalid communication size.
14020:OSP communications error - bad communication value.
14030:OSP communications error - parser error.
14040:OSP communications error - no more memory available.
14050:OSP communications error - communication channel currently in use.
14060:OSP communications error - invalid argument passed.
14070:OSP communications error - no service points present.
14080:OSP communications error - no service points available.
14085:OSP communications error - thread initialization failed.
14086:OSP communications error - communications is shutdown.
14110:OSP message queue error - no more memory available.
14120:OSP message queue error - failed to add a request.
14130:OSP message queue error - no event queue present.
14140:OSP message queue error - invalid arguments passed.
14210:OSP HTTP error - 100 - bad header.
14220:OSP HTTP error - 200 - bad header.
14221:OSP HTTP error - 400 - bad request.
14222:OSP HTTP error - bas service port present.
14223:OSP HTTP error - failed to add a request.
14230:OSP HTTP error - invalid queue present.
14240:OSP HTTP error - bad message received.
14250:OSP HTTP error - invalid argument passed.
14260:OSP HTTP error - memory allocation failed.
14270:OSP HTTP error - failed to create a new connection.
14280:OSP HTTP error - server error.
14290:OSP HTTP error - HTTP server is shutdown.
14292:OSP HTTP error - failed to create a new SSL connection.
14295:OSP HTTP error - failed to create a new SSL context.
14297:OSP HTTP error - service unavailable.
14300:OSP socket error - socket select failed.
14310:OSP socket error - socket receive failed.
14315:OSP socket error - socket send failed.
14320:OSP socket error - failed to allocate memory for the receive buffer.
14320:OSP socket error - socket reset.
14330:OSP socket error - failed to create the socket.
14340:OSP socket error - failed to close the socket.
14350:OSP socket error - failed to connect the socket.
14360:OSP socket error - failed to block I/O on the socket.
14370:OSP socket error - failed to disable nagle on the socket.
14400:OSP SSL error - failed to allocate memory.
14410:OSP SSL error - failed to initialize the context.
14420:OSP SSL error - failed to retrieve the version.
```

```
14430:OSP SSL error - failed to initialize the session.
14440:OSP SSL error - failed to attach the socket.
14450:OSP SSL error - handshake failed.
14460:OSP SSL error - failed to close SSL.
14470:OSP SSL error - failed to read from SSL.
14480:OSP SSL error - failed to write to SSL.
14490:OSP SSL error - could not get certificate.
14495:OSP SSL error - no root certificate found.
14496:OSP SSL error - failed to set the private key.
14497:OSP SSL error - failed to parse the private key.
14498:OSP SSL error - failed to add certificates.
14499:OSP SSL error - failed to add DN.
15410:OSP utility error - not enough space for copy.
15420:OSP utility error - no time stamp has been created.
15430:OSP utility error - value not found.
15440:OSP utility error - failed to allocate memory.
15450:OSP utility error - invalid argument passed.
15500:OSP buffer error - buffer is empty.
15510:OSP buffer error - buffer is incomplete.
15980:OSP POW error.
15990:OSP Operating system conditional variable timeout.
16010:OSP X509 error - serial number undefined.
16020:OSP X509 error - certificate undefined.
16030:OSP X509 error - invalid context.
16040:OSP X509 error - decoding error.
16050:OSP X509 error - unable to allocate space.
16060:OSP X509 error - invalid data present.
16070:OSP X509 error - certificate has expired.
16080:OSP X509 error - certificate not found.
17010:OSP PKCS1 error - tried to access invalid private key pointer
17020:OSP PKCS1 error - unable to allocate space.
17030:OSP PKCS1 error - invalid context found.
17040:OSP PKCS1 error - tried to access NULL pointer.
17050:OSP PKCS1 error - private key overflow.
18010:OSP PKCS7 error - signer missing.
18020:OSP PKCS7 error - invalid signature found.
18020:OSP PKCS7 error - unable to allocate space.
18030:OSP PKCS7 error - encoding error.
18040:OSP PKCS7 error - tried to access invalid pointer.
18050:OSP PKCS7 error - buffer overflow.
19010:OSP ASN1 error - tried to access NULL pointer.
19020:OSP ASN1 error - invalid element tag found.
19030:OSP ASN1 error - unexpected high tag found.
19040:OSP ASN1 error - invalid primitive tag found.
19050:OSP ASN1 error - unable to allocate space.
19060:OSP ASN1 error - invalid context found.
19070:OSP ASN1 error - invalid time found.
19080:OSP ASN1 error - parser error occurred.
19090:OSP ASN1 error - parsing complete.
19100:OSP ASN1 error - parsing defaulted.
19110:OSP ASN1 error - length overflow.
19120:OSP ASN1 error - unsupported tag found.
19130:OSP ASN1 error - object ID not found.
19140:OSP ASN1 error - object ID mismatch.
19150:OSP ASN1 error - unexpected int base.
19160:OSP ASN1 error - buffer overflow.
19170:OSP ASN1 error - invalid data reference ID found.
19180:OSP ASN1 error - no content value for element found.
19190:OSP ASN1 error - integer overflow.
20010:OSP Crypto error - invalid parameters found.
20020:OSP Crypto error - unable to allocate space.
20030:OSP Crypto error - could not verify signature.
20040:OSP Crypto error - implementation specific error.
20050:OSP Crypto error - tried to access invalid pointer.
20060:OSP Crypto error - not enough space to perform operation.
21010:OSP PKCS8 error - invalid private key pointer found.
21020:OSP PKCS8 error - unable to allocate space for operation.
21030:OSP PKCS8 error - invalid context found.
21040:OSP PKCS8 error - tried to access NULL pointer.
21050:OSP PKCS8 error - private key overflow.
22010:OSP Base 64 error - encode failed.
22020:OSP Base 64 error - decode failed.
22510:OSP audit error - failed to allocate memory.
```

```
156010:OSP RSN failure error - no data present.
156020:OSP RSN failure error - data is invalid.
```

Examples

The following is sample output from the **debug voip settlement error** command:

```
Router# debug voip settlement error
00:45:50:OSP:OSPPSockProcessRequest:http recv init header failed
00:45:50:OSP:osppHttpSetupAndMonitor:attempt#0 on http=0x6141A514, limit=1 error=14310
```

debug voip settlement exit

To show all the settlement function exits, enter the **debug voip settlement exit** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voip settlement exit

no debug voip settlement exit

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(4)XH1	This command was introduced.

Examples

The following is sample output from the **debug voip settlement exit**command:

```
Router# debug voip settlement exit

01:21:10:OSP:EXIT :OSPPMimeMessageInit()
01:21:10:OSP:EXIT :OSPPMimeMessageSetContentAndLength()
01:21:10:OSP:EXIT :OSPPMimeMessageBuild()
01:21:10:OSP:EXIT :OSPPMimePartFree()
01:21:10:OSP:EXIT :OSPPMimePartFree()
01:21:10:OSP:EXIT :OSPPMimeDataFree()
01:21:10:OSP:EXIT :OSPPMimeMessageCreate()
01:21:10:OSP:EXIT :OSPPMsgInfoAssignRequestMsg()
01:21:10:OSP:EXIT :osppHttpSelectConnection
01:21:10:OSP:EXIT :OSPPSockCheckServicePoint() isconnected(1)
01:21:10:OSP:EXIT :osppHttpBuildMsg()
01:21:10:OSP:EXIT :OSPPSockWrite() (0)
01:21:10:OSP:EXIT :OSPPSSLSessionWrite() (0)
01:21:10:OSP:EXIT :OSPPSSLSessionRead() (0)
01:21:10:OSP:EXIT :OSPPSSLSessionRead() (0)
01:21:10:OSP:EXIT :OSPPHttpParseHeader
01:21:10:OSP:EXIT :OSPPHttpParseHeader
01:21:10:OSP:EXIT :OSPPSSLSessionRead() (0)
01:21:10:OSP:EXIT :OSPPUtilMemCaseCmp()
```

debug voip settlement misc

To show the details on the code flow of each settlement transaction, enter the **debug voip settlement misc** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voip settlement misc

no debug voip settlement misc

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(4)XH1	This command was introduced.

Examples

The following is sample output from the **debug voip settlement misc**command:

```
Router# debug voip settlement misc
```

```
00:52:03:OSP:osp_authorize:callp=0x6142770C
00:52:03:OSP:OSPPTransactionRequestNew:ospvTrans=0x614278A8
00:52:03:OSP:osppCommMonitor:major:minor=(0x2:0x1)
00:52:03:OSP:HTTP connection:reused
00:52:03:OSP:osppHttpSetupAndMonitor:HTTP=0x6141A514, QUEUE_EVENT from eventQ=0x6141A87C,
comm=0x613F16C4, msginfo=0x6142792C
00:52:03:OSP:osppHttpSetupAndMonitor:connected = <TRUE>
00:52:03:OSP:osppHttpSetupAndMonitor:HTTP=0x6141A514, build msginfo=0x6142792C, trans=0x2
00:52:04:OSP:osppHttpSetupAndMonitor:HTTP=0x6141A514, msg built and sent:error=0,
msginfo=0x6142792C
00:52:04:OSP:osppHttpSetupAndMonitor:monitor exit. errorcode=0
00:52:04:OSP:osppHttpSetupAndMonitor:msginfo=0x6142792C, error=0, shutdown=0
00:52:04:OSP:OSPPMsgInfoProcessResponse:msginfo=0x6142792C, err=0, trans=0x614278A8, handle=2
00:52:04:OSP:OSPPMsgInfoChangeState:transp=0x614278A8, msgtype=12 current state=2
00:52:04:OSP:OSPPMsgInfoChangeState:transp=0x614278A8, new state=4
00:52:04:OSP:OSPPMsgInfoProcessResponse:msginfo=0x6142792C, context=0x6142770C, error=0
00:52:04:OSP:osp_get_destination:trans_handle=2, get_first=1, callinfop=0x614275E0
00:52:04:OSP:osp_get_destination:callinfop=0x614275E0 get dest=1.14.115.51,
validafter=1999-01-20T02:04:32Z, validuntil=1999-01-20T02:14:32Z
00:52:04:OSP:osp_parse_destination:dest=1.14.115.51
00:52:04:OSP:osp_get_destination:callinfop=0x614275E0, error=0, ip_addr=1.14.115.51, credit=60
00:52:06:OSP:stop_settlement_ccapi_accounting:send report for callid=0x11, transhandle=2
00:52:06:OSP:osp_report_usage:transaction=2, duration=0, lostpkts=0, lostfrs=0, lostpktr=0,
 lostfrr=0
```

debug voip settlement network

To show all the messages exchanged between a router and a settlement provider, enter the **debug voip settlement network** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voip settlement network

no debug voip settlement network

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(4)XH1	This command was introduced.

Usage Guidelines

Using the **debug voip settlement network** command shows messages, in detail, in HTTP and XML formats.

Examples

The following is sample output from the **debug voip settlement network**command:

```
Router# debug voip settlement network

00:47:25:OSP:HTTP connection:reused
00:47:25:OSP:OSPPSockWaitTillReady:HTTPCONN=0x6141A514, fd=0
00:47:25:OSP:OSPPSockWaitTillReady:read=0, timeout=0, select=1
00:47:25:OSP:osppHttpBuildAndSend():http=0x6141A514 sending:
POST /scripts/simulator.dll?handler HTTP/1.1
Host:1.14.115.12
content-type:text/plain
Content-Length:439
Connection:Keep-Alive
Content-Type:text/plain
Content-Length:370
<?xml version="1.0"?><Message messageId="1" random="8896">
<AuthorisationRequest componentId="1">
<Timestamp>
1993-03-01T00:47:25Z</Timestamp>
<CallId>
<![CDATA[12]]></CallId>
<SourceInfo type="e164">
5551111</SourceInfo>
<DestinationInfo type="e164">
5552222</DestinationInfo>
<Service/>
<MaximumDestinations>
3</MaximumDestinations>
</AuthorisationRequest>
```

```
</Message>
00:47:25:OSP:OSPPSockWaitTillReady:HTTPCONN=0x6141A514, fd=0
00:47:25:OSP:OSPPSockWaitTillReady:read=0, timeout=1, select=1
00:47:25:OSP:OSPM_SEND:bytes_sent = 577
00:47:25:OSP:OSPPSockProcessRequest:SOCKFD=0, Expecting 100, got
00:47:25:OSP:OSPPSockWaitTillReady:HTTPCONN=0x6141A514, fd=0
00:47:25:OSP:OSPPSockWaitTillReady:read=1, timeout=1, select=1
00:47:25:OSP:OSPPSSLSessionRead() recving 1 bytes:
HTTP/1.1 100 Continue
Server:Microsoft-IIS/4.0
Date:Wed, 20 Jan 1999 02:01:54 GMT
00:47:25:OSP:OSPPSockProcessRequest:SOCKFD=0, Expecting 200, got
00:47:25:OSP:OSPPSockWaitTillReady:HTTPCONN=0x6141A514, fd=0
00:47:25:OSP:OSPPSockWaitTillReady:read=1, timeout=1, select=1
00:47:25:OSP:OSPPSSLSessionRead() recving 1 bytes:
HTTP/1.1 200 OK
Server:Microsoft-IIS/4.0
Date:Wed, 20 Jan 1999 02:01:54 GMT
Connection:Keep-Alive
Content-Type:multipart/signed; protocol="application/pkcs7-signature"; micalg=sha1;
boundary=bar
Content-Length:1689
00:47:25:OSP:OSPPSockProcessRequest:SOCKFD=0, error=0, HTTP response
00:47:25:OSP:OSPPSockWaitTillReady:HTTPCONN=0x6141A514, fd=0
00:47:25:OSP:OSPPSockWaitTillReady:read=1, timeout=1, select=1
00:47:25:OSP:OSPPSSLSessionRead() recving 1689 bytes:
--bar
Content-Type:text/plain
Content-Length:1510
<?xml version="1.0"?><Message messageId="1" random="27285">
<AuthorisationResponse componentId="1">
<Timestamp>
1999-01-20T02:01:54Z</Timestamp>
<Status>
<Description>
success</Description>
<Code>
200</Code>
</Status>
<TransactionId>
101</TransactionId>
<Destination>
<AuthorityURL>
http://www.myauthority.com</AuthorityURL>
<CallId>
<![CDATA[12]]></CallId>
<DestinationInfo type="e164">
5552222</DestinationInfo>
<DestinationSignalAddress>
1.14.115.51</DestinationSignalAddress>
<Token encoding="base64">
[illegible]
<UsageDetail>
<Amount>
60</Amount>
<Increment>
1</Increment>
<Service/>
<Unit>
s</Unit>
</UsageDetail>
<ValidAfter>
1999-01-20T01:59:54Z</ValidAfter>
<ValidUntil>
1999-01-20T02:09:54Z</ValidUntil>
</Destination>
<transnexus.com:DelayLimit critical="False">
1000</transnexus.com:DelayLimit>
<transnexus.com:DelayPreference critical="False">
1</transnexus.com:DelayPreference>
</AuthorisationResponse>
</Message>
--bar
```

```
Content-Type:application/pkcs7-signature
Content-Length:31
This is your response signature
--bar--
```

debug voip settlement security

To show all the tracing related to security, such as Secure Socket Layer (SSL) or Secure Multipurpose Internet Mail Extensions (S/MIME), enter the **debug voip settlement security** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voip settlement security

no debug voip settlement security

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(4)XH1	This command was introduced.

Examples

Not available because of security issues.

debug voip settlement ssl

To display information about the Secure Socket Layer (SSL) connection, use the **debug voip settlement ssl**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voip settlement ssl

no debug voip settlement ssl

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(11)T	This command was introduced.

Usage Guidelines

For complete information about the SSL connection, use the **debug voip settlement ssl** command if you see one of the following errors generated from the **debug voip settlement error** command.

```
14400:OSP SSL error - failed to allocate memory.
14410:OSP SSL error - failed to initialize the context.
14420:OSP SSL error - failed to retrieve the version.
14430:OSP SSL error - failed to initialize the session.
14440:OSP SSL error - failed to attach the socket.
14450:OSP SSL error - handshake failed.
14460:OSP SSL error - failed to close SSL.
14470:OSP SSL error - failed to read from SSL.
14480:OSP SSL error - failed to write to SSL.
14490:OSP SSL error - could not get certificate.
14495:OSP SSL error - no root certificate found.
14496:OSP SSL error - failed to set the private key.
14497:OSP SSL error - failed to parse the private key.
14498:OSP SSL error - failed to add certificates.
14499:OSP SSL error - failed to add DN.
```

Examples

The following example shows the debug output when the SSL is making a good connection to the Open Settlement Protocol server:

```
*May 15 11:53:42.871:OSP:
*May 15 11:53:42.871:OSPPSSLConnect:****** SSL HANDSHAKE SUCCEED !!**** retry=2
```

When the SSL connection is closed, the following message appears:

```
*May 15 11:57:42.541:OSP:osp_ssl_close:OSPPSSLClose succeed
```

The following are possible output trace messages:

```
osp_ssl_callback_add_session:session not found, add it.
osp_ssl_callback_add_session:session found, but not equal, delete old one
osp_ssl_callback_add_session:Copy new session data
osp_ssl_callback_add_session:session found and equal. no add
osp_ssl_callback_get_session:No Session exist
osp_ssl_callback_get_session:Session found, copy to sslref length=756
osp_ssl_callback_delete_session:session not found
```

These messages do not indicate an error but indicate the result of the operation.

To display actual error messages, enter the **debug voip settlement error** command.

debug voip settlement transaction

To see all the attributes of the transactions on the settlement gateway, use the **debug voip settlement transaction in privileged EXEC**mode. To disable debugging output, use the **no** form of this command.

debug voip settlement transaction

no debug voip settlement transaction

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(4)XH1	This command was introduced.

Examples

The following is sample output from the originating gateway:

```
00:44:54:OSP:OSPPTransactionNew:trans=0, err=0
00:44:54:OSP:osp_authorize:authorizing trans=0, err=0
router>
00:45:05:OSP:stop_settlement_ccapi_accounting:send report for
callid=7, trans
=0, calling=5710868, called=15125551212, curr_Dest=1
00:45:05:OSP:OSPPTransactionDelete:deleting trans=0
```

The following is sample output from the terminating gateway:

```
00:44:40:OSP:OSPPTransactionNew:trans=0, err=0
00:44:40:OSP:osp_validate:validated trans=0, error=0, authorised=1
```

debug voip tsp

To display information about the telephony service provider (TSP), use the **debug voip tsp** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voip tsp [**all**| **default**| **error** [**call [informational]**| **software [informational]**]| **event**| **function**| **individual** *range*| **inout**| **rose**]

no debug voip tsp

Syntax Description

all	(Optional) Displays all TSP debugging messages.
default	(Optional) Displays TSP inout, error, and event debugging messages. This option also runs if no keywords are added.
error	(Optional) Displays TSP error messages.
call	(Optional) Displays TSP call-related debugs not generated by other TSP debug options.
informational	(Optional) Displays minor errors and major errors. Without the **informational** keyword, only major errors are displayed.
software	(Optional) Displays software processing errors.
event	(Optional) Displays TSP events.
function	(Optional) Displays TSP functions.
individual	(Optional) Enables individual TSP debugs.
range	For the **individual** keyword, the range is an integer value from 1 to 68. For specific range values, see the table below.
inout	(Optional) Displays TSP function entry/exit debugs.
rose	(Optional) Enables the remote operations service element. This debug displays information about ISDN-related elements.

Table 64: TSP Individual Debug Values

Value	TSP Debug Function
1	INDIVIDUAL_TSP_DEBUG_TDM_HAIRPIN_CONNECT_001
2	INDIVIDUAL_TSP_DEBUG_TDM_HAIRPIN_DISCONNECT_002
3	INDIVIDUAL_TSP_DEBUG_CCRAWMSG_ENCAP_003
4	INDIVIDUAL_TSP_DEBUG_CDAPI_FORM_MSG_BASIC_SS_INFO_004
5	INDIVIDUAL_TSP_DEBUG_CDAPI_FORM_MSG_005
6	INDIVIDUAL_TSP_DEBUG_CDAPI_FORM_MSG_006
7	INDIVIDUAL_TSP_DEBUG_CDAPI_SEND_MSG_007
8	INDIVIDUAL_TSP_DEBUG_CDAPI_SEND_MSG_008
9	INDIVIDUAL_TSP_DEBUG_CDAPI_SEND_INFO_MSG_009
10	INDIVIDUAL_TSP_DEBUG_ALLOC_CDB_010
11	INDIVIDUAL_TSP_DEBUG_DEALLOC_CDB_011
12	INDIVIDUAL_TSP_DEBUG_CONNECT_IND_012
13	INDIVIDUAL_TSP_DEBUG_CONNECT_IND_EXIT_013
14	INDIVIDUAL_TSP_DEBUG_CONNECT_IND_EXIT_014
15	INDIVIDUAL_TSP_DEBUG_CONNECT_IND_EXIT_015
16	INDIVIDUAL_TSP_DEBUG_CONNECT_IND_EXIT_016
17	INDIVIDUAL_TSP_DEBUG_CONNECT_IND_EXIT_017
18	INDIVIDUAL_TSP_DEBUG_CDAPI_SETUP_ACK_018
19	INDIVIDUAL_TSP_DEBUG_CDAPI_PROCEEDING_019
20	INDIVIDUAL_TSP_DEBUG_CDAPI_ALERT_020
21	INDIVIDUAL_TSP_DEBUG_CDAPI_CONNECT_021
22	INDIVIDUAL_TSP_DEBUG_CDAPI_INFO_022
23	INDIVIDUAL_TSP_DEBUG_CDAPI_PROGRESS_023
24	INDIVIDUAL_TSP_DEBUG_CDAPI_FACILITY_024

Value	TSP Debug Function
25	INDIVIDUAL_TSP_DEBUG_CDAPI_FACILITY_025
26	INDIVIDUAL_TSP_DEBUG_CDAPI_PRE_CONN_DISC_REQ_026
27	INDIVIDUAL_TSP_DEBUG_CDAPI_DISC_PROG_IND_027
28	INDIVIDUAL_TSP_DEBUG_CDAPI_DISCONNECT_REQ_028
29	INDIVIDUAL_TSP_DEBUG_CDAPI_DISCONNECT_REQ_029
30	INDIVIDUAL_TSP_DEBUG_CDAPI_TSP_SS_RESP_030
31	INDIVIDUAL_TSP_DEBUG_CDAPI_TSP_INFO_IND_031
32	INDIVIDUAL_TSP_DEBUG_CDAPI_TSP_PROCEEDING_032
33	INDIVIDUAL_TSP_DEBUG_CDAPI_TSP_ALERT_033
34	INDIVIDUAL_TSP_DEBUG_CDAPI_TSP_ALERT_EXIT_034
35	INDIVIDUAL_TSP_DEBUG_CDAPI_TSP_ALERT_EXIT_035
36	INDIVIDUAL_TSP_DEBUG_CDAPI_TSP_PROGRESS_036
37	INDIVIDUAL_TSP_DEBUG_CDAPI_TSP_INFO_037
38	INDIVIDUAL_TSP_DEBUG_CDAPI_TSP_CONNECT_038
39	INDIVIDUAL_TSP_DEBUG_CDAPI_TSP_CONNECT_CONF_039
40	INDIVIDUAL_TSP_DEBUG_CDAPI_TSP_DISC_PROG_IND_040
41	INDIVIDUAL_TSP_DEBUG_CDAPI_TSP_PROG_IND_PROGRESS_041
42	INDIVIDUAL_TSP_DEBUG_CDAPI_TSP_RELEASE_IND_042
43	INDIVIDUAL_TSP_DEBUG_CDAPI_TSP_RELEASE_IND_EXIT_043
44	INDIVIDUAL_TSP_DEBUG_CDAPI_TSP_RELEASE_COMP_044
45	INDIVIDUAL_TSP_DEBUG_CDAPI_TSP_RELEASE_COMP_CLEAR_045
46	INDIVIDUAL_TSP_DEBUG_SETUP_REQ_EXIT_046
47	INDIVIDUAL_TSP_DEBUG_SETUP_REQ_EXIT_047
48	INDIVIDUAL_TSP_DEBUG_SETUP_REQ_EXIT_048

Value	TSP Debug Function
49	INDIVIDUAL_TSP_DEBUG_TSP_SET_TRANSFER_INFO_049
50	INDIVIDUAL_TSP_DEBUG_TSP_CALL_VOICE_CUT_THROUGH_050
51	INDIVIDUAL_TSP_DEBUG_TSP_CALL_VOICE_CUT_THROUGH_051
52	INDIVIDUAL_TSP_DEBUG_TSP_CALL_VOICE_CUT_THROUGH_052
53	INDIVIDUAL_TSP_DEBUG_TSP_CALL_VOICE_CUT_THROUGH_053
54	INDIVIDUAL_TSP_DEBUG_TSP_MAIN_054
55	INDIVIDUAL_TSP_DEBUG_DO_GLOBAL_END_TO_END_DISC_055
56	INDIVIDUAL_TSP_DEBUG_TSP_CDAPI_MSG_DUMP_056
57	INDIVIDUAL_TSP_DEBUG_TSP_COT_TIMER_START_057
58	INDIVIDUAL_TSP_DEBUG_TSP_COT_TIMER_STOP_058
59	INDIVIDUAL_TSP_DEBUG_TSP_COT_RESULT_059
60	INDIVIDUAL_TSP_DEBUG_TSP_COT_DONE_060
61	INDIVIDUAL_TSP_DEBUG_TSP_COT_TIMEOUT_061
62	INDIVIDUAL_TSP_DEBUG_TSP_COT_REQ_062
63	INDIVIDUAL_TSP_DEBUG_CDAPI_TSP_COT_SETUP_ACK_063
64	INDIVIDUAL_TSP_DEBUG_CDAPI_TSP_RCV_COT_MSG_064
65	INDIVIDUAL_TSP_DEBUG_CDAPI_TSP_RCV_COT_MSG_065
66	INDIVIDUAL_TSP_DEBUG_TSP_CDAPI_PUT_CAUSE_IE_066
67	INDIVIDUAL_TSP_DEBUG_CDAPI_TSP_SETUP_ACK_067
68	INDIVIDUAL_TSP_DEBUG_CDAPI_TSP_RCV_MSG_068

Command Default

Debugging is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(8)T	This command replaces the **debug tsp** command.

Examples

The following is sample output from the **debug voip tsp**command:

```
Router# debug voip tsp

Apr  4 2002 14:04:11.034 UTC://-1/xxxxxxxxxxxx/TSP:():-1/FFFF/tsp_voice_call_check:Query#9
   Overlap=FALSE, Called Number=222, Calling Number=4321
*Apr  4 2002 14:04:11.034 UTC://-1/xxxxxxxxxxxx/TSP:():-1/FFFF/tsp_check_call_type:Query#9

   Peer Search Type=Voice
*Apr  4 2002 14:04:11.034 UTC://-1/xxxxxxxxxxxx/TSP:():-1/FFFF/tsp_check_call_type:Query#9

   Matched Incoming Dialpeer With=Port, Peer=299
*Apr  4 2002 14:04:11.034 UTC://-1/xxxxxxxxxxxx/TSP:():-1/FFFF/tsp_check_call_type:Query#9

   DID=TRUE
*Apr  4 2002 14:04:11.034 UTC://-1/xxxxxxxxxxxx/TSP:():-1/FFFF/tsp_check_call_type:Query#9

   Incoming Dial-peer=SUCCESS
*Apr  4 2002 14:04:11.034 UTC://-1/xxxxxxxxxxxx/TSP:():-1/FFFF/tsp_check_call_type:Query#9

   Matched Outgoing Dialpeer=221
*Apr  4 2002 14:04:11.034 UTC://-1/xxxxxxxxxxxx/TSP:():-1/FFFF/tsp_voice_call_check:Query#9

   Call Type=VOICE, Result=ACCEPT
```

The table below describes the significant fields shown in the display.

Table 65: debug voip tsp dialpeer Field Descriptions

Field	Description
//-1/xxxxxxxxxxxx/TSP:():-1/FFFF/ tsp_voice_call_check:	The format of this message is //callid/GUID/DMSP/function name: • CallEntry ID is -1. This indicates that a call leg has not been identified. • GUID is xxxxxxxxxxxx. This indicates that the call has not been specified. • TSP:():-1/FFFFis the module name and module-specific parameters. • Thetsp_voice_call_checkfield shows that the accounting for an onramp fax is active.
Called Number=222, Calling Number=4321	Shows the calling and called numbers for the call.
Matched Incoming Dialpeer With=Port, Peer=299	Shows that the incoming dial peer was matched and identifies the dial peer.

Field	Description
DID=TRUE	Indicates that the call is a direct-inward dial (DID) call.
Matched Outgoing Dialpeer=221	Shows that the outgoing dial peer was matched and identifies the dial peer.

Related Commands

Command	Description
debug track	Displays information about the telephony service provider.
debug voip rawmsg	Displays the raw message owner, length, and pointer.

debug voip vtsp

To display information about the voice telephony service provider (VTSP), use the **debug voip vtsp** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voip vtsp [**all**| **default**| **error**| **event**| **function**| **individual** *range*| **inout**| **session**| **tone**]

no debug voip vtsp

Syntax Description

all	(Optional) Displays all VTSP debugging messages.
default	(Optional) Displays VTSP inout, error, and event debugging messages. This option also runs if no keywords are added.
error	(Optional) Displays VTSP error messages.
event	(Optional) Displays VTSP events.
function	(Optional) Displays VTSP functions.
individual	(Optional) Enables individual VTSP debugs.
range	For the **individual** keyword, the range is an integer value from 1 to 102. For specific range values, see the table below.
inout	(Optional) Displays VTSP function entry/exit debugs.
session	(Optional) Traces how the router interacts with the digital signal processor (DSP) based on the signaling indications from the signaling stack and requests from the application.
tone	(Optional) Displays the VTSP messages showing the types of tones generated by the Voice over IP (VoIP) gateway.

Table 66: VTSP Individual Debug Values

Value	VTSP Debug Function
1	INDIVIDUAL_VTSP_DEBUG_SETUP_REQ_PEND_DEFER_001
2	INDIVIDUAL_VTSP_DEBUG_SETUP_REQ_WAIT_PEND_SUCCESS_002
3	INDIVIDUAL_VTSP_DEBUG_SETUP_REQ_WAIT_PEND_FAIL_003

Value	VTSP Debug Function
4	INDIVIDUAL_VTSP_DEBUG_TDM_HPM_COMPLETE_004
5	INDIVIDUAL_VTSP_DEBUG_TDM_HPM_COMPLETE_EXIT_005
6	INDIVIDUAL_VTSP_DEBUG_TDM_HPM_CHECK_006
7	INDIVIDUAL_VTSP_DEBUG_TDM_HPM_CHECK_EXIT_007
8	INDIVIDUAL_VTSP_DEBUG_GENERATE_DISC_008
9	INDIVIDUAL_VTSP_DEBUG_GENERATE_DISC_EXIT_009
10	INDIVIDUAL_VTSP_DEBUG_SETUP_IND_ACK_010
11	INDIVIDUAL_VTSP_DEBUG_SETUP_IND_ACK_EXIT_011
12	INDIVIDUAL_VTSP_DEBUG_PROCEEDING_012
13	INDIVIDUAL_VTSP_DEBUG_PRE_CON_DISCONNECT_013
14	INDIVIDUAL_VTSP_DEBUG_PRE_CON_DISCONNECT_EXIT_014
15	INDIVIDUAL_VTSP_DEBUG_SET_DIGIT_TIMEOUTS_015
16	INDIVIDUAL_VTSP_DEBUG_CONNECT_016
17	INDIVIDUAL_VTSP_DEBUG_LOOPBACK_017
18	INDIVIDUAL_VTSP_DEBUG_RING_NOAN_TIMER_018
19	INDIVIDUAL_VTSP_DEBUG_ALERT_CONNECT_019
20	INDIVIDUAL_VTSP_DEBUG_PRE_CON_DISC_REL_EXIT_020
21	INDIVIDUAL_VTSP_DEBUG_HOST_DISC_CLEANUP_021
22	INDIVIDUAL_VTSP_DEBUG_HOST_DISC_CLEANUP_EXIT_022
23	INDIVIDUAL_VTSP_DEBUG_DISCONNECT_023
24	INDIVIDUAL_VTSP_DEBUG_DISCONNECT_EXIT_024
25	INDIVIDUAL_VTSP_DEBUG_DISCONNECT_EXIT_025
26	INDIVIDUAL_VTSP_DEBUG_CONNECT_DIAL_026
27	INDIVIDUAL_VTSP_DEBUG_SETUP_PEND_DIAL_027

Value	VTSP Debug Function
28	INDIVIDUAL_VTSP_DEBUG_PRE_DISC_CAUSE_028
29	INDIVIDUAL_VTSP_DEBUG_SETUP_PEND_CONNECT_029
30	INDIVIDUAL_VTSP_DEBUG_SETUP_REQ_PEND_FAIL_030
31	INDIVIDUAL_VTSP_DEBUG_SETUP_REQ_DISC_031
32	INDIVIDUAL_VTSP_DEBUG_RELEASE_TIMEOUT_032
33	INDIVIDUAL_VTSP_DEBUG_SETUP_PEND_PROCEEDING_EXIT_033
34	INDIVIDUAL_VTSP_DEBUG_SETUP_PEND_PROCEEDING_EXIT_034
35	INDIVIDUAL_VTSP_DEBUG_PEND_RELEASE_IND_035
36	INDIVIDUAL_VTSP_DEBUG_PEND_RELEASE_IND_EXIT_036
37	INDIVIDUAL_VTSP_DEBUG_DISCONNECT_NO_DSP_CHAN_037
38	INDIVIDUAL_VTSP_DEBUG_DISCONNECT_NO_DSP_CHAN_EXIT_038
39	INDIVIDUAL_VTSP_DEBUG_CALL_FEATURE_IND_039
40	INDIVIDUAL_VTSP_DEBUG_SETUP_PEND_PROGRESS_040
41	INDIVIDUAL_VTSP_DEBUG_SETUP_PEND_ALERT_041
42	INDIVIDUAL_VTSP_DEBUG_SETUP_PEND_ALERT_EXIT_042
43	INDIVIDUAL_VTSP_DEBUG_SETUP_PEND_FIRST_PROGRESS_043
44	INDIVIDUAL_VTSP_DEBUG_SETUP_PEND_FIRST_PROGRESS_EXIT_044
45	INDIVIDUAL_VTSP_DEBUG_SETUP_PEND_FIRST_PROGRESS_EXIT_045
46	INDIVIDUAL_VTSP_DEBUG_SETUP_PEND_PROG_PROCEEDING_046
47	INDIVIDUAL_VTSP_DEBUG_PROCEEDING_R2_PEND_DIAL_047
48	INDIVIDUAL_VTSP_DEBUG_ALERT_R2_PEND_DIAL_048
49	INDIVIDUAL_VTSP_DEBUG_CONN_R2_PEND_DIAL_049
50	INDIVIDUAL_VTSP_DEBUG_SETUP_R2_PEND_DIAL_050
51	INDIVIDUAL_VTSP_DEBUG_R2_PEND_DIAL_ALL_051

Value	VTSP Debug Function
52	INDIVIDUAL_VTSP_DEBUG_INFO_IND_052
53	INDIVIDUAL_VTSP_DEBUG_ALERT_053
54	INDIVIDUAL_VTSP_DEBUG_ALERT_EXIT_054
55	INDIVIDUAL_VTSP_DEBUG_PROGRESS_055
56	INDIVIDUAL_VTSP_DEBUG_DISC_PROG_IND_056
57	INDIVIDUAL_VTSP_DEBUG_SETUP_PEND_DISC_PI_IND_057
58	INDIVIDUAL_VTSP_DEBUG_INFO_058
59	INDIVIDUAL_VTSP_DEBUG_FEATURE_059
60	INDIVIDUAL_VTSP_DEBUG_SETUP_PEND_ALERT_NO_TIMEOUT_060
61	INDIVIDUAL_VTSP_DEBUG_SETUP_PEND_ALERT_NO_TIMEOUT_EXIT_061
62	INDIVIDUAL_VTSP_DEBUG_CALL_FEATURE_ENABLE_062
63	INDIVIDUAL_VTSP_DEBUG_XCCSM_COT_TEST_DONE_063
64	INDIVIDUAL_VTSP_DEBUG_XCCSM_COT_TEST_TIMEOUT_064
65	INDIVIDUAL_VTSP_DEBUG_XCCSM_COT_TEST_065
66	INDIVIDUAL_VTSP_DEBUG_CALL_FEATURE_066
67	INDIVIDUAL_VTSP_DEBUG_TCSM_COT_TEST_DONE_067
68	INDIVIDUAL_VTSP_DEBUG_TCSM_COT_TEST_TIMEOUT_068
69	INDIVIDUAL_VTSP_DEBUG_TCSM_ACT_COT_TEST_069
70	INDIVIDUAL_VTSP_DEBUG_PLAY_BUSY_TIMER_START_070
71	INDIVIDUAL_VTSP_DEBUG_PLAY_BUSY_TIMER_STOP_071
72	INDIVIDUAL_VTSP_DEBUG_RING_NOAN_TIMER_START_072
73	INDIVIDUAL_VTSP_DEBUG_RING_NOAN_TIMER_STOP_073
74	INDIVIDUAL_VTSP_DEBUG_VTSP_TIMER_074
75	INDIVIDUAL_VTSP_DEBUG_VTSP_TIMER_STOP_075

Value	VTSP Debug Function
76	INDIVIDUAL_VTSP_DEBUG_VTSP_ALLOCATE_CDB_076
77	INDIVIDUAL_VTSP_DEBUG_VTSP_DO_CALL_SETUP_IND_077
78	INDIVIDUAL_VTSP_DEBUG_VTSP_DO_CALL_SETUP_IND_EXIT_078
79	INDIVIDUAL_VTSP_DEBUG_VTSP_REQUEST_CALL_079
80	INDIVIDUAL_VTSP_DEBUG_VTSP_REQUEST_CALL_EXIT_080
81	INDIVIDUAL_VTSP_DEBUG_VTSP_REALLOC_CDB_081
82	INDIVIDUAL_VTSP_DEBUG_VTSP_OG_CALL_REQ_EXIT_082
83	INDIVIDUAL_VTSP_DEBUG_VTSP_FREE_CDB_083
84	INDIVIDUAL_VTSP_DEBUG_TGRM_DISC_REL_084
85	INDIVIDUAL_VTSP_DEBUG_VTSP_CC_CALL_DISCONNECTED_085
86	INDIVIDUAL_VTSP_DEBUG_SIGO_BDROP_086
87	INDIVIDUAL_VTSP_DEBUG_SIGO_PRE_CON_DISCONNECT_087
88	INDIVIDUAL_VTSP_DEBUG_SIGO_PROCEEDING_088
89	INDIVIDUAL_VTSP_DEBUG_SIGO_GENERATE_DISC_089
90	INDIVIDUAL_VTSP_DEBUG_SIGO_ALERT_090
91	INDIVIDUAL_VTSP_DEBUG_SIGO_ALERT_CONNECT_091
92	INDIVIDUAL_VTSP_DEBUG_SIGO_SETUP_PEND_CONNECT_092
93	INDIVIDUAL_VTSP_DEBUG_DO_SIGO_CALL_SETUP_REQ_093
94	INDIVIDUAL_VTSP_DEBUG_DO_SIGO_CALL_SETUP_REQ_SESSION_094
95	INDIVIDUAL_VTSP_DEBUG_DSM_MEDIA_EVENT_CB_095
96	INDIVIDUAL_VTSP_DEBUG_DSM_PEER_EVENT_CB_096
97	INDIVIDUAL_VTSP_DEBUG_DSM_FEATURE_NOTIFY_CB_097
98	INDIVIDUAL_VTSP_DEBUG_DSM_BRIDGE_CHECK_CB_098
99	INDIVIDUAL_VTSP_DEBUG_DSM_BRIDGE_STATUS_EXIT_099

Value	VTSP Debug Function
100	INDIVIDUAL_VTSP_DEBUG_DSM_SET_FAX_FEAT_EXIT_100
101	INDIVIDUAL_VTSP_DEBUG_DS_DO_DIAL_101
102	INDIVIDUAL_VTSP_DEBUG_DS_DIALING_DEFAULT_102

Command Default

Debugging is not enabled.

Command Modes

Privileged EXEC

Command History

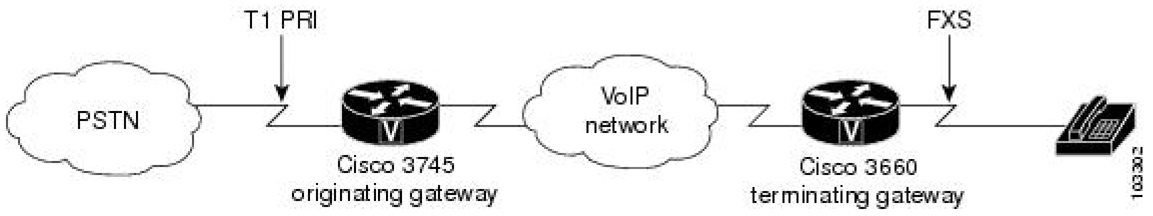
Release	Modification
12.3(8)T	This command replaces the **debug vtsp** command.
12.3(14)T	T.38 fax relay call statistics were made available to Call Detail Records (CDRs) through Vendor-Specific Attributes (VSAs) and added to the call log.

Examples

The following examples show output for variations of the **debug voip vtsp** command:

For these examples, the topology shown in the figure below is used.

Figure 5: Network Topology for debug voip vtsp Examples


T1 PRI
FXS
PSTN
VoIP network
Cisco 3745 originating gateway
Cisco 3660 terminating gateway
103302

Examples

```
Router# debug voip vtsp event
voip vtsp event debugging is on
*May  1 20:03:47.703: //-1/xxxxxxxxxxxx/VTSP:(4/0/0):-1:-1:-1/vtsp_process_event:
   [state:INVALID STATE MACHINE, event:E_CC_SETUP_REQ]
```

At the setup request, the CallEntry ID and GUID are set. The remainder of the output follows the progress of the call.

```
*May  1 20:03:47.707: //78/CDDFE7FF8029/VTSP:(4/0/0):-1:1:1/vtsp_process_event:
   [state:S_SETUP_REQUEST, event:E_TSP_PROCEEDING]
*May  1 20:03:47.707: //78/CDDFE7FF8029/VTSP:(4/0/0):-1:1:1/vtsp_process_event:
   [state:S_SETUP_REQ_PROC, event:E_TSP_PROGRESS]
*May  1 20:03:49.955: //78/CDDFE7FF8029/VTSP:(4/0/0):-1:1:1/vtsp_process_event:
   [state:S_SETUP_REQ_PROC, event:E_TSP_CONNECT]
*May  1 20:03:49.959: //78/CDDFE7FF8029/VTSP:(4/0/0):-1:1:1/vtsp_process_event:
   [state:S_CONNECT, event:E_CC_FEATURE]
*May  1 20:04:14.851: //78/CDDFE7FF8029/VTSP:(4/0/0):-1:1:1/vtsp_process_event:
   [state:S_CONNECT, event:E_CC_DISCONNECT]
*May  1 20:04:14.855: //78/CDDFE7FF8029/VTSP:(4/0/0):-1:1:1/vtsp_process_event:
   [state:S_WAIT_STATS, event:E_VTSP_DSM_STATS_COMPLETE]
*May  1 20:04:15.759: //78/CDDFE7FF8029/VTSP:(4/0/0):-1:1:1/vtsp_process_event:
   [state:S_WAIT_RELEASE, event:E_TSP_CALL_FEATURE_IND]
*May  1 20:04:15.811: //78/CDDFE7FF8029/VTSP:(4/0/0):-1:1:1/vtsp_process_event:
   [state:S_WAIT_RELEASE, event:E_TSP_CALL_FEATURE_IND]
*May  1 20:04:15.811: //78/CDDFE7FF8029/VTSP:(4/0/0):-1:1:1/vtsp_process_event:
   [state:S_WAIT_RELEASE, event:E_TSP_DISCONNECT_CONF]
*May  1 20:04:15.811: //78/CDDFE7FF8029/VTSP:(4/0/0):-1:-1:-1/vtsp_process_event:
   [state:S_CLOSE_DSPRM, event:E_VTSP_DSM_CLOSE_COMPLETE]
```

Examples

```
Router# **debug voip vtsp function**

voip vtsp function debugging is on
*Apr 18 21:48:25.671: //-1/xxxxxxxxxxxx/VTSP:(2/1:23):-1:-1:-1/vtsp_do_call_setup_ind:
```

At the setup request, the CallEntry ID and GUID are set. The call setup functions are shown.

```
*Apr 18 21:48:25.671: //-1/D87794B9802B/VTSP:(2/1:23):0:-1:-1/vtsp_do_normal_call_setup_ind:
*Apr 18 21:48:25.671: //-1/D87794B9802B/VTSP:(2/1:23):0:8:4/vtsp_indicate_call:
*Apr 18 21:48:25.675: //88/D87794B9802B/VTSP:(2/1:23):0:8:4/vtsp_insert_cdb:
*Apr 18 21:48:25.675: //88/D87794B9802B/VTSP:(2/1:23):0:8:4/act_proceeding:
*Apr 18 21:48:25.687: //88/D87794B9802B/VTSP:(2/1:23):0:8:4/act_progress:
*Apr 18 21:48:25.687: //88/D87794B9802B/VTSP:(2/1:23):0:8:4/vtsp_dsm_bridge_check_cb:
**\*Apr 18 21:48:25.687: //88/D87794B9802B/VTSP:(2/1:23):0:8:4/vtsp_dsm_bridge_check_cb:exit@1066**
*Apr 18 21:48:25.687: //88/D87794B9802B/VTSP:(2/1:23):0:8:4/vtsp_dsm_bridge_status_cb:
*Apr 18 21:48:25.687: //88/D87794B9802B/VTSP:(2/1:23):0:8:4/vtsp_dsm_save_fax_config:
*Apr 18 21:48:25.687: //88/D87794B9802B/VTSP:(2/1:23):0:8:4/vtsp_dsm_set_fax_feat_param:
*Apr 18 21:48:25.687: //88/D87794B9802B/VTSP:(2/1:23):0:8:4/vtsp_dsm_peer_event_cb:
*Apr 18 21:48:25.687: //88/D87794B9802B/VTSP:(2/1:23):0:8:4/vtsp_dsm_peer_event_cb:
*Apr 18 21:48:25.687: //88/D87794B9802B/VTSP:(2/1:23):0:8:4/vtsp_dsm_feature_notify_cb:
*Apr 18 21:48:25.687: //88/D87794B9802B/VTSP:(2/1:23):0:8:4/vtsp_dsm_reactivate_ringback:
**\*Apr 18 21:48:25.687:**
**//88/D87794B9802B/VTSP:(2/1:23):0:8:4/vtsp_dsm_reactivate_ringback:exit@871**
*Apr 18 21:48:27.451: //88/D87794B9802B/VTSP:(2/1:23):0:8:4/vtsp_dsm_peer_event_cb:
```

At this point, the ringback to the caller has occurred and the next event shows a connection.

```
*Apr 18 21:48:28.635: //88/D87794B9802B/VTSP:(2/1:23):0:8:4/act_connect:
*Apr 18 21:48:29.003: //88/D87794B9802B/VTSP:(2/1:23):0:8:4/vtsp_dsm_peer_event_cb:
*Apr 18 21:48:34.059: //88/D87794B9802B/VTSP:(2/1:23):0:8:4/vtsp_dsm_peer_event_cb:
*Apr 18 21:48:36.587: //88/D87794B9802B/VTSP:(2/1:23):0:8:4/act_generate_disc:
*Apr 18 21:48:36.587: //88/D87794B9802B/VTSP:(2/1:23):0:8:4/vtsp_cc_call_disconnected:
*Apr 18 21:48:36.587: //88/D87794B9802B/VTSP:(2/1:23):0:8:4/vtsp_dsm_bridge_status_cb:
```

The next event shows the call disconnect. There are several VTSP functions that follow the call disconnection to release and terminate the call.

```
*Apr 18 21:48:36.587: //88/D87794B9802B/VTSP:(2/1:23):0:8:4/act_disconnect:
*Apr 18 21:48:36.587: //88/D87794B9802B/VTSP:(2/1:23):0:8:4/vtsp_dsm_save_fax_config:
*Apr 18 21:48:36.587: //88/D87794B9802B/VTSP:(2/1:23):0:8:4/vtsp_dsm_set_fax_feat_param:
*Apr 18 21:48:36.595: //88/D87794B9802B/VTSP:(2/1:23):0:8:4/act_dsm_dsp_stats_complete:
*Apr 18 21:48:36.595: //88/D87794B9802B/VTSP:(2/1:23):0:8:4/act_wrelease_release:
*Apr 18 21:48:36.595: //88/D87794B9802B/VTSP:(2/1:23):0:8:4/vtsp_do_call_history:
*Apr 18 21:48:36.595: //88/D87794B9802B/VTSP:(2/1:23):0:8:4/vtsp_dsm_closed_cb:
*Apr 18 21:48:36.595: //88/D87794B9802B/VTSP:(2/1:23):0:-1:-1/act_terminate:
*Apr 18 21:48:36.595: //-1/D87794B9802B/VTSP:(2/1:23):0:-1:-1/vtsp_free_cdb:
```

Examples

```
Router# **debug voip vtsp inout**

voip vtsp inout debugging is on
*Apr 18 21:48:59.239: //-1/xxxxxxxxxxxx/VTSP:(2/1:23):-1:-1:-1/vtsp_allocate_cdb:
   CDB=0x65289878
*Apr 18 21:48:59.239: //-1/xxxxxxxxxxxx/VTSP:(2/1:23):-1:-1:-1/vtsp_do_call_setup_ind:
   Event=E_TSP_SETUP_IND
   Progress Indication=0, CarrierIDCode=, Info Trans Capability=0, Source Carrier ID=,
tg_label_flag=0
```

The following two events show the calling number, called number, and related parameters:

```
*Apr 18 21:48:59.239: //-1/xxxxxxxxxxxx/VTSP:(2/1:23):-1:-1:-1/vtsp_do_call_setup_ind:
   Calling Number=4085550111, TON=National, NPI=ISDN, Screening=User, Passed,
Presentation=Allowed
   CLIR=FALSE, CLID Transparent=FALSE, Null Originating Calling Number=FALSE, Calling
Translated=FALSE
*Apr 18 21:48:59.239: //-1/xxxxxxxxxxxx/VTSP:(2/1:23):-1:-1:-1/vtsp_do_call_setup_ind:
   Called Number=83103, TON=Unknown, NPI=Unknown
*Apr 18 21:48:59.239: //90/EC79A754802C/VTSP:(2/1:23):0:8:4/vtsp_insert_cdb:
*Apr 18 21:48:59.243: //90/EC79A754802C/VTSP:(2/1:23):0:8:4/act_proceeding:
   Progress Indication=0
*Apr 18 21:48:59.255: //90/EC79A754802C/VTSP:(2/1:23):0:8:4/act_progress:
   Progress Indication=8
*Apr 18 21:48:59.255: //90/EC79A754802C/VTSP:(2/1:23):0:8:4/vtsp_timer_stop:
   Timer Stop Time=538706
```

The following event shows fax parameters associated with the call:

```
*Apr 18 21:48:59.255: //90/EC79A754802C/VTSP:(2/1:23):0:8:4/vtsp_dsm_save_fax_config:
   Fax Relay=DISABLED - 'fax rate disabled' set (dial-peer)
   Primary Fax Protocol=IGNORE_FAX_RELAY, Fallback Fax Protocol=IGNORE_FAX_RELAY
   Fax Parameters Set By=Dialpeer, Peer=3600
*Apr 18 21:48:59.255: //90/EC79A754802C/VTSP:(2/1:23):0:8:4/vtsp_dsm_peer_event_cb:
   Event=E_DSM_CC_CAPS_IND
*Apr 18 21:48:59.255: //90/EC79A754802C/VTSP:(2/1:23):0:8:4/vtsp_dsm_peer_event_cb:
   Event=E_DSM_CC_CAPS_ACK
*Apr 18 21:48:59.255: //90/EC79A754802C/VTSP:(2/1:23):0:8:4/vtsp_dsm_feature_notify_cb:
   Feature ID=0, Feature Status=1
```

The following event shows the call connection:

```
*Apr 18 21:49:03.779: //90/EC79A754802C/VTSP:(2/1:23):0:8:4/act_connect:
   Progress Indication=2
*Apr 18 21:49:03.779: //90/EC79A754802C/VTSP:(2/1:23):0:8:4/vtsp_ring_noan_timer_stop:
   Timer Stop Time=539158
Router#
```

The following event shows the call disconnect:

```
*Apr 18 21:49:25.943: //90/EC79A754802C/VTSP:(2/1:23):0:8:4/act_generate_disc:
   Cause Value=16
*Apr 18 21:49:25.943: //90/EC79A754802C/VTSP:(2/1:23):0:8:4/vtsp_ring_noan_timer_stop:
   Timer Stop Time=541374
```

The following event shows that it was the calling party that initiated the call disconnect:

```
*Apr 18 21:49:25.943: //90/EC79A754802C/VTSP:(2/1:23):0:8:4/vtsp_set_release_source:
   Release Direction=PSTN, Release Source=Calling Party-PSTN
*Apr 18 21:49:25.943: //90/EC79A754802C/VTSP:(2/1:23):0:8:4/vtsp_timer:
   Timer Start Time=541374, Timer Value=15000(ms)
*Apr 18 21:49:25.943: //90/EC79A754802C/VTSP:(2/1:23):0:8:4/act_generate_disc:
   Return Code=0
*Apr 18 21:49:25.943: //90/EC79A754802C/VTSP:(2/1:23):0:8:4/act_disconnect:
   Cause Value=16, Previous Cause Value=16
*Apr 18 21:49:25.943: //90/EC79A754802C/VTSP:(2/1:23):0:8:4/vtsp_ring_noan_timer_stop:
   Timer Stop Time=541374
*Apr 18 21:49:25.943: //90/EC79A754802C/VTSP:(2/1:23):0:8:4/vtsp_dsm_save_fax_config:
   Fax Relay=DISABLED - 'fax rate disabled' set (dial-peer)
   Primary Fax Protocol=IGNORE_FAX_RELAY, Fallback Fax Protocol=IGNORE_FAX_RELAY
   Fax Parameters Set By=Dialpeer, Peer=3600
*Apr 18 21:49:25.951: //90/EC79A754802C/VTSP:(2/1:23):0:8:4/act_dsm_dsp_stats_complete:
*Apr 18 21:49:25.951: //90/EC79A754802C/VTSP:(2/1:23):0:8:4/vtsp_timer:
   Timer Start Time=541375, Timer Value=60000(ms)
```

The following two events show the call being released and the timer stopping:

```
*Apr 18 21:49:25.951: //90/EC79A754802C/VTSP:(2/1:23):0:8:4/act_wrelease_release:
*Apr 18 21:49:25.951: //90/EC79A754802C/VTSP:(2/1:23):0:8:4/vtsp_play_busy_timer_stop:
   Timer Stop Time=541375
*Apr 18 21:49:25.951: //90/EC79A754802C/VTSP:(2/1:23):0:8:4/vtsp_timer_stop:
   Timer Stop Time=541375
*Apr 18 21:49:25.951: //90/EC79A754802C/VTSP:(2/1:23):0:-1:-1/act_terminate:
*Apr 18 21:49:25.951: //90/EC79A754802C/VTSP:(2/1:23):0:-1:-1/vtsp_timer_stop:
   Timer Stop Time=541375
*Apr 18 21:49:25.951: //-1/EC79A754802C/VTSP:(2/1:23):0:-1:-1/vtsp_free_cdb:
   CDB=0x65289878
Router#
```

Examples

```
Router# debug voip vtsp tone
voip vtsp tone debugging is on
```

This output shows a wrong number dialed in the two-stage call to see the tone generated by the router.

```
*Apr 18 21:52:26.595: //98/657C0B9C8030/VTSP:(2/1:23):0:8:4/act_pre_con_disconnect:
   [Number Unobtainable]-Tone Played In Direction [Network]
```

Examples

This output shows the fax relay statistics.

```
Router# debug voip vtsp
VTSP:
  debug voip vtsp event is ON (filter is OFF)
  debug voip vtsp error software is ON
  debug voip vtsp error call is ON (filter is OFF)
  debug voip vtsp inout is ON (filter is OFF)
May  7 21:37:35.322 UTC: //-1/xxxxxxxxxxxx/VTSP:(3/1:D):-1:-1:-1/vtsp_allocate_cdb:
   CDB=0x63088050
May  7 21:37:35.322 UTC: //-1/xxxxxxxxxxxx/VTSP:(3/1:D):-1:-1:-1/vtsp_do_call_setup_ind:
   Event=E_TSP_SETUP_IND
   Progress Indication=3, CarrierIDCode=, Info Trans Capability=16, Source Carrier ID=,
tg_label_flag=0
May  7 21:37:35.322 UTC: //-1/xxxxxxxxxxxx/VTSP:(3/1:D):-1:-1:-1/vtsp_do_call_setup_ind:
   Called Number=41021, TON=National, NPI=ISDN
May  7 21:37:35.326 UTC: //-1/96A4C0C48006/VTSP:(3/1:D):0:-1:-1/vtsp_timer:
   Timer Start Time=1019501, Timer Value=180000(ms)
May  7 21:37:35.326 UTC: //-1/96A4C0C48006/VTSP:(3/1:D):0:-1:-1/vtsp_insert_cdb:
May  7 21:37:35.326 UTC: //-1/96A4C0C48006/VTSP:(3/1:D):0:0:0/vtsp_process_event:
   [state:S_SETUP_IND_PEND, event:E_VTSP_DSM_OPEN_SUCCESS]
May  7 21:37:35.326 UTC: //-1/96A4C0C48006/VTSP:(3/1:D):0:0:0/act_setup_ind_pend_success:
May  7 21:37:35.326 UTC: //-1/96A4C0C48006/VTSP:(3/1:D):0:0:0/vtsp_realloc_cdb:
   CDB=0x63088050
May  7 21:37:35.326 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:0:0/vtsp_insert_cdb:
May  7 21:37:35.326 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:0:0/vtsp_timer:
   Timer Start Time=1019501, Timer Value=180000(ms)
May  7 21:37:35.330 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:0:0/vtsp_process_event:
   [state:S_SETUP_INDICATED, event:E_CC_PROCEEDING]
May  7 21:37:35.330 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:0:0/act_proceeding:
   Progress Indication=0
May  7 21:37:35.330 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:0:0/vtsp_timer_stop:
   Timer Stop Time=1019502
May  7 21:37:35.394 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:0:0/vtsp_process_event:
   [state:S_PROCEEDING, event:E_CC_ALERT]
May  7 21:37:35.394 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:0:0/act_alert:
   Progress Indication=0, Signal Indication=1, Setup Progress Indication=3
May  7 21:37:35.394 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:0:0/vtsp_timer_stop:
   Timer Stop Time=1019508
May  7 21:37:35.398 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:0:0/act_alert:
   Progress Indication=0, Tone=
May  7 21:37:37.422 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:0:0/vtsp_dsm_save_fax_config:
   Fax Relay=ENABLED
   Primary Fax Protocol=T38_FAX_RELAY, Fallback Fax Protocol=NONE_FAX_RELAY
   Fax Parameters Set By=Dialpeer, Peer=2
May  7 21:37:37.426 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:0:0/vtsp_dsm_save_fax_config:
   LS Red=0, HS Red=0
May  7 21:37:37.426 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:0:0/vtsp_process_event:
   [state:S_ALERTING, event:E_CC_DO_CAPS_IND]
May  7 21:37:37.426 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:0:0/vtsp_dsm_peer_event_cb:
   Event=E_DSM_CC_CAPS_IND
May  7 21:37:37.426 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:0:0/vtsp_process_event:
   [state:S_ALERTING, event:E_CC_CAPS_IND]
May  7 21:37:37.426 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:0:0/vtsp_dsm_peer_event_cb:
   Event=E_DSM_CC_CAPS_ACK
May  7 21:37:37.426 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:0:0/vtsp_process_event:
   [state:S_ALERTING, event:E_CC_SERVICE_MSG]
May  7 21:37:37.426 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:0:0/act_service_msg_down:
May  7 21:37:37.426 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:0:0/vtsp_timer_stop:
   Timer Stop Time=1019711
May  7 21:37:37.426 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:0:0/vtsp_process_event:
   [state:S_ALERTING, event:E_CC_CONNECT]
May  7 21:37:37.426 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:0:0/act_alert_connect:
   Progress Indication=0
May  7 21:37:37.426 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:0:0/vtsp_ring_noan_timer_stop:
   Timer Stop Time=1019711
May  7 21:37:37.598 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:0:0/vtsp_dsm_feature_notify_cb:
   Feature ID=0, Feature Status=1
May  7 21:37:37.598 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:0:0/vtsp_dsm_feature_notify_cb:
   Feature ID=0, Feature Status=1
May  7 21:37:44.123 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:0:0/vtsp_dsm_peer_event_cb:
   Event=E_DSM_CC_MC_LOCAL_DNLD_DONE
May  7 21:37:44.123 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:0:0/vtsp_dsm_peer_event_cb:
   Event=E_DSM_CC_SET_FAX_MODE
May  7 21:37:44.123 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:0:0/vtsp_dsm_peer_event_cb:
   Event=E_DSM_CC_MC_LOCAL_DNLD_DONE
May  7 21:37:44.123 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:0:0/vtsp_dsm_peer_event_cb:
   Event=E_DSM_CC_SET_FAX_MODE
May  7 21:38:02.911 UTC: %ALIGN-3-SPURIOUS: Spurious memory access made at 0x6040A40C reading
 0x1
May  7 21:38:02.911 UTC: %ALIGN-3-TRACE: -Traceback= 6040A40C 60409198 603F8338 603F85F8
613EA398 619B369C 619B40BC 613DFEE4
May  7 21:38:02.915 UTC: %ALIGN-3-TRACE: -Traceback= 6040A54C 60409198 603F8338 603F85F8
613EA398 619B369C 619B40BC 613DFEE4
May  7 21:38:37.483 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:0:0/vtsp_process_event:
   [state:S_CONNECT, event:E_CC_CAPS_IND]
May  7 21:38:37.483 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:0:0/vtsp_dsm_peer_event_cb:
   Event=E_DSM_CC_CAPS_ACK
May  7 21:38:37.663 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:0:0/vtsp_process_event:
   [state:S_CONNECT, event:E_TSP_DISCONNECT_IND]
May  7 21:38:37.663 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:0:0/act_generate_disc:
   Cause Value=16
May  7 21:38:37.663 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:0:0/vtsp_timer_stop:
   Timer Stop Time=1025735
May  7 21:38:37.663 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:0:0/vtsp_ring_noan_timer_stop:
   Timer Stop Time=1025735
May  7 21:38:37.663 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:0:0/vtsp_set_release_source:
   Release Direction=PSTN, Release Source=Calling Party-PSTN
May  7 21:38:37.663 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:0:0/vtsp_timer:
   Timer Start Time=1025735, Timer Value=15000(ms)
May  7 21:38:37.663 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:0:0/act_generate_disc:
   Return Code=0
May  7 21:38:37.667 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:0:0/vtsp_process_event:
   [state:S_CONNECT, event:E_CC_DISCONNECT]
May  7 21:38:37.667 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:0:0/act_disconnect:
   Cause Value=16, Previous Cause Value=16
May  7 21:38:37.667 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:0:0/vtsp_ring_noan_timer_stop:
   Timer Stop Time=1025735
May  7 21:38:37.667 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:0:0/vtsp_dsm_save_fax_config:
   Fax Relay=ENABLED
   Primary Fax Protocol=T38_FAX_RELAY, Fallback Fax Protocol=NONE_FAX_RELAY
   Fax Parameters Set By=Dialpeer, Peer=2
May  7 21:38:37.667 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:0:0/vtsp_dsm_save_fax_config:
   LS Red=0, HS Red=0
May  7 21:38:37.691 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:0:0/vtsp_process_event:
   [state:S_WAIT_STATS, event:E_VTSP_DSM_STATS_COMPLETE]
May  7 21:38:37.691 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:0:0/act_dsm_dsp_stats_complete:
```

```
May  7 21:38:37.691 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:0:0/vtsp_timer:
   Timer Start Time=1025738, Timer Value=60000(ms)
May  7 21:38:37.691 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:0:0/vtsp_process_event:
   [state:S_WAIT_RELEASE, event:E_TSP_DISCONNECT_CONF]
May  7 21:38:37.691 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:0:0/act_wrelease_release:
May  7 21:38:37.691 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:0:0/vtsp_play_busy_timer_stop:
   Timer Stop Time=1025738
May  7 21:38:37.691 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:0:0/vtsp_timer_stop:
   Timer Stop Time=1025738
May  7 21:38:37.695 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:0:0/vtsp_process_event:
   [state:S_CLOSE_DSPRM, event:E_VTSP_DSM_STATS_COMPLETE]
May  7 21:38:37.695 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:0:0/vtsp_process_event:
   Unexpected EVENT [E_VTSP_DSM_STATS_COMPLETE] Received For STATE [S_CLOSE_DSPRM];
   Previous STATE [0.17 ]
May  7 21:38:37.695 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:-1:-1/vtsp_process_event:
   [state:S_CLOSE_DSPRM, event:E_VTSP_DSM_CLOSE_COMPLETE]
May  7 21:38:37.695 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:-1:-1/act_terminate:
May  7 21:38:37.695 UTC: //9/96A4C0C48006/VTSP:(3/1:D):0:-1:-1/vtsp_timer_stop:
   Timer Stop Time=1025738
May  7 21:38:37.695 UTC: //-1/96A4C0C48006/VTSP:(3/1:D):0:-1:-1/vtsp_free_cdb:
   CDB=0x63088050
```

Related Commands

Command	Description
debug voip dsm	Displays information about the DSM.
show debug	Displays which debug commands are enabled.
voice call debug	Allows configuration of the voice call debug output.

debug voip xcodemsp

To display debugging information from the Transcoding Media Service Processor and its related applications, use the **debug voip xcodemsp** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug voip xcodemsp

no debug voip xcodemsp

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.3(8)T	This command was introduced.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.

Examples

The following is sample output from the **debug voip xcodemsp** command:

```
Router# debug voip xcodemsp

XCODEMSP debugging is on
Router#
Router#
*Jul  8 18:36:53: xcmsp_call_setup_request:
*Jul  8 18:36:53: xcmsp_call_setup_request: callID 20, rscID 1 srvcDef.srvc_id 2
*Jul  8 18:36:53: xcmsp_bridge:
*Jul  8 18:36:53: xc_add_streams:
*Jul  8 18:36:53: xc_add_streams: stream id 1 added
*Jul  8 18:36:53: xc_add_streams: stream id 2 added
*Jul  8 18:36:53: xcmsp_bridge:
*Jul  8 18:36:53: xc_add_streams:
*Jul  8 18:36:53: xc_add_streams: stream id 5 added
*Jul  8 18:36:53: xc_add_streams: stream id 6 added
*Jul  8 18:36:53: xc_connect_bridges:
*Jul  8 18:36:53: xcmsp_dsmp_response
Router#
Router#
Router# show sccp connections

sess_id    conn_id    stype mode     codec   ripaddr         rport sport
16777223   16777905   xcode sendrecv g711a   1.4.177.1       16628 18870
16777223   16777921   xcode sendrecv g729ab  1.4.177.5       31318 18146
Total number of active session(s) 1, and connection(s) 2
Router#
*Jul  8 18:37:22: xcmsp_bridge_drop:
*Jul  8 18:37:22: xcmsp_bridge_drop: deleting stream id 5
*Jul  8 18:37:22: xcmsp_bridge_drop: deleting stream id 6
```

```
*Jul  8 18:37:22: xcmsp_dsmp_response
*Jul  8 18:37:22: xcmsp_dsmp_response: DSMP_DISCONNECTED
*Jul  8 18:37:22: xcmsp_bridge_drop:
*Jul  8 18:37:22: xcmsp_bridge_drop: deleting stream id 1
*Jul  8 18:37:22: xcmsp_bridge_drop: deleting stream id 2
*Jul  8 18:37:22: xcmsp_call_disconnect:
```

debug vpdn

To troubleshoot Layer 2 Forwarding (L2F) or Layer 2 Tunnel Protocol (L2TP) virtual private dial-up network (VPDN) tunneling events and infrastructure, use the **debug vpdn** command in privileged EXEC mode. To disable the debugging of L2TP VPDN tunneling events and infrastructure, use the **no** form of this command.

Note Effective with Cisco IOS Release 12.4(11)T, the L2F protocol is not supported in Cisco IOS software.

Cisco IOS Release 12.2(33)XNA and Later Releases

debug vpdn {**call** {**event**| **fsm**}| **authorization** {**error**| **event**}| **error**| **event** [**disconnect** [**traceback**]]| **l2tp-sequencing**| **l2x-data**| **l2x-errors**| **l2x-events**| **l2x-packets**| **message**| **packet** [**detail**| **errors**]| **sss** {**error**| **event**| **fsm**}| **subscriber** {**error**| **event**| **fsm**}}

no debug vpdn {**call** {**event**| **fsm**}| **authorization** {**error**| **event**}| **error**| **event** [**disconnect** [**traceback**]]| **l2tp-sequencing**| **l2x-data**| **l2x-errors**| **l2x-events**| **l2x-packets**| **message**| **packet** [**detail**| **errors**]| **sss** {**error**| **event**| **fsm**}| **subscriber** {**error**| **event**| **fsm**}}

Cisco IOS Releases Prior to 12.2(33)XNA

debug vpdn {**call** {**event**| **fsm**}| **authorization** {**error**| **event**}| **error**| **event** [**disconnect**]| **l2tp-sequencing**| **l2x-data**| **l2x-errors**| **l2x-events**| **l2x-packets**| **message**| **packet** [**detail**| **errors**]| **sss** {**error**| **event**| **fsm**}| **subscriber** {**error**| **event**| **fsm**}}

no debug vpdn {**call** {**event**| **fsm**}| **authorization** {**error**| **event**}| **error**| **event** [**disconnect**]| **l2tp-sequencing**| **l2x-data**| **l2x-errors**| **l2x-events**| **l2x-packets**| **message**| **packet** [**detail**| **errors**]| **sss** {**error**| **event**| **fsm**}| **subscriber** {**error**| **event**| **fsm**}}

Syntax Description

call event	Displays significant events in the VPDN call manager.
call fsm	Displays significant events in the VPDN call manager finite state machine (FSM).
authorization error	Displays authorization errors.
authorization event	Displays authorization events.
error	Displays VPDN errors.
event	Displays VPDN events.
disconnect	(Optional) Displays VPDN disconnect events. **Note** The disconnect keyword is required in Cisco IOS Release 12.2(33)XNA and later releases.
traceback	(Optional) Displays traceback messages that provide reasons for VPDN disconnect.

l2tp-sequencing	Displays significant events related to L2TP sequence numbers such as mismatches, resend queue flushes, and drops.
l2x-data	Displays errors that occur in data packets.
l2x-errors	Displays errors that occur in protocol-specific conditions.
l2x-events	Displays events resulting from protocol-specific conditions.
l2x-packets	Displays detailed information about control packets in protocol-specific conditions.
message	Displays VPDN interprocess messages.
packet	Displays information about VPDN packets.
detail	(Optional) Displays detailed packet information, including packet dumps.
errors	(Optional) Displays errors that occur in packet processing.
sss error	Displays debug information about VPDN Subscriber Service Switch (SSS) errors.
sss event	Displays debug information about VPDN SSS events.
sss fsm	Displays debug information about the VPDN SSS FSM.
subscriber error	Displays debug information about VPDN Subscriber errors.
subscriber event	Displays debug information about VPDN Subscriber events.
subscriber fsm	Displays debug information about the VPDN Subscriber FSM.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
11.2 T	This command was introduced.

Release	Modification
12.0(5)T	This command was modified. Support was added for L2TP debugging messages. The **l2tp-sequencing** and **error** keywords were added. The **l2f-errors**, **l2f-events**, and **l2f-packets** keywords were changed to **l2x-errors**, **l2x-events**, and **l2x-packets**.
12.2(4)T	This command was modified. The **call**, **event**, **fsm**, and **message** keywords were added.
12.2(11)T	This command was modified. The **detail** keyword was added.
12.0(23)S	This command was integrated into Cisco IOS Release 12.0(23)S.
12.2(13)T	This command was modified. The **sss**, **error**, **event**, and **fsm** keywords were added.
12.3(14)T	This command was modified. Support was added to decode the outbound control channel authentication events.
12.0(31)S	This command was modified. The output was enhanced to display messages about control channel authentication events.
12.2(27)SBC	This command was modified. Support for enhanced display of messages about control channel authentication events was added.
12.2(28)SB	This command was modified. Support for the display of messages about congestion avoidance events was added.
12.2(31)SB	This command was modified. Support was added to decode the outbound control channel authentication events.
12.4(15)T	This command was modified. The **authorization**, **error**, and **event** keywords were added.
12.2(33)XNA	This command was modified. The **traceback** keyword was added.
12.4(20)T	This command was modified. The **subscriber** keyword was added and the **sss** keyword was removed.
Cisco IOS XE Release 2.6	This command was modified. Authentication failure messages for L2TPv3 were added.

Usage Guidelines

The **debug vpdn packet** and **debug vpdn packet detail** commands generate several debug operations per packet. Depending on the L2TP traffic pattern, these commands may cause the CPU load to increase to a high level that impacts performance.

Examples

Examples

The following example shows the VPDN configuration on a network access server (NAS):

```
vpdn-group 1
 request-dialin
  protocol l2f
  domain example.com
 initiate-to ip 172.17.33.125
username nas1 password nas1
```

The following is sample output from the **debug vpdn event** command on a NAS when an L2F tunnel is brought up and Challenge Handshake Authentication Protocol (CHAP) authentication of the tunnel succeeds:

```
Device# debug vpdn event

%LINK-3-UPDOWN: Interface Async6, changed state to up
*Mar 2 00:26:05.537: looking for tunnel –– example.com ––
*Mar 2 00:26:05.545: Async6 VPN Forwarding...
*Mar 2 00:26:05.545: Async6 VPN Bind interface direction=1
*Mar 2 00:26:05.553: Async6 VPN vpn_forward_user user6@example.com is forwarded
%LINEPROTO-5-UPDOWN: Line protocol on Interface Async6, changed state to up
*Mar 2 00:26:06.289: L2F: Chap authentication succeeded for nas1.
```

The following is sample output from the **debug vpdn event** command on a NAS when the L2F tunnel is brought down normally:

```
Device# debug vpdn event

%LINEPROTO-5-UPDOWN: Line protocol on Interface Async6, changed state to down
%LINK-5-CHANGED: Interface Async6, changed state to reset
*Mar 2 00:27:18.865: Async6 VPN cleanup
*Mar 2 00:27:18.869: Async6 VPN reset
*Mar 2 00:27:18.873: Async6 VPN Unbind interface
%LINK-3-UPDOWN: Interface Async6, changed state to down
```

The table below describes the significant fields shown in the two previous displays. The output describes normal operations when an L2F tunnel is brought up or down on a NAS.

Table 67: debug vpdn event Field Descriptions for the NAS

Field	Description
Asynchronous interface coming up	
%LINK-3-UPDOWN: Interface Async6, changed state to up	Asynchronous interface 6 came up.
looking for tunnel –– example.com –– Async6 VPN Forwarding...	Domain name is identified.

Field	Description
Async6 VPN Bind interface direction=1	Tunnel is bound to the interface. These are the direction values: • 1—From the NAS to the tunnel server • 2—From the tunnel server to the NAS
Async6 VPN vpn_forward_user user6@example.com is forwarded	Tunnel for the specified user and domain name is forwarded.
%LINEPROTO-5-UPDOWN: Line protocol on Interface Async6, changed state to up	Line protocol is up.
L2F: Chap authentication succeeded for nas1.	Tunnel was authenticated with the tunnel password nas1.
Virtual access interface coming down	
%LINEPROTO-5-UPDOWN: Line protocol on Interface Async6, changed state to down	Normal operation when the virtual access interface is taken down.
Async6 VPN cleanup Async6 VPN reset Async6 VPN Unbind interface	Normal cleanup operations performed when the line or virtual access interface goes down.

Examples

The following example shows the VPDN configuration on a tunnel server, which uses *nas1* as the tunnel name and the tunnel authentication name. The tunnel authentication name can be entered in a user's file on an authentication, authorization, and accounting (AAA) server and used to define authentication requirements for the tunnel.

```
vpdn-group 1
 accept-dialin
  protocol l2f
  virtual-template 1
 terminate-from hostname nas1
```

The following is sample output from the **debug vpdn event** command on a tunnel server when an L2F tunnel is brought up successfully:

```
Device# debug vpdn event

L2F: Chap authentication succeeded for nas1.
Virtual-Access3 VPN Virtual interface created for user6@example.com
Virtual-Access3 VPN Set to Async interface
Virtual-Access3 VPN Clone from Vtemplate 1 block=1 filterPPP=0
%LINK-3-UPDOWN: Interface Virtual-Access3, changed state to up
Virtual-Access3 VPN Bind interface direction=2
Virtual-Access3 VPN PPP LCP accepted sent & rcv CONFACK
%LINEPROTO-5-UPDOWN: Line protocol on Interface Virtual-Access3, changed state to up
```

The following is sample output from the **debug vpdn event** command on a tunnel server when an L2F tunnel is brought down normally:

```
Device# debug vpdn event

%LINK-3-UPDOWN: Interface Virtual-Access3, changed state to down
Virtual-Access3 VPN cleanup
Virtual-Access3 VPN reset
Virtual-Access3 VPN Unbind interface
Virtual-Access3 VPN reset
%LINEPROTO-5-UPDOWN: Line protocol on Interface Virtual-Access3, changed state to down
```

The table below describes the fields shown in two previous outputs. The output describes normal operations when an L2F tunnel is brought up or down on a tunnel server.

Table 68: debug vpdn event Field Descriptions

Field	Description
L2F: Chap authentication succeeded for nas1.	PPP CHAP authentication status for the tunnel named *nas1*.
Virtual-Access3 VPN Virtual interface created for user6@example.com	Virtual access interface was set up on a tunnel server for the user user6@example.com.
Virtual-Access3 VPN Set to Async interface	Virtual access interface 3 was set to asynchronous for character-by-character transmission.
Virtual-Access3 VPN Clone from Vtemplate 1 block=1 filterPPP=0	Virtual template 1 was applied to virtual access interface 3.
%LINK-3-UPDOWN: Interface Virtual-Access3, changed state to up	Link status is set to up.
Virtual-Access3 VPN Bind interface direction=2	Tunnel is bound to the interface. These are the direction values: • 1—From the NAS to the tunnel server • 2—From the tunnel server to the NAS
Virtual-Access3 VPN PPP LCP accepted sent & rcv CONFACK	PPP link control protocol (LCP) configuration settings (negotiated between the remote client and the NAS) were copied to the tunnel server and acknowledged.
%LINEPROTO-5-UPDOWN: Line protocol on Interface Virtual-Access3, changed state to up	Line protocol is up; the line can be used.
%LINK-3-UPDOWN: Interface Virtual-Access3, changed state to down	Virtual access interface is coming down.

Field	Description
Virtual-Access3 VPN cleanup Virtual-Access3 VPN reset Virtual-Access3 VPN Unbind interface Virtual-Access3 VPN reset	Device is performing normal cleanup operations when a virtual access interface used for an L2F tunnel comes down.
%LINEPROTO-5-UPDOWN: Line protocol on Interface Virtual-Access3, changed state to down	Line protocol is down for virtual access interface 3; the line cannot be used.

Examples

The following is sample output from the **debug vpdn event disconnect traceback** command on a tunnel server when an L2TP Network Server (LNS) tunnel session is disconnected:

```
Device# debug vpdn event disconnect traceback

*Aug  8 07:13:56.795: VPDN Vi2.1 disconnect (L2X) IETF: 18/host-request Ascend: 66/VPDN
Local PPP Disconnect
*Aug  8 07:13:56.795: VPDN Vi2.1 vpdn shutdown session, result=2, error=6, vendor_err=0,
syslog_error_code=2, syslog_key_type=1
*Aug  8 07:13:56.795: VPDN Vi2.1 VPDN/AAA: accounting stop sent
*Aug  8 07:13:56.795: VPDN Vi2.1 Unbinding session from idb, informational traceback:
*Aug  8 07:13:56.795: -Traceback= DFFFE7z 30EE221z 30DFBA8z 30E2F26z 30DF1DCz 30DF12Fz
1F0170Fz 1F015A1z 31E695Bz 31E674Dz 1F019F6z
*Aug  8 07:13:56.795: Vi2.1 VPDN: Resetting interface, informational traceback below:
LNS#
*Aug  8 07:13:56.795: -Traceback= DFFFE7z 30EDE74z 30EE2D4z 37996B7z 37A3019z 30EE408z
30DFBB3z 30E2F26z 30DF1DCz 30DF12Fz 1F0170Fz 1F015A1z 31E695Bz 31E674Dz 1F019F6z
```

Examples

The following is sample output from the **debug vpdn event** command on the NAS when an L2TP tunnel is brought up successfully:

```
Device# debug vpdn event

20:19:17: L2TP: I SCCRQ from ts1 tnl 8
20:19:17: L2X: Never heard of ts1
20:19:17: Tnl 7 L2TP: New tunnel created for remote ts1, address 172.21.9.4
20:19:17: Tnl 7 L2TP: Got a challenge in SCCRQ, ts1
20:19:17: Tnl 7 L2TP: Tunnel state change from idle to wait-ctl-reply
20:19:17: Tnl 7 L2TP: Got a Challenge Response in SCCCN from ts1
20:19:17: Tnl 7 L2TP: Tunnel Authentication success
20:19:17: Tnl 7 L2TP: Tunnel state change from wait-ctl-reply to established
20:19:17: Tnl 7 L2TP: SM State established
20:19:17: Tnl/Cl 7/1 L2TP: Session FS enabled
20:19:17: Tnl/Cl 7/1 L2TP: Session state change from idle to wait-for-tunnel
20:19:17: Tnl/Cl 7/1 L2TP: New session created
20:19:17: Tnl/Cl 7/1 L2TP: O ICRP to ts1 8/1
20:19:17: Tnl/Cl 7/1 L2TP: Session state change from wait-for-tunnel to wait-connect
20:19:17: Tnl/Cl 7/1 L2TP: Session state change from wait-connect to established
20:19:17: Vi1 VPDN: Virtual interface created for example1@example.com
20:19:17: Vi1 VPDN: Set to Async interface
20:19:17: Vi1 VPDN: Clone from Vtemplate 1 filterPPP=0 blocking
20:19:18: %LINK-3-UPDOWN: Interface Virtual-Access1, changed state to up
20:19:18: Vi1 VPDN: Bind interface direction=2
20:19:18: Vi1 VPDN: PPP LCP accepting rcv CONFACK
```
20:19:19: %LINEPROTO-5-UPDOWN: Line protocol on Interface Virtual-Access1, changed state to up

Examples

The following is sample output from the **debug vpdn event** command on a tunnel server when an L2TP tunnel is brought up successfully:

```
Device# debug vpdn event

20:47:33: %LINK-3-UPDOWN: Interface Async7, changed state to up
20:47:35: As7 VPDN: Looking for tunnel —— example.com ——
20:47:35: As7 VPDN: Get tunnel info for example.com with NAS nas1, IP 172.21.9.13
20:47:35: As7 VPDN: Forward to address 172.21.9.13
20:47:35: As7 VPDN: Forwarding...
20:47:35: As7 VPDN: Bind interface direction=1
20:47:35: Tnl/Cl 8/1 L2TP: Session FS enabled
20:47:35: Tnl/Cl 8/1 L2TP: Session state change from idle to wait-for-tunnel
20:47:35: As7 8/1 L2TP: Create session
20:47:35: Tnl 8 L2TP: SM State idle
20:47:35: Tnl 8 L2TP: Tunnel state change from idle to wait-ctl-reply
20:47:35: Tnl 8 L2TP: SM State wait-ctl-reply
20:47:35: As7 VPDN: example1@example.com is forwarded
20:47:35: Tnl 8 L2TP: Got a challenge from remote peer, nas1
20:47:35: Tnl 8 L2TP: Got a response from remote peer, nas1
20:47:35: Tnl 8 L2TP: Tunnel Authentication success
20:47:35: Tnl 8 L2TP: Tunnel state change from wait-ctl-reply to established
20:47:35: Tnl 8 L2TP: SM State established
20:47:35: As7 8/1 L2TP: Session state change from wait-for-tunnel to wait-reply
20:47:35: As7 8/1 L2TP: Session state change from wait-reply to established
20:47:36: %LINEPROTO-5-UPDOWN: Line protocol on Interface Async7, changed state to up
```

Examples

The following is sample output from the **debug vpdn l2x-events** command on the NAS when an L2F tunnel is brought up successfully:

```
Device# debug vpdn l2x-events

%LINK-3-UPDOWN: Interface Async6, changed state to up
*Mar 2 00:41:17.365: L2F Open UDP socket to 172.21.9.26
*Mar 2 00:41:17.385: L2F_CONF received
*Mar 2 00:41:17.389: L2F Removing resend packet (type 1)
*Mar 2 00:41:17.477: L2F_OPEN received
*Mar 2 00:41:17.489: L2F Removing resend packet (type 2)
*Mar 2 00:41:17.493: L2F building nas2gw_mid0
%LINEPROTO-5-UPDOWN: Line protocol on Interface Async6, changed state to up
*Mar 2 00:41:18.613: L2F_OPEN received
*Mar 2 00:41:18.625: L2F Got a MID management packet
*Mar 2 00:41:18.625: L2F Removing resend packet (type 2)
*Mar 2 00:41:18.629: L2F MID synced NAS/HG Clid=7/15 Mid=1 on Async6
```

The following is sample output from the **debug vpdn l2x-events** command on a NAS when an L2F tunnel is brought down normally:

```
Device# debug vpdn l2x-events

%LINEPROTO-5-UPDOWN: Line protocol on Interface Async6, changed state to down
%LINK-5-CHANGED: Interface Async6, changed state to reset
*Mar 2 00:42:29.213: L2F_CLOSE received
*Mar 2 00:42:29.217: L2F Destroying mid
*Mar 2 00:42:29.217: L2F Removing resend packet (type 3)
*Mar 2 00:42:29.221: L2F Tunnel is going down!
*Mar 2 00:42:29.221: L2F Initiating tunnel shutdown.
*Mar 2 00:42:29.225: L2F_CLOSE received
*Mar 2 00:42:29.229: L2F_CLOSE received
*Mar 2 00:42:29.229: L2F Got closing for tunnel
*Mar 2 00:42:29.233: L2F Removing resend packet
*Mar 2 00:42:29.233: L2F Closed tunnel structure
%LINK-3-UPDOWN: Interface Async6, changed state to down
*Mar 2 00:42:31.793: L2F Closed tunnel structure
*Mar 2 00:42:31.793: L2F Deleted inactive tunnel
```

The table below describes the fields shown in the displays.

Table 69: debug vpdn l2x-events Field Descriptions—NAS

Field	Descriptions
%LINK-3-UPDOWN: Interface Async6, changed state to up	Asynchronous interface came up normally.
L2F Open UDP socket to 172.21.9.26	L2F opened a User Datagram Protocol (UDP) socket to the tunnel server IP address.
L2F_CONF received	L2F_CONF signal was received. When sent from the tunnel server to the NAS, an L2F_CONF indicates the tunnel server's recognition of the tunnel creation request.
L2F Removing resend packet (type ...)	Removing the resend packet for the L2F management packet. There are two resend packets that have different meanings in different states of the tunnel.
L2F_OPEN received	L2F_OPEN management message was received, indicating that the tunnel server accepted the NAS configuration of an L2F tunnel.
L2F building nas2gw_mid0	L2F is building a tunnel between the NAS and the tunnel server using the multiplex ID (MID) MID0.
%LINEPROTO-5-UPDOWN: Line protocol on Interface Async6, changed state to up	Line protocol came up. Indicates whether the software processes that handle the line protocol regard the interface as usable.
L2F_OPEN received	L2F_OPEN management message was received, indicating that the tunnel server accepted the NAS configuration of an L2F tunnel.
L2F Got a MID management packet	MID management packets are used to communicate between the NAS and the tunnel server.
L2F MID synced NAS/HG Clid=7/15 Mid=1 on Async6	L2F synchronized the client IDs on the NAS and the tunnel server, respectively. An MID is assigned to identify this connection in the tunnel.
Tunnel coming down	
%LINEPROTO-5-UPDOWN: Line protocol on Interface Async6, changed state to down	Line protocol came down. Indicates whether the software processes that handle the line protocol regard the interface as usable.

Field	Descriptions
%LINK-5-CHANGED: Interface Async6, changed state to reset	Interface was marked as reset.
L2F_CLOSE received	NAS received a request to close the tunnel.
L2F Destroying mid	Connection identified by the MID is being taken down.
L2F Tunnel is going down!	Advisory message about impending tunnel shutdown.
L2F Initiating tunnel shutdown.	Tunnel shutdown has started.
L2F_CLOSE received	NAS received a request to close the tunnel.
L2F Got closing for tunnel	NAS began tunnel closing operations.
%LINK-3-UPDOWN: Interface Async6, changed state to down	Asynchronous interface was taken down.
L2F Closed tunnel structure	NAS closed the tunnel.
L2F Deleted inactive tunnel	Now-inactivated tunnel was deleted.

Examples

The following is sample output from the **debug vpdn l2x-events** command on a tunnel server when an L2F tunnel is created:

```
Device# debug vpdn l2x-events

L2F_CONF received
L2F Creating new tunnel for nas1
L2F Got a tunnel named nas1, responding
L2F Open UDP socket to 172.21.9.25
L2F_OPEN received
L2F Removing resend packet (type 1)
L2F_OPEN received
L2F Got a MID management packet
%LINK-3-UPDOWN: Interface Virtual-Access1, changed state to up
%LINEPROTO-5-UPDOWN: Line protocol on Interface Virtual-Access1, changed state to up
```

The following is sample output from the **debug vpdn l2x-events** command on a tunnel server when the L2F tunnel is brought down normally:

```
Device# debug vpdn l2x-events

L2F_CLOSE received
L2F Destroying mid
L2F Removing resend packet (type 3)
L2F Tunnel is going down!
L2F Initiating tunnel shutdown.
%LINK-3-UPDOWN: Interface Virtual-Access1, changed state to down
L2F_CLOSE received
L2F Got closing for tunnel
L2F Removing resend packet
L2F Removing resend packet
L2F Closed tunnel structure
```

```
L2F Closed tunnel structure
L2F Deleted inactive tunnel
%LINEPROTO-5-UPDOWN: Line protocol on Interface Virtual-Access1, changed state to down
```

The table below describes the significant fields shown in the displays.

Table 70: debug vpdn l2x-events Field Descriptions—Tunnel Server

Field	Description
L2F_CONF received	L2F configuration is received from the NAS. When sent from a NAS to a tunnel server, the L2F_CONF is the initial packet in the conversation.
L2F Creating new tunnel for nas1	Tunnel named nas1 is being created.
L2F Got a tunnel named nas1, responding	Tunnel server is responding.
L2F Open UDP socket to 172.21.9.25	Opening a socket to the NAS IP address.
L2F_OPEN received	L2F_OPEN management message was received, indicating that the NAS is opening an L2F tunnel.
L2F Removing resend packet (type 1)	Removing the resend packet for the L2F management packet. The two resend packet types have different meanings in different states of the tunnel.
L2F Got a MID management packet	L2F MID management packets are used to communicate between the NAS and the tunnel server.
%LINK-3-UPDOWN: Interface Virtual-Access1, changed state to up	Tunnel server is bringing up virtual access interface 1 for the L2F tunnel.
%LINEPROTO-5-UPDOWN: Line protocol on Interface Virtual-Access1, changed state to up	Line protocol is up. The line can be used.
Tunnel coming down	
L2F_CLOSE received	NAS or tunnel server received a request to close the tunnel.
L2F Destroying mid	Connection identified by the MID is being taken down.
L2F Removing resend packet (type 3)	Removing the resend packet for the L2F management packet. There are two resend packets that have different meanings in different states of the tunnel.
L2F Tunnel is going down! L2F Initiating tunnel shutdown.	Device is performing normal operations when a tunnel is coming down.

Field	Description
%LINK-3-UPDOWN: Interface Virtual-Access1, changed state to down	The virtual access interface is coming down.
L2F_CLOSE received L2F Got closing for tunnel L2F Removing resend packet L2F Removing resend packet L2F Closed tunnel structure L2F Closed tunnel structure L2F Deleted inactive tunnel	Device is performing normal cleanup operations when the tunnel is being brought down.
%LINEPROTO-5-UPDOWN: Line protocol on Interface Virtual-Access1, changed state to down	Line protocol is down; virtual access interface 1 cannot be used.

Examples

The following partial example of the **debug vpdn l2x-events** command is useful for monitoring a network running the L2TP Congestion Avoidance feature. The report shows that the congestion window (Cwnd) has been reset to 1 because of packet retransmissions:

```
Device# debug vpdn l2x-events
.
.
.
*Jul 15 19:02:57.963:  Tnl 47100 L2TP: Congestion Control event received is retransmission
*Jul 15 19:02:57.963:  Tnl 47100 L2TP: Congestion Window size, Cwnd 1
*Jul 15 19:02:57.963:  Tnl 47100 L2TP: Slow Start threshold, Ssthresh 2
*Jul 15 19:02:57.963:  Tnl 47100 L2TP: Remote Window size, 500
*Jul 15 19:02:57.963:  Tnl 47100 L2TP: Control channel retransmit delay set to 4 seconds
*Jul 15 19:03:01.607:  Tnl 47100 L2TP: Update ns/nr, peer ns/nr 2/5, our ns/nr 5/2
```

The following partial example shows that traffic has been restarted with L2TP congestion avoidance throttling traffic:

```
Device# debug vpdn l2x-events
.
.
.
*Jul 15 14:45:16.123:  Tnl 30597 L2TP: Control channel retransmit delay set to 2 seconds
*Jul 15 14:45:16.123:  Tnl 30597 L2TP: Tunnel state change from idle to wait-ctl-reply
*Jul 15 14:45:16.131:  Tnl 30597 L2TP: Congestion Control event received is positive
acknowledgement
*Jul 15 14:45:16.131:  Tnl 30597 L2TP: Congestion Window size, Cwnd 2
*Jul 15 14:45:16.131:  Tnl 30597 L2TP: Slow Start threshold, Ssthresh 500
*Jul 15 14:45:16.131:  Tnl 30597 L2TP: Remote Window size, 500
*Jul 15 14:45:16.131:  Tnl 30597 L2TP: Congestion Ctrl Mode is Slow Start
```

The table below describes the significant fields shown in the displays. See RFC 2661 for more details about the information in the reports for L2TP congestion avoidance.

Table 71: debug vpdn l2x-events Field Descriptions—L2TP Congestion Avoidance

Field	Description
Control channel retransmit delay set to ...	Indicates the current value set for the retransmit delay.
Tunnel state...	Indicates the tunnel's current Control Connection State, per RFC 2661.
Congestion Control event received is...	Indicates the received congestion control event. • Retransmission—Indicates packet retransmission has been detected in the resend queue. • Positive acknowledgement—Indicates that a packet was received and acknowledged by the peer tunnel endpoint.
Congestion Window size, Cwnd 2	Current size of the Cwnd.
Slow Start threshold, Ssthresh 500	Current value of the slow start threshold (Ssthresh).
Remote Window size, 500	Size of the advertised receive window configured on the remote peer with the **l2tp tunnel receive-window** command.
Congestion Ctrl Mode is...	Indicates whether the device is operating in Slow Start or Congestion Avoidance mode.
Update ns/nr, peer ns/nr 2/5, our ns/nr 5/2	See RFC 2661.

Examples

The following is sample output from the **debug vpdn error** command on a NAS when the L2F tunnel is not set up:

```
Device# debug vpdn error

%LINEPROTO-5-UPDOWN: Line protocol on Interface Async1, changed state to down
%LINK-5-CHANGED: Interface Async1, changed state to reset
%LINK-3-UPDOWN: Interface Async1, changed state to down
%LINK-3-UPDOWN: Interface Async1, changed state to up
%LINEPROTO-5-UPDOWN: Line protocol on Interface Async1, changed state to up
VPDN tunnel management packet failed to authenticate
VPDN tunnel management packet failed to authenticate
```

The table below describes the significant fields shown in the display.

Table 72: debug vpdn error Field Descriptions for the NAS

Field	Description
%LINEPROTO-5-UPDOWN: Line protocol on Interface Async1, changed state to down	Line protocol on the asynchronous interface went down.
%LINK-5-CHANGED: Interface Async1, changed state to reset	Asynchronous interface 1 was reset.
%LINK-3-UPDOWN: Interface Async1, changed state to down %LINK-3-UPDOWN: Interface Async1, changed state to up	Link from asynchronous interface 1 link went down and then came back up.
%LINEPROTO-5-UPDOWN: Line protocol on Interface Async1, changed state to up	Line protocol on the asynchronous interface came back up.
VPDN tunnel management packet failed to authenticate	Tunnel authentication failed. This is the most common VPDN error. **Note** Verify the password for the NAS and the tunnel server name. If you store the password on an AAA server, you can use the **debug aaa authentication** command.

The following is sample output from the **debug vpdn l2x-errors** command:

```
Device# debug vpdn l2x-errors

%LINK-3-UPDOWN: Interface Async1, changed state to up
L2F Out of sequence packet 0 (expecting 0)
L2F Tunnel authentication succeeded for example.com
 L2F Received a close request for a non-existent mid
 L2F Out of sequence packet 0 (expecting 0)
 L2F packet has bogus1 key 1020868 D248BA0F
L2F packet has bogus1 key 1020868 D248BA0F
```

The table below describes the significant fields shown in the display.

Table 73: debug vpdn l2x-errors Field Descriptions

Field	Description
%LINK-3-UPDOWN: Interface Async1, changed state to up	The line protocol on the asynchronous interface came up.
L2F Out of sequence packet 0 (expecting 0)	Packet was expected to be the first in a sequence starting at 0, but an invalid sequence number was received.
L2F Tunnel authentication succeeded for example.com	Tunnel was established from the NAS to the tunnel server, example.com.

Field	Description
L2F Received a close request for a non-existent mid	Multiplex ID was not used previously; cannot close the tunnel.
L2F Out of sequence packet 0 (expecting 0)	Packet was expected to be the first in a sequence starting at 0, but an invalid sequence number was received.
L2F packet has bogus1 key 1020868 D248BA0F	Value based on the authentication response given to the peer during tunnel creation. This packet, in which the key does not match the expected value, must be discarded.
L2F packet has bogus1 key 1020868 D248BA0F	Another packet was received with an invalid key value. The packet must be discarded.

Examples

The following is sample output from the **debug vpdn l2x-packets** command on a NAS. This example displays a trace for a **ping** command.

```
Device# debug vpdn l2x-packets

L2F SENDING (17): D0 1 1 10 0 0 0 4 0 11 0 0 81 94 E1 A0 4
L2F header flags: 53249 version 53249 protocol 1 sequence 16 mid 0 cid 4
length 17 offset 0 key 1701976070
L2F RECEIVED (17): D0 1 1 10 0 0 0 4 0 11 0 0 65 72 18 6 5
L2F SENDING (17): D0 1 1 11 0 0 0 4 0 11 0 0 81 94 E1 A0 4
L2F header flags: 53249 version 53249 protocol 1 sequence 17 mid 0 cid 4
length 17 offset 0 key 1701976070
L2F RECEIVED (17): D0 1 1 11 0 0 0 4 0 11 0 0 65 72 18 6 5
L2F header flags: 57345 version 57345 protocol 2 sequence 0 mid 1 cid 4
length 32 offset 0 key 1701976070
L2F-IN Output to Async1 (16): FF 3 C0 21 9 F 0 C 0 1D 41 AD FF 11 46 87
L2F-OUT (16): FF 3 C0 21 A F 0 C 0 1A C9 BD FF 11 46 87
L2F header flags: 49153 version 49153 protocol 2 sequence 0 mid 1 cid 4
length 32 offset 0 key -2120949344
L2F-OUT (101): 21 45 0 0 64 0 10 0 0 FF 1 B9 85 1 0 0 3 1 0 0 1 8 0 62 B1
0 0 C A8 0 0 0 0 0 11 E E0 AB CD AB CD AB CD AB CD AB CD AB CD AB CD AB CD
AB CD AB CD AB CD AB CD AB CD AB CD AB CD AB CD AB CD AB CD AB CD AB CD AB
CD AB CD AB CD AB CD AB CD AB CD AB CD AB CD AB CD AB CD AB CD AB CD
L2F header flags: 49153 version 49153 protocol 2 sequence 0 mid 1 cid 4
length 120 offset 3 key -2120949344
L2F header flags: 49153 version 49153 protocol 2 sequence 0 mid 1 cid 4
length 120 offset 3 key 1701976070
L2F-IN Output to Async1 (101): 21 45 0 0 64 0 10 0 0 FF 1 B9 85 1 0 0 1 1 0
0 3 0 0 6A B1 0 0 C A8 0 0 0 0 0 11 E E0 AB CD AB CD AB CD AB CD AB CD AB CD
AB CD AB CD AB CD AB CD AB CD AB CD AB CD AB CD AB CD AB CD AB CD AB CD AB
CD AB CD AB CD AB CD AB CD AB CD AB CD AB CD AB CD AB CD AB CD AB CD AB CD
```

The table below describes the significant fields shown in the display.

Table 74: debug vpdn l2x-packets Field Descriptions

Field	Description
L2F SENDING (17)	Number of bytes being sent. The first set of “SENDING”...“RECEIVED” lines displays L2F keepalive traffic. The second set displays L2F management data.
L2F header flags:	Version and flags, in decimal.
version 53249	Version number.
protocol 1	Protocol for negotiation of the point-to-point link between the NAS and the tunnel server is always 1, indicating L2F management.
sequence 16	Sequence numbers start at 0. Each subsequent packet is sent with the next increment of the sequence number. The sequence number is thus a free running counter represented modulo 256. There is a distinct sequence counter for each distinct MID value.
mid 0	MID, which identifies a particular connection within the tunnel. Each new connection is assigned a MID currently unused within the tunnel.
cid 4	Client ID used to assist endpoints in demultiplexing tunnels.
length 17	Size in octets of the entire packet, including header, all fields pre-sent, and payload. Length does not reflect the addition of the checksum, if present.
offset 0	Number of bytes past the L2F header at which the payload data is expected to start. If it is 0, the first byte following the last byte of the L2F header is the first byte of payload data.
key 1701976070	Value based on the authentication response given to the peer during tunnel creation. During the life of a session, the key value serves to resist attacks based on spoofing. If a packet is received in which the key does not match the expected value, the packet must be silently discarded.
L2F RECEIVED (17)	Number of bytes received.
L2F-IN Output to Async1 (16)	Payload datagram. The data came in to the VPDN code.

Field	Description
L2F-OUT (16):	Payload datagram sent out from the VPDN code to the tunnel.
L2F-OUT (101)	Ping payload datagram. The value 62 in this line is the ping packet size in hexadecimal (98 in decimal). The three lines that follow this line show ping packet data.

Examples

The following example shows output from the **debug vpdn l2x-events** command for an L2TP version 3 (L2TPv3) xconnect session on an Ethernet interface:

```
Device# debug vpdn l2x-events

23:31:18: L2X: l2tun session [1669204400], event [client request], old state [open], new
state [open]
 23:31:18: L2X: L2TP: Received L2TUN message <Connect>
 23:31:18: Tnl/Sn58458/28568 L2TP: Session state change from idle to wait-for-tunnel
 23:31:18: Tnl/Sn58458/28568 L2TP: Create session
 23:31:18: Tnl58458 L2TP: SM State idle
 23:31:18: Tnl58458 L2TP: O SCCRQ
 23:31:18: Tnl58458 L2TP: Control channel retransmit delay set to 1 seconds
 23:31:18: Tnl58458 L2TP: Tunnel state change from idle to wait-ctl-reply
 23:31:18: Tnl58458 L2TP: SM State wait-ctl-reply
 23:31:18: Tnl58458 L2TP: I SCCRP from router
 23:31:18: Tnl58458 L2TP: Tunnel state change from wait-ctl-reply to established
 23:31:18: Tnl58458 L2TP: O SCCCN to router tnlid 8012
 23:31:18: Tnl58458 L2TP: Control channel retransmit delay set to 1 seconds
 23:31:18: Tnl58458 L2TP: SM State established
 23:31:18: Tnl/Sn58458/28568 L2TP: O ICRQ to router 8012/0
 23:31:18: Tnl/Sn58458/28568 L2TP: Session state change from wait-for-tunnel to wait-reply

 23:31:19: Tnl58458 L2TP: Control channel retransmit delay set to 1 seconds
 23:31:20: %LINK-3-UPDOWN: Interface Ethernet2/1, changed state to up
 23:31:21: %LINEPROTO-5-UPDOWN: Line protocol on Interface Ethernet2/1, changed state to
up
 23:31:25: L2X: Sending L2TUN message <Connect OK>
 23:31:25: Tnl/Sn58458/28568 L2TP: O ICCN to router 8012/35149
 23:31:25: Tnl58458 L2TP: Control channel retransmit delay set to 1 seconds
 23:31:25: Tnl/Sn58458/28568 L2TP: Session state change from wait-reply to established
 23:31:25: L2X: l2tun session [1669204400], event [server response], old state [open], new
 state [open]
 23:31:26: Tnl58458 L2TP: Control channel retransmit delay set to 1 seconds
```

Examples

The following example shows debug messages for control channel authentication failure events in Cisco IOS Release 12.0(31)S:

```
Device# debug vpdn l2x-events

Tnl41855 L2TP: Per-Tunnel auth counter, Overall Failed, now 1
Tnl41855 L2TP: Tunnel auth counter, Overall Failed, now 219
```

Related Commands

Command	Description
debug aaa authentication	Displays information on AAA/TACACS+ authentication.
debug acircuit	Displays events and failures related to attachment circuits.
debug pppoe	Displays debugging information for PPPoE sessions.
debug vpdn pppoe-data	Displays data packets of PPPoE sessions.
debug vpdn pppoe-error	Displays PPPoE protocol errors that prevent a session from being established or errors that cause an established sessions to be closed.
debug vpdn pppoe-events	Displays PPPoE protocol messages about events that are part of normal session establishment or shutdown.
debug vpdn pppoe-packet	Displays each PPPoE protocol packet exchanged.
debug xconnect	Displays errors and events related to an xconnect configuration.

debug vpdn pppoe-data

Note Effective with Cisco IOS Release 12.2(13)T, the **debug vpdn pppoe-data** command is replaced by the **debug pppoe** command. See the **debug pppoe** command for more information.

To display data packets of PPP over Ethernet (PPPoE) sessions, use the **debug vpdn pppoe-data**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug vpdn pppoe-data

no debug vpdn pppoe-data

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.1(1)T	This command was introduced.
12.2(13)T	This command was replaced by the **debug pppoe** command.

Usage Guidelines

The **debug vpdn pppoe-data**command displays a large number of debug messages and should generally be used only on a debug chassis with a single active session.

Examples

The following is sample output from the **debug vpdn pppoe-data**command:

```
Router# debug vpdn pppoe-data
6d20h:%LINK-3-UPDOWN:Interface Virtual-Access1, changed state to up
6d20h:PPPoE:OUT
 contiguous pak, size 19
     FF 03 C0 21 01 01 00 0F 03 05 C2 23 05 05 06 D3
     FF 2B DA
6d20h:PPPoE:IN
 particle pak, size 1240
     C0 21 01 01 00 0A 05 06 39 53 A5 17 00 00 00 00
     00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
     00 00 00 00 00 00 00 00
6d20h:PPPoE:OUT
 contiguous pak, size 14
     FF 03 C0 21 02 01 00 0A 05 06 39 53 A5 17
6d20h:PPPoE:OUT
 contiguous pak, size 19
     FF 03 C0 21 01 02 00 0F 03 05 C2 23 05 05 06 D3
     FF 2B DA
6d20h:PPPoE:IN
 particle pak, size 1740
     C0 21 02 02 00 0F 03 05 C2 23 05 05 06 D3 FF 2B
     DA 00 80 C2 00 07 00 00 00 10 7B 01 2C D9 00 B0
     C2 EB 10 38 88 64 11 00
6d20h:PPPoE:OUT
 contiguous pak, size 30
     FF 03 C2 23 01 06 00 1A 10 99 1E 6E 8F 8C F2 C6
     EE 91 0A B0 01 CB 89 68 13 47 61 6E 67 61
6d20h:PPPoE:IN
 particle pak, size 3840
     C2 23 02 06 00 24 10 E6 84 FF 3A A4 49 19 CE D7
     AC D7 D5 96 CC 23 B3 41 6B 61 73 68 40 63 69 73
     63 6F 2E 63 6F 6D 00 00
6d20h:PPPoE:OUT
 contiguous pak, size 8
     FF 03 C2 23 03 06 00 04
6d20h:PPPoE:OUT
 contiguous pak, size 14
     FF 03 80 21 01 01 00 0A 03 06 65 65 00 66
6d20h:PPPoE:IN
 particle pak, size 1240
     80 21 01 01 00 0A 03 06 00 00 00 00 49 19 CE D7
     AC D7 D5 96 CC 23 B3 41 6B 61 73 68 40 63 69 73
     63 6F 2E 63 6F 6D 00 00
6d20h:PPPoE:OUT
 contiguous pak, size 14
     FF 03 80 21 03 01 00 0A 03 06 65 65 00 67
6d20h:PPPoE:IN
 particle pak, size 1240
     80 21 02 01 00 0A 03 06 65 65 00 66 00 04 AA AA
     03 00 80 C2 00 07 00 00 00 10 7B 01 2C D9 00 B0
     C2 EB 10 38 88 64 11 00
6d20h:PPPoE:IN
 particle pak, size 1240
     80 21 01 02 00 0A 03 06 65 65 00 67 49 19 CE D7
     AC D7 D5 96 CC 23 B3 41 6B 61 73 68 40 63 69 73
     63 6F 2E 63 6F 6D 00 00
6d20h:PPPoE:OUT
 contiguous pak, size 14
     FF 03 80 21 02 02 00 0A 03 06 65 65 00 67
6d20h:%LINEPROTO-5-UPDOWN:Line protocol on Interface Virtual-Access1,
changed state to up
6d20h:PPPoE:OUT
 contiguous pak, size 16
     FF 03 C0 21 09 01 00 0C D3 FF 2B DA 4C 4D 49 A4
6d20h:PPPoE:IN
 particle pak, size 1440
     C0 21 0A 01 00 0C 39 53 A5 17 4C 4D 49 A4 AA AA
     03 00 80 C2 00 07 00 00 00 10 7B 01 2C D9 00 B0
     C2 EB 10 38 88 64 11 00
6d20h:PPPoE:IN
 particle pak, size 1440
     C0 21 09 01 00 0C 39 53 A5 17 00 00 00 00 00 00
     00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
     00 00 00 00 00 00 00 00
```

The table below describes the significant fields shown in the display.

Table 75: debug vpdn pppoe-data Field Descriptions

Field	Descriptions
6d20h:%LINK-3-UPDOWN:Interface Virtual-Access1, changed state to up	Virtual access interface 1 came up.
6d20h:PPPoE:OUT	The host delivered a PPPoE session packet to the access concentrator.
6d20h:PPPoE:IN	The access concentrator received a PPPoE session packet.

Field	Descriptions
6d20h:%LINEPROTO-5-UPDOWN:Line protocol on Interface Virtual-Access1, changed state to up	Line protocol is up; the line can be used.
contiguous pak, size 19	Size 19 contiguous packet.
particle pak, size 1240	Size 1240 particle packet.

Related Commands

Command	Description
debug pppoe	Displays debugging information for PPPoE sessions.
debug vpdn pppoe-error	Displays PPPoE protocol errors that prevent a session from being established or errors that cause an established session to be closed.
debug vpdn pppoe-events	Displays PPPoE protocol messages about events that are part of normal session establishment or shutdown.
debug vpdn pppoe-packet	Displays each PPPoE protocol packet exchanged.
protocol (VPDN)	Specifies the L2TP that the VPDN subgroup will use.
show vpdn	Displays information about active L2F protocol tunnel and message identifiers in a VPDN.
vpdn enable	Enables virtual private dialup networking on the router and informs the router to look for tunnel definitions in a local database and on a remote authorization server (home gateway), if one is present.

debug vpdn pppoe-error

Note Effective with Cisco IOS Release 12.2(13)T, the **debug vpdn pppoe-error** command is replaced by the **debug pppoe** command. See the **debug pppoe** command for more information.

To display PPP over Ethernet (PPPoE) protocol errors that prevent a session from being established or errors that cause an established sessions to be closed, use the **debug vpdn pppoe-error**command in privileged EX**EC**mode. To disable debugging output, use the **no** form of this command.

debug vpdn pppoe-error

no debug vpdn pppoe-error

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.1(1)T	This command was introduced.
12.2(13)T	This command was replaced by the **debug pppoe** command.

Examples

The following is a full list of error messages displayed by the **debug vpdn pppoe-error**command:

```
PPPOE:pppoe_acsys_err cannot grow packet
PPPoE:Cannot find PPPoE info
PPPoE:Bad MAC address:00b0c2eb1038
PPPOE:PADI has no service name tag
PPPoE:pppoe_handle_padi cannot add AC name/Cookie.
PPPoE:pppoe_handle_padi cannot grow packet
PPPoE:pppoe_handle_padi encap failed
PPPoE cannot create virtual access.
PPPoE cannot allocate session structure.
PPPoE cannot store session element in tunnel.
PPPoE cannot allocate tunnel structure.
PPPoE cannot store tunnel
PPPoE:VA221:No Session, Packet Discarded
PPPOE:Tried to shutdown a null session
PPPoE:Session already open, closing
PPPoE:Bad cookie:src_addr=00b0c2eb1038
PPPoE:Max session count on mac elem exceeded:mac=00b0c2eb1038
PPPoE:Max session count on vc exceeded:vc=3/77
PPPoE:Bad MAC address - dropping packet
PPPoE:Bad version or type - dropping packet
```

The table below describes the significant fields shown in the display.

Table 76: debug vpdn pppoe-error Field Descriptions

Field	Descriptions
PPPOE:pppoe_acsys_err cannot grow packet	Asynchronous PPPoE packet initialization error.
PPPoE:Cannot find PPPoE info	The access concentrator sends a PADO to the host.
PPPoE:Bad MAC address:00b0c2eb1038	The host was unable to identify the Ethernet MAC address.
PPPOE:PADI has no service name tag	PADI requires a service name tag.
PPPoE:pppoe_handle_padi cannot add AC name/Cookie.	pppoe_handle_padi could not append AC name.
PPPoE:pppoe_handle_padi cannot grow packet	pppoe_handle_padi could not append packet.
PPPoE:pppoe_handle_padi encap failed	pppoe_handle_padi could not specify PPPoE on ATM encapsulation.
PPPoE cannot create virtual access.	PPPoE session unable to verify virtual access interface.
PPPoE cannot allocate session structure.	PPPoE session unable to allocate Stage Protocol.
PPPoE cannot store session element in tunnel.	PPPoE tunnel cannot allocate session element.
PPPoE cannot allocate tunnel structure.	PPPoE tunnel unable to allocate Stage Protocol.
PPPoE cannot store tunnel	PPPoE configuration settings unable to initialize a tunnel.
PPPoE:VA221:No Session, Packet Discarded	No sessions created. All packets dropped.
PPPOE:Tried to shutdown a null session	Null session shutdown.
PPPoE:Session already open, closing	PPPoE session already open.
PPPoE:Bad cookie:src_addr=00b0c2eb1038	PPPoE session unable to append new cookie.
PPPoE:Max session count on mac elem exceeded:mac=00b0c2eb1038	The maximum number of sessions exceeded the Ethernet MAC address.
PPPoE:Max session count on vc exceeded:vc=3/77	The maximum number of sessions exceeded the PVC connection.
PPPoE:Bad MAC address - dropping packet	The host was unable to identify the MAC address. Packet dropped.

Field	Descriptions
PPPoE:Bad version or type - dropping packet	The host was unable to identify the encapsulation type.

Related Commands

Command	Description
debug pppoe	Displays debugging information for PPPoE sessions.
debug vpdn pppoe-data	Displays data packets of PPPoE sessions.
debug vpdn pppoe-events	Displays PPPoE protocol messages about events that are part of normal session establishment or shutdown.
debug vpdn pppoe-packet	Displays each PPPoE protocol packet exchanged.
protocol (VPDN)	Specifies the L2TP that the VPDN subgroup will use.
show vpdn	Displays information about active L2F protocol tunnel and message identifiers in a VPDN.
vpdn enable	Enables virtual private dialup networking on the router and informs the router to look for tunnel definitions in a local database and on a remote authorization server (home gateway), if one is pre-sent.

debug vpdn pppoe-events

Note Effective with Cisco IOS Release 12.2(13)T, the **debug vpdn pppoe-events**command is replaced by the **debug pppoe** command. See the **debug pppoe** command for more information.

To display PPP over Ethernet (PPPoE) protocol messages about events that are part of normal session establishment or shutdown, use the **debug vpdn pppoe-events**command in privileged E**XEC**mode. To disable debugging output, use the **no** form of this command.

debug vpdn pppoe-events

no debug vpdn pppoe-events

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.1(1)T	This command was introduced.
12.2(13)T	This command was replaced by the **debug pppoe** command.

Examples

The following is sample output from the **debug vpdn pppoe-events**command:

```
1w5d:IN PADI from PPPoE tunnel
1w5d:OUT PADO from PPPoE tunnel
1w5d:IN PADR from PPPoE tunnel
1w5d:PPPoE:VPN session created.
1w5d:%LINK-3-UPDOWN:Interface Virtual-Access2, changed state to up
1w5d:%LINEPROTO-5-UPDOWN:Line protocol on Interface Virtual-Access2, changed state to up
```

The table below describes the significant fields shown in the display.

Table 77: debug vpdn pppoe-events Field Descriptions

Field	Descriptions
1w5d:IN PADI from PPPoE tunnel	The access concentrator receives an Active Discovery Initiation (PADI) packet from the PPPoE tunnel.
1w5d:OUT PADO from PPPoE tunnel	The access concentrator sends an Active Discovery Offer (PADO) to the host.

Field	Descriptions
1w5d:IN PADR from PPPoE tunnel	The host sends a single Active Discovery Request (PADR) to the access concentrator that it has chosen.
1w5d:PPPoE:VPN session created.	The access concentrator receives the PADR packet and creates a virtual private network (VPN) session.
1w5d:%LINK-3-UPDOWN:Interface Virtual-Access2, changed state to up	Virtual access interface 2 came up.
1w5d:%LINEPROTO-5-UPDOWN:Line protocol on Interface Virtual-Access2, changed state to up	Line protocol is up. The line can be used.

Related Commands

Command	Description
debug pppoe	Displays debugging information for PPPoE sessions.
debug vpdn pppoe-data	Displays data packets of PPPoE sessions.
debug vpdn pppoe-error	Displays PPPoE protocol errors that prevent a session from being established or errors that cause an established session to be closed.
debug vpdn pppoe-packet	Displays each PPPoE protocol packet exchanged.
protocol (VPDN)	Specifies the L2TP that the VPDN subgroup will use.
show vpdn	Displays information about active L2F protocol tunnel and message identifiers in a VPDN.
vpdn enable	Enables virtual private dialup networking on the router and informs the router to look for tunnel definitions in a local database and on a remote authorization server (home gateway), if one is pre-sent.

debug vpdn pppoe-packet

Note Effective with Cisco IOS Release 12.2(13)T, the **debug vpdn pppoe-packet**command is replaced by the **debug pppoe** command. See the **debug pppoe** command for more information.

To display each PPP over Ethernet (PPPoE) protocol packet exchanged, use the **debug vpdn pppoe-packet**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug vpdn pppoe-packet

no debug vpdn pppoe-packet

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.1(1)T	This command was introduced.
12.2(13)T	This command was replaced by the **debug pppoe** command.

Usage Guidelines

The **debug vpdn pppoe-packet**command displays a large number of debug messages and should generally only be used on a debug chassis with a single active session.

Examples

The following is sample output from the **debug vpdn pppoe-packet**command:

```
PPPoE control packets debugging is on
1w5d:PPPoE:discovery packet
 contiguous pak, size 74
     FF FF FF FF FF FF 00 10 7B 01 2C D9 88 63 11 09
     00 00 00 04 01 01 00 00 00 00 00 00 00 00 00 00
     00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ...
1w5d:OUT PADO from PPPoE tunnel
 contiguous pak, size 74
     00 01 09 00 AA AA 03 00 80 C2 00 07 00 00 00 10
     7B 01 2C D9 00 90 AB 13 BC A8 88 63 11 07 00 00
     00 20 01 01 00 00 01 02 00 04 41 67 6E 69 01 ...
1w5d:PPPoE:discovery packet
 contiguous pak, size 74
     00 90 AB 13 BC A8 00 10 7B 01 2C D9 88 63 11 19
     00 00 00 20 01 01 00 00 01 02 00 04 41 67 6E 69
     01 04 00 10 B7 4B 86 5B 90 A5 EF 11 64 A9 BA ...
```

The table below describes the significant fields shown in the display.

Table 78: debug vpdn pppoe-packet Field Descriptions

Field	Descriptions
PPPoE control packets debugging is on	PPPoE debugging of packets is enabled.
1w5d:PPPoE:discovery packet	The host performs a discovery to initiate a PPPoE session.
1w5d:OUT PADO from PPPoE tunnel	The access concentrator sends a PADO to the host.
1w5d:PPPoE:discovery packet	The host performs a discovery to initiate a PPPoE session.
contiguous pak, size 74	Size 74 contiguous packet.

Related Commands

Command	Description
debug pppoe	Displays debugging information for PPPoE sessions.
debug vpdn pppoe-data	Displays data packets of PPPoE sessions.
debug vpdn pppoe-error	Displays PPPoE protocol errors that prevent a session from being established or errors that cause an established session to be closed.
debug vpdn pppoe-events	Displays PPPoE protocol messages about events that are part of normal session establishment or shutdown.
protocol (VPDN)	Specifies the L2TP that the VPDN subgroup will use.
show vpdn	Displays information about active L2F protocol tunnel and message identifiers in a VPDN.
vpdn enable	Enables virtual private dialup networking on the router and informs the router to look for tunnel definitions in a local database and on a remote authorization server (home gateway), if one is pre-sent.

debug vpdn redundancy

To debug virtual private dial-up network (VPDN) sessions that contain redundancy status, use the **debug vpdn redundancy**command in user or privileged EXEC mode. To disable this debugging, use the **no** form of this command.

debug vpdn redundancy {**cf**| **detail**| **error**| **event**| **fsm**| **resync**| **rf**}

no debug vpdn redundancy

Syntax Description

cf	Displays VPDN redundancy-facility (cf) events.
detail	Displays VPDN redundancy details.
error	Displays VPDN redundancy errors.
event	Displays VPDN redundancy events.
fsm	Displays VPDN redundancy forwarding-service manager (fsm) events.
resync	Displays VPDN redundancy resynchronizations.
rf	Displays VPDN redundancy-facility (rf) events.

Command Modes

User EXEC (>) Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS XE Release 2.2.	This command was introduced in Cisco IOS XE Release 2.2.

Usage Guidelines

Use the **debug vpdn redundancy** command in privileged EXEC mode to display a list of VPDN sessions that have redundancy events and errors.

Use the **show vpdn redundancy**command in privileged EXEC mode to display information on the state of the VPDN session redundancy data.

Examples

The following example shows how to display a debug of redundancy events during the setup and termination of an tunnel for an LNS active Route Processor (RP):

```
LNS1> debug
 enable
LNS1# debug
```

```
 vpdn
 redundancy
 cf
L2TP redundancy cf debugging is on
LNS1# debug
 vpdn
 redundancy
 detail
L2TP redundancy details debugging is on
LNS1# debug
 vpdn
 redundancy
 error
L2TP redundancy errors debugging is on
LNS1# debug
 vpdn
 redundancy
 event
L2TP redundancy events debugging is on
LNS1# debug
 vpdn
 redundancy
 fsm
L2TP redundancy fsm debugging is on
LNS1# debug
 vpdn
 redundancy
 resync
L2TP redundancy resync debugging is on
LNS1# debug
 vpdn
 redundancy
 rf
L2TP redundancy rf debugging is on
LNS1#
*Aug 26 18:00:00.467: %SYS-5-CONFIG_I: Configured from console by console
LNS1#
*Aug 26 18:00:45.631: L2TP tnl   01000:________: CCM initialized CCM session
*Aug 26 18:00:45.631: : L2TP HA:CC playback chkpt skipped, CC not doing HA
*Aug 26 18:00:45.711: : L2TP HA FSM:Receive proto FSM event 19
*Aug 26 18:00:45.711: : L2TP HA FSM:Receive RxSCCRQ
*Aug 26 18:00:45.711: : L2TP HA:lcm_cc alloc: l2tp_cc 070B45B8, lcm_cc 02FE55E8
*Aug 26 18:00:45.711: : L2TP HA FSM:FSM-CC ev Rx-SCCRQ
*Aug 26 18:00:45.711: : L2TP HA FSM:FSM-CC    Idle->Wt-ChkptSidRmt
*Aug 26 18:00:45.711: : L2TP HA FSM:FSM-CC do Block-Tx-AckSCCRQ
*Aug 26 18:00:45.711: : L2TP HA FSM:Checkpoint Two Cc IDs
*Aug 26 18:00:45.711: L2TP HA CF: Chkpt send: s/c id 0/52631, BothCcId, seq 0, ns/nr 0/0,
rid 51583, len 52; flush = 1, ctr 1
*Aug 26 18:00:45.711: 01000:0000CD97: L2TP HA:Enqueue peer Ns 0 to ns_q, seq 1 (q sz 0)
*Aug 26 18:00:45.711: L2TP tnl   01000:0000CD97: Encoding SCCRQ-IN CHKPT
*Aug 26 18:00:45.711: L2TP tnl   01000:0000CD97: Tx CHKPT
*Aug 26 18:00:45.739: L2TP tnl   01000:0000CD97: Encoding SCCRP-OUT CHKPT
*Aug 26 18:00:45.739: L2TP tnl   01000:0000CD97: Tx CHKPT
*Aug 26 18:00:45.739: : L2TP HA:Adjust local window size to 10
*Aug 26 18:00:45.739: 01000:0000CD97: L2TP HA FSM:Receive proto TxCM event SCCRP
*Aug 26 18:00:45.739: : L2TP HA FSM:Receive TxSCCRP
LNS1#
*Aug 26 18:00:45.739: : L2TP HA FSM:FSM-CC ev Tx-SCCRP
*Aug 26 18:00:45.739: : L2TP HA FSM:FSM-CC    Wt-ChkptSidRmt->WtCcIdRmt2
*Aug 26 18:00:45.739: : L2TP HA FSM:FSM-CC do Block-Tx-SCCRP
*Aug 26 18:00:45.739: 01000:0000CD97: L2TP HA FSM:Found blocked RxSCCRQ, seq_num 1
*Aug 26 18:00:45.739: 01000:0000CD97: L2TP HA FSM:Queued SCCRP to CC hold_q
*Aug 26 18:00:46.863: : L2TP HA FSM:CHKPT status callback: status 0, len 56
*Aug 26 18:00:46.863: : L2TP HA FSM:Context s/c id 0/52631, BothCcId, seq 1, ns/nr 0/0, rid
 51583, len 52
*Aug 26 18:00:46.863: L2TP HA CF: Rcvd status s/c id 0/52631, BothCcId, seq 1, ns/nr 0/0,
rid 51583, len 52
*Aug 26 18:00:46.863: L2TP HA CF: Rcvd status 0: len 56
*Aug 26 18:00:46.863: L2TP HA CF: Status content s/c id 0/52631, BothCcId, seq 1, ns/nr
0/0, rid 51583, len 52
*Aug 26 18:00:46.863: 01000:0000CD97: L2TP HA FSM:Recv chkpt ack: s/c id 0/52631, BothCcId,
 seq 1, ns/nr 0/0, rid 51583, len 52
```

```
*Aug 26 18:00:46.863: : L2TP HA FSM:Receive CC-ChkptAck
*Aug 26 18:00:46.863: : L2TP HA FSM:FSM-CC ev Rx-CkpACK-CcID-Rmt
*Aug 26 18:00:46.863: : L2TP HA FSM:FSM-CC    WtCcIdRmt2->Wt-RxSccn
*Aug 26 18:00:46.863: : L2TP HA FSM:FSM-CC do Allow-Tx-SCCRP2
*Aug 26 18:00:46.863: : L2TP HA FSM:Received Chkpt of local + remote CC ID
*Aug 26 18:00:46.863: 01000:0000CD97: L2TP HA:Try to remove from CC's ns_q: seq num 1
(current Ns 1)
*Aug 26 18:00:46.863: 01000:0000CD97: L2TP HA:Ns entry to remove: found (current Ns 1)
*Aug 26 18:00:46.863: 01000:0000CD97: L2TP HA:Advance peer Nr to 1 (ns_q sz 0)
*Aug 26 18:00:46.863: 01000:0000CD97: L2TP HA:CC send all unblocked if can
LNS1#
*Aug 26 18:00:46.863: 01000:0000CD97: L2TP HA:CC send one blocked CM (SCCRP): ns 0 (0), nr
 1
*Aug 26 18:00:46.863: L2TP HA CF: O SCCRP 51583/0 ns/nr 0/1
*Aug 26 18:00:47.867: 01000:0000CD97: L2TP HA FSM:Receive Cm-Ack
*Aug 26 18:00:47.867: 01000:0000CD97: L2TP HA FSM:Receive CC Cm-Ack
*Aug 26 18:00:47.867: : L2TP HA FSM:FSM-CC ev Rx-CmACK
*Aug 26 18:00:47.867: : L2TP HA FSM:FSM-CC    in Wt-RxSccn
*Aug 26 18:00:47.867: : L2TP HA FSM:FSM-CC do Ignore
*Aug 26 18:00:47.867: 01000:0000CD97: L2TP HA FSM:Ignore event
*Aug 26 18:00:47.867: 01000:0000CD97: L2TP HA FSM:Check for Ns/Nr update 1, peer 1
*Aug 26 18:00:47.867: 01000:0000CD97: L2TP HA FSM:Receive peer Ns/Nr update (1,0/1,1, int
1, rx 1, 1) (ns_q sz 0)
*Aug 26 18:00:47.867: 01000:0000CD97: L2TP HA FSM:Peer Ns 1 (1), Nr 1 (ns_q sz 0)
*Aug 26 18:00:48.087: 01000:0000CD97: L2TP HA FSM:Check for Ns/Nr update 1, peer 1
*Aug 26 18:00:48.087: 01000:0000CD97: L2TP HA FSM:Receive peer Ns/Nr update (1,0/1,1, int
1, rx 1, 1) (ns_q sz 0)
*Aug 26 18:00:48.087: 01000:0000CD97: L2TP HA FSM:Peer Ns 1 (2), Nr 1 (ns_q sz 0)
*Aug 26 18:00:48.087: : L2TP HA FSM:Receive proto FSM event 21
*Aug 26 18:00:48.087: 01000:0000CD97: L2TP HA FSM:Receive RxSCCCN
*Aug 26 18:00:48.087: : L2TP HA FSM:FSM-CC ev Rx-SCCCN
*Aug 26 18:00:48.087: : L2TP HA FSM:FSM-CC    Wt-RxSccn->WtCcsUp
*Aug 26 18:00:48.087: : L2TP HA FSM:FSM-CC do Allow-Tx-AckSCCCN
*Aug 26 18:00:48.087: 01000:0000CD97: L2TP HA FSM:Allow TxSCCCN-ACK
*Aug 26 18:00:48.087: 01000:0000CD97: L2TP HA FSM:Receive CcUp
*Aug 26 18:00:48.087: : L2TP HA FSM:FSM-CC ev Proto CcUp
*Aug 26 18:00:48.087: : L2TP HA FSM:FSM-CC    WtCcsUp->Wt-CkptCcUp
*Aug 26 18:00:48.087: : L2TP HA FSM:FSM-CC do Chkpt-CcUp2
*Aug 26 18:00:48.087: : L2TP HA FSM:Checkpoint CcUp
*Aug 26 18:00:48.087: L2TP HA CF: Chkpt send: s/c id 0/52631, CcUp, seq 0, ns/nr 1/1, rid
0, len 52; flush = 1, ctr 2
*Aug 26 18:00:48.091: L2TP tnl   01000:0000CD97: CCM added sync data
*Aug 26 18:00:48.095: 01000:0000CD97: L2TP HA FSM:Check for Ns/Nr update 2, peer 1
*Aug 26 18:00:48.095: 01000:0000CD97: L2TP HA FSM:Receive peer Ns/Nr update (2,1/1,1, int
2, rx 1, 2) (ns_q sz 0)
*Aug 26 18:00:48.095: 01000:0000CD97: L2TP HA FSM:Peer Ns 2 (3), Nr 1 (ns_q sz 0)
*Aug 26 18:00:48.095: L2TP _____:01000:000036F8: Encoding ICRQ-IN CHKPT
*Aug 26 18:00:48.095: L2TP _____:01000:000036F8: Tx CHKPT
*Aug 26 18:00:48.095: : L2TP HA FSM:Receive proto FSM event 3
*Aug 26 18:00:48.095: : L2TP HA FSM:Receive RxICRQ
*Aug 26 18:00:48.095: _____:01000:000036F8: L2TP HA FSM:  Using ICRQ FSM
*Aug 26 18:00:48.095: _____:01000:000036F8: L2TP HA FSM:FSM-Sn ev created
*Aug 26 18:00:48.095: _____:01000:000036F8: L2TP HA FSM:FSM-Sn    Init->Idle
*Aug 26 18:00:48.095: _____:01000:000036F8: L2TP HA FSM:FSM-Sn do none
*Aug 26 18:00:48.095: _____:01000:000036F8: L2TP HA FSM:FSM-Sn ev Rx-xCRQ
*Aug 26 18:00:48.095: _____:01000:000036F8: L2TP HA FSM:FSM-Sn    Idle->Wt-ChkptSidRmt
*Aug 26 18:00:48.095: _____:01000:000036F8: L2TP HA FSM:FSM-Sn do Block-Tx-AckXCRQ
*Aug 26 18:00:48.095: _____:01000:000036F8: L2TP HA FSM:Checkpoint TwoSessionIDs
*Aug 26 18:00:48.095: L2TP HA CF: Chkpt send: s/c id 14072/52631, BothSesId, seq 0, ns/nr
1/2, rid 40276, len 52; flush = 1, ctr 3
*Aug 26 18:00:48.095: _____:01000:000036F8: L2TP HA:Enqueue peer Ns 2 to ns_q, seq 3 (q sz
 0)
*Aug 26 18:00:48.131: : L2TP HA:Try to buffer sock msg type 19
*Aug 26 18:00:48.131: : L2TP HA:Buffering skipped
*Aug 26 18:00:48.131: L2TP _____:01000:000036F8: Encoding ICRP-OUT CHKPT
*Aug 26 18:00:48.131: L2TP _____:01000:000036F8: Tx CHKPT
*Aug 26 18:00:48.131: 01000:0000CD97: L2TP HA FSM:Receive proto TxCM event ICRP
*Aug 26 18:00:48.131: _____:_____:000036F8: L2TP HA FSM:Receive TxICRP
*Aug 26 18:00:48.131: _____:01000:000036F8: L2TP HA FSM:FSM-Sn ev Tx-xCRP
*Aug 26 18:00:48.131: _____:01000:000036F8: L2TP HA FSM:FSM-Sn    Wt-ChkptSidRmt->Wt-SesIdRmt2
*Aug 26 18:00:48.131: _____:01000:000036F8: L2TP HA FSM:FSM-Sn do Block-Tx-xCRP
*Aug 26 18:00:48.131: _____:01000:000036F8: L2TP HA FSM:Found blocked RxICRQ, seq_num 3
LNS1#
```

```
*Aug 26 18:00:48.131: _____:01000:000036F8: L2TP HA FSM:Queued xCRP to session hold_q
*Aug 26 18:00:48.131: : L2TP HA:Try to buffer sock msg type 23
*Aug 26 18:00:48.131: : L2TP HA:CC not in resync state, buffering skipped
*Aug 26 18:00:49.115: 01000:0000CD97: L2TP HA FSM:Check for Ns/Nr update 2, peer 1
*Aug 26 18:00:49.115: 01000:0000CD97: L2TP HA FSM:Receive peer Ns/Nr update (2,2/1,1, int
3, rx 1, 3) (ns_q sz 1)
*Aug 26 18:00:49.211: : L2TP HA FSM:CHKPT status callback: status 0, len 56
*Aug 26 18:00:49.211: : L2TP HA FSM:Context s/c id 0/52631, CcUp, seq 2, ns/nr 1/1, rid 0,
 len 52
*Aug 26 18:00:49.211: : L2TP HA FSM:CHKPT status callback: status 0, len 56
*Aug 26 18:00:49.211: : L2TP HA FSM:Context s/c id 14072/52631, BothSesId, seq 3, ns/nr
1/2, rid 40276, len 52
*Aug 26 18:00:49.211: L2TP HA CF: Rcvd status s/c id 0/52631, CcUp, seq 2, ns/nr 1/1, rid
0, len 52
*Aug 26 18:00:49.211: L2TP HA CF: Rcvd status 0: len 56
*Aug 26 18:00:49.211: L2TP HA CF: Status content s/c id 0/52631, CcUp, seq 2, ns/nr 1/1,
rid 0, len 52
*Aug 26 18:00:49.211: 01000:0000CD97: L2TP HA FSM:Recv chkpt ack: s/c id 0/52631, CcUp, seq
 2, ns/nr 1/1, rid 0, len 52
*Aug 26 18:00:49.211: : L2TP HA FSM:Receive CC-ChkptAck
*Aug 26 18:00:49.211: : L2TP HA FSM:FSM-CC ev Rx-CkpACK-CcUp
*Aug 26 18:00:49.211: : L2TP HA FSM:FSM-CC    Wt-CkptCcUp->ProcCcsUp
*Aug 26 18:00:49.211: : L2TP HA FSM:FSM-CC do Proc-ChpACK-CcUp2
*Aug 26 18:00:49.211: : L2TP HA FSM:Received chkpt ACK of CcUp
*Aug 26 18:00:49.211: L2TP HA CF: Rcvd status s/c id 14072/52631, BothSesId, seq 3, ns/nr
1/2, rid 40276, len 52
*Aug 26 18:00:49.211: L2TP HA CF: Rcvd status 0: len 56
*Aug 26 18:00:49.211: L2TP HA CF: Status content s/c id 14072/52631, BothSesId, seq 3, ns/nr
 1/2, rid 40276, len 52
*Aug 26 18:00:49.211: 01000:0000CD97: L2TP HA FSM:Recv chkpt ack: s/c id 14072/52631,
BothSesId, seq 3, ns/nr 1/2, rid 40276, len 52
*Aug 26 18:00:49.211: _____:_____:000036F8: L2TP HA FSM:Receive Session-ChkptAck
*Aug 26 18:00:49.211: _____:01000:000036F8: L2TP HA FSM:FSM-Sn ev Rx-CktACK-SesID-Rmt
*Aug 26 18:00:49.211: _____:01000:000036F8: L2TP HA FSM:FSM-Sn    Wt-SesIdRmt2->Wt-RxXccn
*Aug 26 18:00:49.211: _____:01000:000036F8: L2TP HA FSM:FSM-Sn do Allow-Tx-xCRP
*Aug 26 18:00:49.211: 01000:0000CD97: L2TP HA:Try to remove from CC's ns_q: seq num 3
(current Ns 3)
*Aug 26 18:00:49.211: _____:01000:000036F8: L2TP HA:Ns entry to remove: found (current Ns
3)
*Aug 26 18:00:49.211: _____:01000:000036F8: L2TP HA:Advance peer Nr to 3 (ns_q sz 0)
*Aug 26 18:00:49.211: _____:01000:000036F8: L2TP HA:Session send all unblocked
*Aug 26 18:00:49.211: 01000:0000CD97: L2TP HA:CC send if can (ICRP): ns 1 (1, 1), nr 3 (3)
*Aug 26 18:00:49.211: L2TP HA CF: O ICRP 51583/40276 ns/nr 1/3
*Aug 26 18:00:49.231: 01000:0000CD97: L2TP HA FSM:Receive Cm-Ack
*Aug 26 18:00:49.231: _____:_____:000036F8: L2TP HA FSM:Receive session Cm-Ack
LNS1#
*Aug 26 18:00:49.231: _____:01000:000036F8: L2TP HA FSM:FSM-Sn ev Rx-CmACK
*Aug 26 18:00:49.231: _____:01000:000036F8: L2TP HA FSM:FSM-Sn    in Wt-RxXccn
*Aug 26 18:00:49.231: _____:01000:000036F8: L2TP HA FSM:FSM-Sn do Ignore
*Aug 26 18:00:49.231: _____:01000:000036F8: L2TP HA FSM:Ignore event
*Aug 26 18:00:49.231: 01000:0000CD97: L2TP HA FSM:Check for Ns/Nr update 3, peer 2
*Aug 26 18:00:49.231: 01000:0000CD97: L2TP HA FSM:Receive peer Ns/Nr update (3,2/2,2, int
3, rx 2, 3) (ns_q sz 0)
*Aug 26 18:00:49.231: 01000:0000CD97: L2TP HA FSM:Peer Ns 3 (3), Nr 2 (ns_q sz 0)
LNS1#
*Aug 26 18:00:50.407: 01000:0000CD97: L2TP HA FSM:Check for Ns/Nr update 3, peer 2
*Aug 26 18:00:50.407: 01000:0000CD97: L2TP HA FSM:Receive peer Ns/Nr update (3,2/2,2, int
3, rx 2, 3) (ns_q sz 0)
*Aug 26 18:00:50.407: 01000:0000CD97: L2TP HA FSM:Peer Ns 3 (4), Nr 2 (ns_q sz 0)
*Aug 26 18:00:50.407: : L2TP HA FSM:Receive proto FSM event 5
*Aug 26 18:00:50.407: _____:_____:000036F8: L2TP HA FSM:Receive RxICCN
*Aug 26 18:00:50.407: _____:01000:000036F8: L2TP HA FSM:FSM-Sn ev Rx-xCCN
*Aug 26 18:00:50.407: _____:01000:000036F8: L2TP HA FSM:FSM-Sn    Wt-RxXccn->Wt-SessUp
*Aug 26 18:00:50.407: _____:01000:000036F8: L2TP HA FSM:FSM-Sn do Allow-Tx-AckXCCN
*Aug 26 18:00:50.407: _____:01000:000036F8: L2TP HA FSM:Allow TxICCN-ACK
*Aug 26 18:00:50.407: L2TP _____:01000:000036F8: Encoding ICCN-IN CHKPT
*Aug 26 18:00:50.407: L2TP _____:01000:000036F8: Tx CHKPT
*Aug 26 18:00:50.407: _____:_____:000036F8: L2TP HA FSM:Receive SessionUp
*Aug 26 18:00:50.407: _____:01000:000036F8: L2TP HA FSM:FSM-Sn ev Proto SessUp
*Aug 26 18:00:50.407: _____:01000:000036F8: L2TP HA FSM:FSM-Sn    Wt-SessUp->Wt-CkptSesUp
*Aug 26 18:00:50.407: _____:01000:000036F8: L2TP HA FSM:FSM-Sn do Chkpt-SesUp2
*Aug 26 18:00:50.407: _____:01000:000036F8: L2TP HA FSM:Checkpoint SessionUP
*Aug 26 18:00:50.407: L2TP HA CF: Chkpt send: s/c id 14072/52631, SesUp, seq 0, ns/nr 2/3,
 rid 0, len 52; flush = 1, ctr 4
*Aug 26 18:00:51.055: : L2TP HA FSM:CHKPT status callback: status 0, len 56
*Aug 26 18:00:51.055: : L2TP HA FSM:Context s/c id 14072/52631, SesUp, seq 4, ns/nr 2/3,
rid 0, len 52
*Aug 26 18:00:51.055: L2TP HA CF: Rcvd status s/c id 14072/52631, SesUp, seq 4, ns/nr 2/3,
 rid 0, len 52
*Aug 26 18:00:51.055: L2TP HA CF: Rcvd status 0: len 56
*Aug 26 18:00:51.055: L2TP HA CF: Status content s/c id 14072/52631, SesUp, seq 4, ns/nr
2/3, rid 0, len 52
*Aug 26 18:00:51.055: 01000:0000CD97: L2TP HA FSM:Recv chkpt ack: s/c id 14072/52631, SesUp,
 seq 4, ns/nr 2/3, rid 0, len 52
*Aug 26 18:00:51.055: _____:_____:000036F8: L2TP HA FSM:Receive Session-ChkptAck
*Aug 26 18:00:51.055: _____:01000:000036F8: L2TP HA FSM:FSM-Sn ev Rx-CktACK-SesUp
*Aug 26 18:00:51.055: _____:01000:000036F8: L2TP HA FSM:FSM-Sn    Wt-CkptSesUp->Proc-SessUp
*Aug 26 18:00:51.055: _____:01000:000036F8: L2TP HA FSM:FSM-Sn do Proc-ChpACK-SesUp
*Aug 26 18:00:51.055: _____:01000:000036F8: L2TP HA FSM:Received chkpt ACK of SessionUP
*Aug 26 18:00:51.347: %LINK-3-UPDOWN: Interface Virtual-Access2, changed state to up
LNS1#
*Aug 26 18:00:51.635: : L2TP HA:Try to buffer sock msg type 26
*Aug 26 18:00:51.635: : L2TP HA:CC not in resync state, buffering skipped
*Aug 26 18:00:51.659: : L2TP HA:Try to buffer sock msg type 26
*Aug 26 18:00:51.659: : L2TP HA:CC not in resync state, buffering skipped
LNS1#
*Aug 26 18:00:52.363: %LINEPROTO-5-UPDOWN: Line protocol on Interface Virtual-Access2,
changed state to up
LNS1#
LNS1# clear
 vpdn
 all
Proceed with clearing all tunnels? [confirm]
LNS1#
*Aug 26 18:01:21.271: 00001:_____:000036F8: L2TP HA FSM:Receive Session-CC-Rm
*Aug 26 18:01:21.271: 00001:_____:000036F8: L2TP HA FSM:Receive SessionRm
*Aug 26 18:01:21.271: 01000:0000CD97: L2TP HA FSM:Receive proto TxCM event StopCCN
*Aug 26 18:01:21.271: 01000:0000CD97: L2TP HA FSM:Receive TxSTOPCCN
*Aug 26 18:01:21.271: : L2TP HA FSM:FSM-CC ev Tx-STOPCCN
*Aug 26 18:01:21.271: : L2TP HA FSM:FSM-CC    ProcCcsUp->Wt-CkptCcDn
*Aug 26 18:01:21.271: : L2TP HA FSM:FSM-CC do Chkpt-CcDwn
*Aug 26 18:01:21.271: 01000:0000CD97: L2TP HA FSM:Receive TxSTOPCCN while CC up
*Aug 26 18:01:21.271: 01000:0000CD97: L2TP HA:CC ns_q cleanup: overall head Ns old/new =
4/4 (Q sz 0)
LNS1#
*Aug 26 18:01:21.271: : L2TP HA FSM:Checkpoint CCDown
*Aug 26 18:01:21.271: L2TP HA CF: Chkpt send: s/c id 0/52631, CcDwn, seq 0, ns/nr 2/3, rid
 0, len 52; flush = 1, ctr 5
*Aug 26 18:01:21.271: 01000:0000CD97: L2TP HA FSM:Queued STOPCCN to cc hold_q
*Aug 26 18:01:21.295: : L2TP HA:Try to buffer sock msg type 22
*Aug 26 18:01:21.295: : L2TP HA:Buffering skipped
*Aug 26 18:01:22.423: : L2TP HA FSM:CHKPT status callback: status 0, len 56
*Aug 26 18:01:22.423: : L2TP HA FSM:Context s/c id 0/52631, CcDwn, seq 5, ns/nr 2/3, rid
0, len 52
*Aug 26 18:01:22.423: L2TP HA CF: Rcvd status s/c id 0/52631, CcDwn, seq 5, ns/nr 2/3, rid
 0, len 52
*Aug 26 18:01:22.423: L2TP HA CF: Rcvd status 0: len 56
*Aug 26 18:01:22.423: L2TP HA CF: Status content s/c id 0/52631, CcDwn, seq 5, ns/nr 2/3,
rid 0, len 52
*Aug 26 18:01:22.423: 01000:0000CD97: L2TP HA FSM:Recv chkpt ack: s/c id 0/52631, CcDwn,
seq 5, ns/nr 2/3, rid 0, len 52
*Aug 26 18:01:22.423: : L2TP HA FSM:Receive CC-ChkptAck
*Aug 26 18:01:22.423: : L2TP HA FSM:FSM-CC ev Rx-CkpACK-CcDwn
*Aug 26 18:01:22.423: : L2TP HA FSM:FSM-CC    Wt-CkptCcDn->Wt-RxStopAck
*Aug 26 18:01:22.423: : L2TP HA FSM:FSM-CC do Allow-Tx-STOPCCN4
*Aug 26 18:01:22.423: : L2TP HA FSM:Received Chkpt of CC removal
*Aug 26 18:01:22.423: 01000:0000CD97: L2TP HA:Try to remove from CC's ns_q: seq num 5
(current Ns 4)
*Aug 26 18:01:22.423: 01000:0000CD97: L2TP HA:Ns entry to remove: not found (current Ns 4)
*Aug 26 18:01:22.423: 01000:0000CD97: L2TP HA:CC send all unblocked if can
*Aug 26 18:01:22.423: 01000:0000CD97: L2TP HA:CC send one blocked CM (SCCRP): ns 2 (2), nr
 4
*Aug 26 18:01:22.451: 01000:0000CD97: L2TP HA FSM:Receive Cm-Ack
*Aug 26 18:01:22.451: 01000:0000CD97: L2TP HA FSM:Receive CC Cm-Ack
*Aug 26 18:01:22.451: : L2TP HA FSM:FSM-CC ev Rx-CmACK
*Aug 26 18:01:22.451: : L2TP HA FSM:FSM-CC    Wt-RxStopAck->Wt-CkptCcRm
```

```
*Aug 26 18:01:22.451: : L2TP HA FSM:FSM-CC do ChkptCcRm3
*Aug 26 18:01:22.451: : L2TP HA FSM:Received STOPCCN-ACK while waiting for it, checkpoint
CCRm and remove cc
*Aug 26 18:01:22.451: 01000:0000CD97: L2TP HA:CC ns_q cleanup: overall head Ns old/new =
4/4 (Q sz 0)
*Aug 26 18:01:22.451: 01000:0000CD97: L2TP HA FSM:Checkpoint CcRm
*Aug 26 18:01:22.451: L2TP HA CF: Chkpt send: s/c id 0/52631, CcRm, seq 0, ns/nr 3/3, rid
0, len 52; flush = 1, ctr 6
*Aug 26 18:01:22.451: 01000:0000CD97: L2TP HA FSM:Check for Ns/Nr update 4, peer 3
*Aug 26 18:01:22.451: 01000:0000CD97: L2TP HA FSM:Receive peer Ns/Nr update (4,3/3,3, int
4, rx 3, 4) (ns_q sz 0)
*Aug 26 18:01:22.451: 01000:0000CD97: L2TP HA FSM:Peer Ns 4 (4), Nr 3 (ns_q sz 0)
*Aug 26 18:01:22.451: 01000:0000CD97: L2TP HA FSM:Receive CC-Rm
*Aug 26 18:01:22.451: : L2TP HA FSM:FSM-CC ev Proto CcRm
*Aug 26 18:01:22.451: : L2TP HA FSM:FSM-CC    Wt-CkptCcRm->End
*Aug 26 18:01:22.451: : L2TP HA FSM:FSM-CC do RmCc3
*Aug 26 18:01:22.451: 01000:0000CD97: L2TP HA FSM:CC destruction after Tx/Rx StopCCN
LNS1#
*Aug 26 18:01:22.451: 01000:0000CD97: L2TP HA:CC ns_q cleanup: overall head Ns old/new =
4/4 (Q sz 0)
*Aug 26 18:01:22.451: : L2TP HA FSM:Checkpoint CCRm
*Aug 26 18:01:22.451: L2TP HA CF: Chkpt send: s/c id 0/52631, CcRm, seq 0, ns/nr 3/3, rid
0, len 52; flush = 1, ctr 7
*Aug 26 18:01:22.451: : L2TP HA:lcm_cc free: l2tp_cc 070B45B8, lcm_cc 02FE55E8
*Aug 26 18:01:22.451: L2TP tnl   _____:________: CCM setting state to DOWN
*Aug 26 18:01:23.571: : L2TP HA FSM:CHKPT status callback: status 0, len 56
*Aug 26 18:01:23.571: : L2TP HA FSM:Context s/c id 0/52631, CcRm, seq 6, ns/nr 3/3, rid 0,
 len 52
*Aug 26 18:01:23.571: : L2TP HA FSM:CHKPT status callback: status 0, len 56
*Aug 26 18:01:23.571: : L2TP HA FSM:Context s/c id 0/52631, CcRm, seq 7, ns/nr 3/3, rid 0,
 len 52
*Aug 26 18:01:23.571: L2TP HA CF: Rcvd status s/c id 0/52631, CcRm, seq 6, ns/nr 3/3, rid
0, len 52
*Aug 26 18:01:23.571: L2TP HA CF: Rcvd status 0: len 56
*Aug 26 18:01:23.571: L2TP HA CF: Status content s/c id 0/52631, CcRm, seq 6, ns/nr 3/3,
rid 0, len 52
*Aug 26 18:01:23.571: : L2TP HA FSM:Ignore chkpt ACK: CC not found.
LNS1#
*Aug 26 18:01:23.571: L2TP HA CF: Rcvd status s/c id 0/52631, CcRm, seq 7, ns/nr 3/3, rid
0, len 52
*Aug 26 18:01:23.571: L2TP HA CF: Rcvd status 0: len 56
*Aug 26 18:01:23.571: L2TP HA CF: Status content s/c id 0/52631, CcRm, seq 7, ns/nr 3/3,
rid 0, len 52
*Aug 26 18:01:23.571: : L2TP HA FSM:Ignore chkpt ACK: CC not found.
LNS1#
*Aug 26 18:01:35.771: %REDUNDANCY-3-STANDBY_LOST: Standby processor fault
(PEER_DOWN_INTERRUPT)
```

The table below describes significant fields shown in the **debug vpdn redundancy** command output.

Table 79: debug vpdn redundancy Command Field Descriptions

Field	Description
cf	Number of L2TP checkpointing-facility events (cf-events).
error	Number of L2TP checkpointing errors.
event	Number of L2TP checkpointing events.
fsm	Number of L2TP checkpointing fsm events.
resync	Number of L2TP checkpointing resynchronized events.

Field	Description
rf	Number of L2TP checkpointing redundancy-facility events (rf-events).

Related Commands

Command	Description
debug l2tp redundancy	Displays information about L2TP sessions that have redundancy events and errors.
l2tp sso enable	Enables L2TP High Availability (HA).
l2tp tunnel resync	Specifies the number of packets sent before waiting for an acknowledgement message.
show l2tp redundancy	Displays L2TP sessions containing redundancy data.
show vpdn redundancy	Displays VPDN sessions containing redundancy data.
sso enable	Enables L2TP HA for VPDN groups.

debug vpm all

To enable all voice port module (VPM) debugging, use the **debug vpm all** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug vpm all

no debug vpm all

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
11.3(1)T	This command was introduced for the Cisco 3600 series.
12.0(7)XK	This command was updated for the Cisco 2600, Cisco 3600, and Cisco MC3810 series devices.
12.1(2)T	This command was integrated into Cisco IOS Release 12.1(2)T.

Usage Guidelines

Use the **debug vpm all** command to enable the complete set of VPM debugging commands: **debug vpm dsp**, **debug vpm error**, **debug vpm port**, **debug vpm spi**, and **debug vpm trunk_sc**.

Execution of **no debug all** will turn off all port level debugging. It is usually a good idea to turn off all debugging and then enter the **debug** commands you are interested in one by one. This will help to avoid confusion about which ports you are actually debugging.

Examples

For sample outputs, refer to the documentation of the other **debup vpm** commands.

Related Commands

Command	Description
debug vpm port	Limits the **debug vpm all** command to a specified port.
show debug	Displays which debug commands are enabled.
debug vpm error	Enables DSP error tracing.

Command	Description
debug vpm voaal2 all	Enables the display of trunk conditioning supervisory component trace information.

debug vpm dsp

To show messages from the digital signal processor (DSP) on the voice port module VPM) to the router, use the **debug vpm dsp**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug vpm dsp

no debug vpm dsp

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

The **debug vpm dsp** command shows messages from the DSP on the VPM to the router; this command can be useful if you suspect that the VPM is not functional. It is a simple way to check if the VPM is responding to off-hook indications and to evaluate timing for signaling messages from the interface.

Examples

The following output shows the DSP time stamp and the router time stamp for each event. For SIG_STATUS, the state value shows the state of the ABCD bits in the signaling message. This sample shows a call coming in on an FXO interface.

The router waits for ringing to terminate before accepting the call. State=0x0 indicates ringing; state 0x4 indicates not ringing.

```
ssm_dsp_message: SEND/RESP_SIG_STATUS: state=0x0 timestamp=58172 systime=40024
ssm_dsp_message: SEND/RESP_SIG_STATUS: state=0x4 timestamp=59472 systime=40154
ssm_dsp_message: SEND/RESP_SIG_STATUS: state=0x4 timestamp=59589 systime=40166
```

The following output shows the digits collected:

```
vcsm_dsp_message: MSG_TX_DTMF_DIGIT: digit=4
vcsm_dsp_message: MSG_TX_DTMF_DIGIT: digit=1
vcsm_dsp_message: MSG_TX_DTMF_DIGIT: digit=0
vcsm_dsp_message: MSG_TX_DTMF_DIGIT: digit=0
vcsm_dsp_message: MSG_TX_DTMF_DIGIT: digit=0
```

This shows the disconnect indication and the final call statistics reported by the DSP (which are then populated in the call history table):

```
ssm_dsp_message: SEND/RESP_SIG_STATUS: state=0xC timestamp=21214 systime=42882
vcsm_dsp_message: MSG_TX_GET_TX_STAT: num_tx_pkts=1019 num_signaling_pkts=0
num_comfort_noise_pkts=0 transmit_durtation=24150 voice_transmit_duration=20380
fax_transmit_duration=0
```

debug vpm error

To enable digital signal processor (DSP) error tracing in voice port modules (VPMs), use the **debug vpm error** command in privileged EXEC mode. To disable DSP error tracing, use the **no** form of this command.

debug vpm error

no debug vpm error

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(7)XK	This command was introduced on the Cisco 2600, 3600, and MC3810 series devices.
12.1(2)T	This command was integrated into Cisco IOS Release 12.1(2)T.

Usage Guidelines

Execution of **no debug all** will turn off all port level debugging. You should turn off all debugging and then enter the **debug** commands you are interested in one by one. This will help avoid confusion about which ports you are actually debugging.

Examples

The following example shows **debug vpm error**messages for Cisco 2600 or Cisco 3600 series router or a Cisco MC3810 series concentrator:

```
Router# debug vpm error
00:18:37:[1:0.1, FXSLS_NULL, E_DSP_SIG_0100] -> ERROR:INVALID INPUT
Router#
```

The following example turns off **debug vpm error**debugging messages:

```
Router# no debug vpm error
```

Related Commands

Command	Description
debug vpm all	Enables all VPM debugging.
debug vpm port	Limits the **debug vpm error**command to a specified port.

Command	Description
show debug	Displays which debug commands are enabled.

debug vpm port

To observe the behavior of the Holst state machine, use the **debug vpm port** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug vpm port [*slot-number*| *subunit-number*| *port*]

no debug vpm port [*slot-number*| *subunit-number*| *port*]

Syntax Description

slot-number	(Optional) Specifies the slot number in the Cisco router where the voice interface card is installed. Valid entries are from 0 to 3, depending on the router being used and the slot where the voice interface card has been installed.
subunit-number	(Optional) Specifies the subunit on the voice interface card where the voice port is located. Valid entries are 0 or 1.
port	(Optional) Specifies the voice port. Valid entries are 0 or 1.

Command Modes

Privileged EXEC

Command History

Release	Modification
11.3(1)	This command was introduced.

Usage Guidelines

This command is not supported on Cisco 7200 series routers or on the Cisco MC3810.

Use this command to limit the debug output to a particular port. The debug output can be quite voluminous for a single channel. A 12-port box might create problems. Use this **debug** command with any or all of the other debug modes.

Execution of **no debug vpm all** will turn off all port level debugging. We recommend that you turn off all debugging and then enter the **debug** commands you are interested in one by one. This process helps to avoid confusion about which ports you are actually debugging.

Examples

The following is sample output from the **debug vpm port 1/1/0** command during trunk establishment after the **no shutdown** command has been executed on the voice port:

```
Router# debug vpm port 1/1/0
*Mar 1 03:21:39.799: htsp_process_event: [1/1/0, 0.1 , 2]act_down_inserve
```

```
*Mar 1 03:21:39.807: htsp_process_event: [1/1/0, 0.0 , 14]
   act_go_trunkhtsp_trunk_createhtsp_trunk_sig_linkfxols_trunk
*Mar 1 03:21:39.807: htsp_process_event: [1/1/0, 1.0 , 1]trunk_offhookfxols_trunk_down
*Mar 1 03:21:39.807: dsp_sig_encap_config: [1/1/0] packet_len=28 channel_id=128
   packet_id=42 transport_protocol=1 playout_delay=100 signaling_mode=0
   t_ssrc=0 r_ssrc=0 t_vpxcc=0 r_vpxcc=0
*Mar 1 03:21:39.811: dsp_set_sig_state: [1/1/0] packet_len=12
   channel_id=128 packet_id=39 state=0xC timestamp=0x0
*Mar 1 03:21:39.811: trunk_offhook: Trunk Retry Timer Enabled
*Mar 1 03:22:13.095: htsp_process_event: [1/1/0, 1.1, 39]act_trunk_setuphtsp_setup_ind
*Mar 1 03:22:13.095: htsp_process_event: [1/1/0, 1.2 , 8]
*Mar 1 03:22:13.099: hdsprm_vtsp_codec_loaded_ok: G726 firmware needs download
*Mar 1 03:22:13.103: dsp_download: p=0x60E73844 size=34182 (t=1213310):39 FA 6D
*Mar 1 03:22:13.103: htsp_process_event: [1/1/0, 1.2 , 6]act_trunk_proc_connect
*Mar 1 03:22:13.191: dsp_receive_packet: MSG_TX_RESTART_INDICATION: code=0 t=1213319
*Mar 1 03:22:13.191: dsp_download: p=0x60EA8924 size=6224 (t=1213319): 8 55 AE
*Mar 1 03:22:13.207: dsp_receive_packet: MSG_TX_RESTART_INDICATION: code=0 t=1213320
*Mar 1 03:22:13.207: htsp_process_event: [1/1/0, 1.3 , 11] trunk_upfxols_trunk_up
*Mar 1 03:22:13.207: dsp_set_sig_state: [1/1/0] packet_len=12
   channel_id=128 packet_id=39 state=0x4 timestamp=0x0
*Mar 1 03:22:13.207: dsp_sig_encap_config: [1/1/0] packet_len=28 channel_id=128
   packet_id=42 transport_protocol=3 playout_delay=100 headerbytes = 0xA0
```

Note in the above display that "transport_protocol = 3" indicates Voice-over-Frame Relay. Also note that the second line of the display indicates that a **shutdown/no shutdown** command sequence was executed on the voice port.

Related Commands

Command	Description
debug vpdn pppoe-data	Enables debugging of all VPM areas.
debug vpm dsp	Shows messages from the DSP on the VPM to the router.
debug vpm signal	Collects debug information only for signaling events.
debug vpm spi	Displays information about how each network indication and application request is handled.

debug vpm signal

To collect debug information only for signaling events, use the **debug vpm signal**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug vpm signal

no debug vpm signal

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

The **debug vpm signal** command collects debug information only for signaling events. This command can also be useful in resolving problems with signaling to a PBX.

Examples

The following output shows that a ring is detected, and that the router waits for the ringing to stop before accepting the call:

```
ssm_process_event: [1/0/1, 0.2, 15] fxols_onhook_ringing
ssm_process_event: [1/0/1, 0.7, 19] fxols_ringing_not
ssm_process_event: [1/0/1, 0.3, 6]
ssm_process_event: [1/0/1, 0.3, 19] fxols_offhook_clear
```

The following output shows that the call is connected:

```
ssm_process_event: [1/0/1, 0.3, 4] fxols_offhook_proc
ssm_process_event: [1/0/1, 0.3, 8] fxols_proc_voice
ssm_process_event: [1/0/1, 0.3, 5] fxols_offhook_connect
```

The following output confirms a disconnect from the switch and release with higher layer code:

```
ssm_process_event: [1/0/1, 0.4, 27] fxols_offhook_disc
ssm_process_event: [1/0/1, 0.4, 33] fxols_disc_confirm
ssm_process_event: [1/0/1, 0.4, 3] fxols_offhook_release
```

debug vpm signaling

To see information about the voice port module signaling, use the **debug vpm signaling** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug vpm signaling

no debug vpm signaling

Syntax Description

This command has no arguments or keywords

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(7)XK	This command was introduced.
12.1(2)T	This command was integrated into Release 12.1(2)T.

Examples

The following is sample output from the **debug vpm signaling** command:

```
Router# debug vpm signaling
01:52:55: [1:1.1, S_TRUNK_BUSYOUT, E_HTSP_OUT_BUSYOUT]
01:52:55: htsp_timer - 0 msec
01:52:55: [1:1.1, S_TRUNK_PEND, E_HTSP_EVENT_TIMER]
01:52:55: htsp_timer_stop htsp_setup_ind
01:52:55: htsp_timer - 2000 msec
01:52:55: [1:1.1, S_TRUNK_PROC, E_HTSP_SETUP_ACK]
01:52:55: htsp_timer_stop
01:52:55: htsp_timer - 20000 msec
01:52:55: [1:6.6, S_TRUNK_PROC, E_HTSP_SETUP_ACK]
01:52:55: htsp_timer_stop
01:52:55: htsp_timer - 20000 msec
01:52:55: [1:1.1, S_TRUNK_PROC, E_HTSP_VOICE_CUT_THROUGH]
01:52:55: %HTSP-5-UPDOWN: Trunk port(channel) [1:1.1] is up
```

debug vpm spi through voice call debug

- debug vpm spi, page 662
- debug vpm trunk_sc, page 664
- debug vpm voaal2 all, page 666
- debug vpm voaal2 type1, page 668
- debug vpm voaal2 type3, page 670
- debug vrf, page 672
- debug vrrp all, page 674
- debug vrrp authentication, page 676
- debug vrrp error, page 677
- debug vrrp events, page 679
- debug vrrp ha, page 681
- debug vrrp packets, page 683
- debug vrrp state, page 685
- debug vrrp vrrs, page 687
- debug vrrs all, page 689
- debug vrrs accounting, page 691
- debug vrrs database, page 692
- debug vrrs infra, page 694
- debug vrrs log, page 696
- debug vrrs pathway, page 697
- debug vrrs plugin, page 700
- debug vsi api, page 702
- debug vsi errors, page 704
- debug vsi events, page 707

- debug vsi packets, page 710
- debug vsi param-groups, page 712
- debug vtemplate, page 715
- debug vtemplate subinterface, page 718
- debug vtsp, page 720
- debug vtsp all, page 723
- debug vtsp dsp, page 729
- debug vtsp error, page 731
- debug vtsp event, page 733
- debug vtsp port, page 736
- debug vtsp rtp, page 739
- debug vtsp send-nse, page 742
- debug vtsp session, page 744
- debug vtsp stats, page 748
- debug vtsp tone, page 750
- debug vtsp vofr subframe, page 751
- debug vwic-mft firmware controller, page 753
- debug vxml, page 756
- debug waas, page 764
- debug waas accelerator cifs-express, page 767
- debug waas accelerator http-express, page 769
- debug waas accelerator ssl-express, page 771
- debug warm-reboot, page 773
- debug wccp, page 775
- debug webvpn, page 778
- debug webvpn dtls, page 784
- debug webvpn license, page 786
- debug wlccp ap, page 788
- debug wlccp ap rm enhanced-neighbor-list, page 790
- debug wlccp packet, page 791
- debug wlccp rmlib, page 792
- debug wlccp wds, page 793
- debug wsma agent, page 795

- debug wsma profile, page 797
- debug wsapi, page 799
- debug x25, page 802
- debug x25 annexg, page 811
- debug x25 aodi, page 813
- debug x25 interface, page 815
- debug x25 vc, page 821
- debug x25 xot, page 827
- debug x28, page 833
- debug xcctsp all, page 834
- debug xcctsp error, page 835
- debug xcctsp session, page 836
- debug xconnect, page 837
- debug xcsp, page 839
- debug xdsl application, page 842
- debug xdsl driver, page 845
- debug xdsl eoc, page 847
- debug xdsl error, page 849
- debug xmpp profile, page 851
- debug zone, page 852
- show memory debug incremental, page 854
- set memory debug incremental starting-time, page 857
- show memory debug leaks, page 859
- show memory debug references, page 867
- show memory debug unused, page 869
- show crypto debug-condition, page 871
- show debugging, page 874
- show debugging condition, page 877
- voice call debug, page 879

debug vpm spi

To trace how the voice port module security parameter index (SPI) interfaces with the call control application programming interface (API), use the **debug vpm spi**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug vpm spi

no debug vpm spi

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

The **debug vpm spi** command traces how the voice port module SPI interfaces with the call control API. This **debug** command displays information about how each network indication and application request is handled.

This debug level shows the internal workings of the voice telephony call state machine.

Examples

The following output shows that the call is accepted and presented to a higher layer code:

```
dsp_set_sig_state: [1/0/1] packet_len=14 channel_id=129 packet_id=39 state=0xC timestamp=0x0
vcsm_process_event: [1/0/1, 0.5, 1] act_up_setup_ind
```

The following output shows that the higher layer code accepts the call, requests addressing information, and starts DTMF and dial-pulse collection. It also shows that the digit timer is started.

```
vcsm_process_event: [1/0/1, 0.6, 11] act_setup_ind_ack
dsp_voice_mode: [1/0/1] packet_len=22 channel_id=1 packet_id=73 coding_type=1
voice_field_size=160 VAD_flag=0 echo_length=128 comfort_noise=1 fax_detect=1
dsp_dtmf_mode: [1/0/1] packet_len=12 channel_id=1 packet_id=65 dtmf_or_mf=0
dsp_CP_tone_on: [1/0/1] packet_len=32 channel_id=1 packet_id=72 tone_id=3 n_freq=2
freq_of_first=350 freq_of_second=440 amp_of_first=4000 amp_of_second=4000 direction=1
on_time_first=65535 off_time_first=0 on_time_second=65535 off_time_second=0
dsp_digit_collect_on: [1/0/1] packet_len=22 channel_id=129 packet_id=35 min_inter_delay=550
 max_inter_delay=3200 mim_make_time=18 max_make_time=75 min_brake_time=18 max_brake_time=75
vcsm_timer: 46653
```

The following output shows the collection of digits one by one until the higher level code indicates it has enough. The input timer is restarted with each digit and the device waits in idle mode for connection to proceed.

```
vcsm_process_event: [1/0/1, 0.7, 25] act_dcollect_digit
dsp_CP_tone_off: [1/0/1] packet_len=10 channel_id=1 packet_id=71
vcsm_timer: 47055
vcsm_process_event: [1/0/1, 0.7, 25] act_dcollect_digit
dsp_CP_tone_off: [1/0/1] packet_len=10 channel_id=1 packet_id=71
vcsm_timer: 47079
vcsm_process_event: [1/0/1, 0.7, 25] act_dcollect_digit
dsp_CP_tone_off: [1/0/1] packet_len=10 channel_id=1 packet_id=71
vcsm_timer: 47173
vcsm_process_event: [1/0/1, 0.7, 25] act_dcollect_digit
dsp_CP_tone_off: [1/0/1] packet_len=10 channel_id=1 packet_id=71
vcsm_timer: 47197
vcsm_process_event: [1/0/1, 0.7, 25] act_dcollect_digit
dsp_CP_tone_off: [1/0/1] packet_len=10 channel_id=1 packet_id=71
vcsm_timer: 47217
vcsm_process_event: [1/0/1, 0.7, 13] act_dcollect_proc
```

```
dsp_CP_tone_off: [1/0/1] packet_len=10 channel_id=1 packet_id=71
dsp_digit_collect_off: [1/0/1] packet_len=10 channel_id=129 packet_id=36
dsp_idle_mode: [1/0/1] packet_len=10 channel_id=1 packet_id=68
```

The following output shows that the network voice path cuts through:

```
vcsm_process_event: [1/0/1, 0.8, 15] act_bridge
vcsm_process_event: [1/0/1, 0.8, 20] act_caps_ind
vcsm_process_event: [1/0/1, 0.8, 21] act_caps_ack
dsp_voice_mode: [1/0/1] packet_len=22 channel_id=1 packet_id=73 coding_type=6
voice_field_size=20 VAD_flag=1 echo_length=128 comfort_noise=1 fax_detect=1
```

The following output shows that the called-party end of the connection is connected:

```
vcsm_process_event: [1/0/1, 0.8, 8] act_connect
```

The following output shows the voice quality statistics collected periodically:

```
vcsm_process_event: [1/0/1, 0.13, 17]
dsp_get_rx_stats: [1/0/1] packet_len=12 channel_id=1 packet_id=87 reset_flag=0
vcsm_process_event: [1/0/1, 0.13, 28]
vcsm_process_event: [1/0/1, 0.13, 29]
vcsm_process_event: [1/0/1, 0.13, 32]
vcsm_process_event: [1/0/1, 0.13, 17]
dsp_get_rx_stats: [1/0/1] packet_len=12 channel_id=1 packet_id=87 reset_flag=0
vcsm_process_event: [1/0/1, 0.13, 28]
vcsm_process_event: [1/0/1, 0.13, 29]
vcsm_process_event: [1/0/1, 0.13, 32]
vcsm_process_event: [1/0/1, 0.13, 17]
dsp_get_rx_stats: [1/0/1] packet_len=12 channel_id=1 packet_id=87 reset_flag=0
vcsm_process_event: [1/0/1, 0.13, 28]
vcsm_process_event: [1/0/1, 0.13, 29]
vcsm_process_event: [1/0/1, 0.13, 32]
```

The following output shows that the disconnection indication is passed to higher-level code. The call connection is torn down, and final call statistics are collected:

```
vcsm_process_event: [1/0/1, 0.13, 4] act_generate_disc
vcsm_process_event: [1/0/1, 0.13, 16] act_bdrop
dsp_CP_tone_off: [1/0/1] packet_len=10 channel_id=1 packet_id=71
vcsm_process_event: [1/0/1, 0.13, 18] act_disconnect
dsp_get_levels: [1/0/1] packet_len=10 channel_id=1 packet_id=89
vcsm_timer: 48762
vcsm_process_event: [1/0/1, 0.15, 34] act_get_levels
dsp_get_tx_stats: [1/0/1] packet_len=12 channel_id=1 packet_id=86 reset_flag=1
vcsm_process_event: [1/0/1, 0.15, 31] act_stats_complete
dsp_CP_tone_off: [1/0/1] packet_len=10 channel_id=1 packet_id=71
dsp_digit_collect_off: [1/0/1] packet_len=10 channel_id=129 packet_id=36
dsp_idle_mode: [1/0/1] packet_len=10 channel_id=1 packet_id=68
vcsm_timer: 48762
dsp_set_sig_state: [1/0/1] packet_len=14 channel_id=129 packet_id=39 state=0x4 timestamp=0x0
vcsm_process_event: [1/0/1, 0.16, 5] act_wrelease_release
dsp_CP_tone_off: [1/0/1] packet_len=10 channel_id=1 packet_id=71
dsp_idle_mode: [1/0/1] packet_len=10 channel_id=1 packet_id=68
dsp_get_rx_stats: [1/0/1] packet_len=12 channel_id=1 packet_id=87 reset_flag=1
```

debug vpm trunk_sc

To enable the display of trunk conditioning supervisory component trace information, use the **debug vpm trunk_sc** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug vpm trunk_sc

no debug vpm trunk_sc

Syntax Description

This command has no arguments or keywords.

Command Default

Trunk conditioning supervisory component trace information is not displayed.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(7)XK	This command was introduced on the Cisco 2600, Cisco 3600, and Cisco MC3810 series devices.
12.1(2)T	This command was integrated into Cisco IOS Release 12.1(2)T.

Usage Guidelines

Use the **debug vpm port** command with the *slot-number/subunit-number/port* argument to limit the **debug vpm trunk_sc** debug output to a particular port. If you do not use the **debug vpm port**command, the **debug vpm trunk_sc**displays output for all ports.

Execution of the **no debug all** command will turn off all port level debugging. It is usually a good idea to turn off all debugging and then enter the **debug** commands you are interested in one by one. This process helps avoid confusion about which ports you are actually debugging.

Examples

The following example shows **debug vpm trunk_sc** messages for port 1/0/0 on a Cisco 2600 or Cisco 3600 series router:

```
Router# debug vpm trunk_sc
Router# debug vpm port 1/0/0
```

The following example shows **debug vpm trunk_sc** messages for port 1/1 on a Cisco MC3810 device:

```
Router# debug vpm trunk_sc
Router# debug vpm port 1/1
```

The following example turns off **debug vpm trunk_sc** debugging messages:

```
Router# no debug vpm trunk_sc
```

Related Commands

Command	Description
debug vpm all	Enables all VPM debugging
debug vpm port	Limits the **debug vpm trunk_sc** command to a specified port.
show debug	Displays which debug commands are enabled.

debug vpm voaal2 all

To display type 1 (voice) and type 3 (control) ATM Adaptation Layer type 2 (AAL2) packets sent to and received from the domain-specific part (DSP), use the **debug vpm voaal2 all** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug vpm voaal2 all {**all_dsp| from_dsp| to_dsp**}

no debug vpm voaal2 all

Syntax Description

all_dsp	Displays messages to and from the DSP.
from_dsp	Displays messages from the DSP.
to_dsp	Displays messages to the DSP.

Command Default

Debugging for display of AAL2 packets is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.1(1)XA	This command was introduced for the Cisco MC3810 series.
12.1(2)T	This command was integrated in Cisco IOS Release 12.1(2)T.
12.2(2)T	Support for this command was integrated on the Cisco 7200 series.

Usage Guidelines

Do not enter this **debug** command on a system carrying live traffic. Continuous display of AAL2 type 1 (voice) packets results in high CPU utilization and loss of console access to the system. Calls will be dropped and trunks may go down. For AAL2 debugging, use the **debug vpm voaal2 type3** debug command and identify a specific type 3 (control) packet type.

Examples

The following is sample output from the **debug vpm voaal2 all** command, where the example selection is to display channel-associated switching (CAS) packets sent to and from the DSP:

```
Router# debug vpm voaal2 all all_dsp
*Jan  9 20:10:36.965:TYPE 3, len = 8, cid = 34, uui = 24 :TO_DSP
*Jan  9 20:10:36.965:CAS
 redundancy = 3, timestamp = 10270, signal = 0
- 22 13 12 E8 1E 0 E 15 -
*Jan  9 20:10:41.617:TYPE 3, len = 8, cid = 34, uui = 24 :FROM_DSP
*Jan  9 20:10:41.617:CAS
```

```
 redundancy = 3, timestamp = 980, signal = 0
- 22 13 12 C3 D4 0 F 87 -
*Jan  9 20:10:41.965:TYPE 3, len = 8, cid = 34, uui = 24 :TO_DSP
*Jan  9 20:10:41.965:CAS
 redundancy = 3, timestamp = 10270, signal = 0
- 22 13 12 E8 1E 0 E 15 -
*Jan  9 20:10:46.621:TYPE 3, len = 8, cid = 34, uui = 24 :FROM_DSP
*Jan  9 20:10:46.621:CAS
 redundancy = 3, timestamp = 980, signal = 0
- 22 13 12 C3 D4 0 F 87 -
....
*Jan  9 20:10:57.101:TYPE 1, len = 43, cid = 34, uui = 8- 22 9D 1 CC FC
C7
 3E 22 23 FE DF F8 DE 1C FF E5 12 22 43 EC 2E 9E CC DE A7 EF 14 E3 F1 2C
2D
 BC 1B FC FE D7 E1 1F 2F ED 11 FC 1F -
*Jan  9 20:10:57.105:TYPE 3, len = 9, cid = 34, uui = 24 :FROM_DSP
*Jan  9 20:10:57.105:DIALED DIGITS
 redundancy = 0,
                   timestamp = 940, digitcode = 1
- 22 17 3 3 AC 1 1 8 E5 -
*Jan  9 20:10:57.113:TYPE 1, len = 43, cid = 34, uui = 10- 22 9D 4B 3F
1F
11 FC CD CC BE B7 E2 F3 32 2E 1F F9 DA CC BF 12 F1 37 31 11 2C FE 9D DA
D2
E1 C7 4A 34 3F FA 21 AD CC 1F EE 16 E1 -
*Jan  9 20:10:57.113:TYPE 3, len = 9, cid = 34, uui = 24 :FROM_DSP
*Jan  9 20:10:57.113:DIALED DIGITS
 redundancy = 1,
                   timestamp = 940, digitcode = 1
- 22 17 3 43 AC 1 1 B 12 -
*Jan  9 20:10:57.121:TYPE 1, len = 43, cid = 34, uui = 12- 22 9D 95 F1
1E
E1 DF 1E 21 31 21 1D D9 EB BB DF 22 17 13 12 1F 58 FF ED ED E1 4D B7 3E
3F
21 F3 8E FD EF DF F4 12 E4 32 FE B4 D8 -
```

Related Commands

Command	Description
debug vpm voaal2 type1	Displays type 1 (voice) AAL2 packets sent to and received from the DSP.
debug vpm voaal2 type3	Displays type 3 (control) AAL2 packets sent to and received from the DSP.
show debug	Shows which debug commands are enabled.

debug vpm voaal2 type1

To display type 1 (voice) ATM Adaptation Layer type 2 (AAL2) packets sent to and received from the domain-specific part (DSP), use the **debug vpm voaal2 type1** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug vpm voaal2 type1 {**all_dsp**| **from_dsp**| **to_dsp**}

no debug vpm voaal2 type1

Syntax Description

all_dsp	Displays messages to and from the DSP.
from_dsp	Displays messages from the DSP.
to_dsp	Displays messages to the DSP.

Command Default

Debugging for display of AAL2 packets is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.1(1)XA	This command was introduced for the Cisco MC3810 series.
12.1(2)T	This command was integrated in Cisco IOS Release 12.1(2)T.
12.2(2)T	Support for this command was implemented on the Cisco 7200 series.

Usage Guidelines

Do not enter this **debug** command on a system carrying live traffic. Continuous display of AAL2 type 1 (voice) packets results in high CPU utilization and loss of console access to the system. Calls will be dropped and trunks may go down. For AAL2 debugging, use the **debug vpm voaal2 type 3** command and identify a specific type 3 (control) packet type.

Examples

The following is sample output from the **debug vpm voaal2 type1** command:

Note The display of voice packets on a live system will continue indefinitely. The debugging output cannot be interrupted, because console access will be lost.

```
Router# debug vpm voaal2 type1 all_dsp
```

```
TYPE 1, len = 43, cid = 17, uui = 15- 11 9D E6 1B 52 9D 95 9B DB 1D 14
1C 5F 9C 95 9C EA 1C 15 1B 74 9C 94 9D 6B 1C 14 1D E4 9B 94 9D 5B 1B 14
1D D7 9B 94 9D 50 1B 14 -
TYPE 1, len = 43, cid = 22, uui = 15- 16 9D ED 1D 14 1B 53 9D 94 9C DB
1D 14 1C 5F 9C 95 9C EB 1C 14 1C 78 9D 94 9D 6F 1C 14 1E E4 9B 94 9D 5B
1B 14 1D D7 9B 94 9E 52 -
TYPE 1, len = 43, cid = 12, uui = 14- C 9D D1 29 AB 96 96 A9 2B 16 16 2A
AA 96 96 AB 2A 16 17 2B A9 96 97 AC 28 16 17 2C A8 96 97 AD 27 15 17 2E
A7 97 97 AE 26 16 17 -
TYPE 1, len = 43, cid = 34, uui = 14- 22 9D DF D7 31 20 19 15 14 15 19
1E 2C 60 AF 9F 99 96 94 95 99 9F AD EC 2F 1F 1A 15 14 15 19 1F 2E ED AD
9F 99 96 93 95 99 9F AF -
TYPE 1, len = 43, cid = 12, uui = 15- C 9D F4 2F A5 96 97 AF 25 15 18 31
A4 95 98 B3 23 15 18 33 A3 95 98 B5 22 15 18 37 A2 95 98 B7 21 15 18 39
A0 95 99 BB 21 14 19 -
TYPE 1, len = 43, cid = 34, uui = 15- 22 9D FA 5D 2D 1E 19 15 14 15 1A
21 31 D9 AC 9E 98 95 94 95 9A A4 B3 52 2B 1D 18 14 14 16 1B 22 36 CA AA
9D 98 94 94 96 9B A4 B6 -
```

Related Commands

Command	Description
debug vpm all	Enables all VPM debugging.
debug vpm voaal2 all	Displays type 1 (voice) and type 3 (control) AAL2 packets sent to and received from the DSP.
debug vpm voaal2 type3	Displays type 3 (control) AAL2 packets sent to and received from the DSP.
show debug	Shows which debug commands are enabled.

debug vpm voaal2 type3

To display type 3 (control) ATM Adaptation Layer type 2 (AAL2) packets sent to and received from the domain-specific part (DSP), use the **debug vpm voaal2 type3** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug vpm voaal2 type3 {**alarms**| **alltype3**| **cas**| **dialed**| **faxrelay**| **state**} {**all_dsp**| **from_dsp**| **to_dsp**}

no debug vpm voaal2 type3

Syntax Description

alarms	Displays type 3 alarm packets.
alltype3	Displays all type 3 packets.
cas	Displays type 3 channel-associated switching (CAS) packets.
dialed	Displays type 3 dialed digit packets.
faxrelay	(Not supported) Displays type 3 fax relay packets.
state	Displays type 3 user state packets.
all_dsp	Displays messages to and from the DSP.
from_dsp	Displays messages from the DSP.
to_dsp	Displays messages to the DSP.

Command Default

Debugging for display of AAL2 packets is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.1(1)XA	This command was introduced for the Cisco MC3810 series.
12.1(2)T	This command was integrated in Cisco IOS Release 12.1(2)T.
12.2(2)T	Support for this command was implemented on the Cisco 7200 series.

Usage Guidelines

This is the preferred **debug** command for displaying specific types of control packets. It is usually preferable to specify a particular type of control packet rather than use the **alltype3** to avoid excessive output display and CPU utilization.

Examples

The following is sample output from the **debug vpm voaal2 type3** command, where the example selection is to display messages to and from the DSP:

```
Router# debug vpm voaal2 type3 all_dsp
00:43:02:TYPE 3, len = 8, cid = 58, uui = 24 :TO_DSP
00:43:02:CAS
 redundancy = 3, timestamp = 10484, signal = 0
- 3A 13 18 E8 F4 0 C DA -
00:43:02:TYPE 3, len = 8, cid = 93, uui = 24 :FROM_DSP
00:43:02:CAS
 redundancy = 3, timestamp = 6528, signal = 0
- 5D 13 1E D9 80 0 F 33 -
00:43:02:TYPE 3, len = 8, cid = 102, uui = 24 :FROM_DSP
00:43:02:CAS
 redundancy = 3, timestamp = 5988, signal = 0
- 66 13 4 D7 64 0 F DF -
00:43:02:TYPE 3, len = 8, cid = 194, uui = 24 :FROM_DSP
00:43:02:CAS
 redundancy = 3, timestamp = 6212, signal = 0
- C2 13 10 D8 44 0 F AC -
00:43:02:TYPE 3, len = 8, cid = 92, uui = 24 :FROM_DSP
TYPE 3, len = 8, cid = 66, uui = 24 :TO_DSP:43:00:CAS
 redundancy = 3, times signal = 0
- 5C 13 5 D9 E4 0 C 1F -
00:43:02:TYPE 3, len = 8, cid = 40, uui = 24 :TO_DSP
00:43:02:CAS
 redundancy = 3, timestamp = 8658, signal = 0
- 28 13 7 E1 D2 0 E 79 -
00:43:02:TYPE 3, len = 8, cid = 137, uui = 24 :FROM_DSP
00:43:02:CAS
 redundancy = 3, timestamp = 6836, signal = 0
- 89 13 B DA B4 0 E 78 -
```

Related Commands

Command	Description
debug vpm voaal2 type1	Displays type 1 (voice) AAL2 packets sent to and received from the DSP.
debug vpm voaal2 type3	Displays type 3 (control) AAL2 packets sent to and received from the DSP.
show debug	Shows which debug commands are enabled.

debug vrf

To get debugging information on virtual routing and forwarding (VRF) instances, use the **debug vrf**command in privileged EXEC mode. To turn off the debug output, use the **undebug** version of the command.

debug vrf {**create**| **delete**| **error**| **ha**| **initialization**| **interface**| **ipv4**| **ipv6**| **issu**| **lock**| **lookup**| **mpls**| **selection**}

undebug vrf {**create**| **delete**| **error**| **ha**| **initialization**| **interface**| **ipv4**| **ipv6**| **issu**| **lock**| **lookup**| **mpls**| **selection**}

Syntax Description

create	Specifies VRF creation debugging.
delete	Specifies VRF deletion debugging.
error	Specifies VRF error debugging.
ha	Specifies VRF high-availability debugging.
initialization	Specifies VRF subsystem initialization debugging.
interface	Specifies VRF interface assignment debugging.
ipv4	Specifies VRF IPv4 address family debugging.
ipv6	Specifies VRF IPv6 address family debugging.
issu	Specifies VRF in-service software upgrade debugging.
lock	Specifies VRF lock debugging.
lookup	Specifies VRF database lookup debugging.
mpls	Specifies VRF multiprotocol label switching debugging.
selection	Specifies VRF selection debugging.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS XE Release 3.2S	This command was introduced.

Usage Guidelines

Use this command to get debugging information on VRFs.

Examples

The following example shows how to turn on debugging of VRF interface assignment:

```
Router# debug vrf interface
```

Related Commands

Command	Description
vrf definition	Defines a virtual routing and forwarding instance.

debug vrrp all

To display debugging messages for Virtual Router Redundancy Protocol (VRRP) errors, events, and state transitions, use the **debug vrrp all** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug vrrp all

no debug vrrp all

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.0(18)ST	This command was introduced.
12.0(22)S	This command was integrated into Cisco IOS Release 12.0(22)S.
12.2(15)T	This command was integrated into Cisco IOS Release 12.2(15)T.
12.2(14)S	This command was integrated into Cisco IOS Release 12.2(14)S.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(31)SB2	This command was integrated into Cisco IOS Release 12.2(31)SB2.
12.2(31)SG	This command was integrated into Cisco IOS Release 12.2(31)SG.
12.2(17d)SXB	This command was integrated into Cisco IOS Release 12.2(17d)SXB.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.
Cisco IOS XE Release 2.1	This command was integrated into Cisco IOS XE Release 2.1.
Cisco IOS XE Release 2.6	This command was modified. This output was modified to display VRRP debugging statements for Virtual Router Redundancy Service (VRRS).

Examples

The following is sample output from the **debug vrrp all**command:

```
Router# debug vrrp all
00:15:30: %IP-4-DUPADDR: Duplicate address 10.18.0.2 on Ethernet1/0, sourced by 0000.5e00.0101

May 22 18:41:54.447: VRRP: Grp 1 Advertisement Primary address 10.18.0.2
         different from ours 10.18.0.1
May 22 18:41:57.443: VRRP: Grp 1 Advertisement Primary address 10.18.0.2
```

```
        different from ours 10.18.0.1
May 22 18:42:00.443: VRRP: Grp 1 Advertisement Primary address 10.18.0.2
        different from ours 10.18.0.1
May 22 18:48:41.521: VRRP: Grp 1 Event - Advert higher or equal priority
May 22 18:48:44.521: VRRP: Grp 1 Event - Advert higher or equal priority
May 22 18:48:47.521: VRRP: Grp 1 Event - Advert higher or equal priority
May 22 18:53:23.390: VRRP: Grp 1 changing to V_STATE_INIT
May 22 18:54:26.143: VRRP: Grp 1 changing to V_STATE_BACKUP
May 22 18:54:35.755: VRRP: Grp 1 changing to V_STATE_MASTER
May 22 18:53:23.390: VRRP: Grp 1 changing to V_STATE_INIT
May 22 18:54:26.143: VRRP: Grp 1 changing to V_STATE_BACKUP
May 22 18:54:35.755: VRRP: Grp 1 changing to V_STATE_MASTER
```

Related Commands

Command	Description
debug vrrp error	Displays debugging messages about VRRP error conditions.
debug vrrp events	Displays debugging messages about VRRP events.
debug vrrp state	Displays debugging messages about the VRRP state transitions.

debug vrrp authentication

To display debugging messages for Virtual Router Redundancy Protocol (VRRP) Message Digest 5 (MD5) authentication, use the **debug vrrp authentication**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug vrrp authentication

no debug vrrp authentication

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.3(14)T	This command was introduced.

Examples

The following sample output shows that MD5 authentication is enabled on one router but not the other:

```
Router# debug vrrp authentication
VRRP: Grp 1 Adv from 172.24.1.2 has incorrect auth type 1 expected 0
```

The following sample output shows that the MD5 key IDs and key strings differ on each router:

```
Router# debug vrrp authentication
VRRP: Sent: 21016401FE050000AC1801FE0000000000000000
VRRP: HshC: B861CBF1B9026130DD34AED849BEC8A1
VRRP: Rcvd: 21016401FE050000AC1801FE0000000000000000
VRRP: HshC: B861CBF1B9026130DD34AED849BEC8A1
VRRP: HshR: C5E193C6D84533FDC750F85FCFB051E1
VRRP: Grp 1 Adv from 172.24.1.2 has failed MD5 auth
```

The following sample output shows that the text authentication strings differ on each router:

```
Router# debug vrrp authentication
VRRP: Grp 1 Adv from 172.24.1.2 has failed TEXT auth
```

Related Commands

Command	Description
debug vrrp error	Displays debugging messages about VRRP error conditions.
debug vrrp events	Displays debugging messages about VRRP events.
debug vrrp state	Displays debugging messages about the VRRP state transitions.

debug vrrp error

To display debugging messages about Virtual Router Redundancy Protocol (VRRP) error conditions, use the **debug vrrp error** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug vrrp error

no debug vrrp error

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.0(18)ST	This command was introduced.
12.0(22)S	This command was integrated into Cisco IOS Release 12.0(22)S.
12.2(15)T	This command was integrated into Cisco IOS Release 12.2(15)T.
12.2(14)S	This command was integrated into Cisco IOS Release 12.2(14)S.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(31)SB2	This command was integrated into Cisco IOS Release 12.2(31)SB2.
12.2(31)SG	This command was integrated into Cisco IOS Release 12.2(31)SG.
12.2(17d)SXB	This command was integrated into Cisco IOS Release 12.2(17d)SXB.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.
Cisco IOS XE Release 2.1	This command was integrated into Cisco IOS XE Release 2.1.

Examples

The following is sample output from the **debug vrrp error**command:

```
Router# debug vrrp error
00:15:30: %IP-4-DUPADDR: Duplicate address 10.18.0.2 on Ethernet1/0, sourced by 0000.5e00.0101

May 22 18:41:54.447: VRRP: Grp 1 Advertisement Primary address 10.18.0.2
         different from ours 10.18.0.1
May 22 18:41:57.443: VRRP: Grp 1 Advertisement Primary address 10.18.0.2
        different from ours 10.18.0.1
May 22 18:42:00.443: VRRP: Grp 1 Advertisement Primary address 10.18.0.2
        different from ours 10.18.0.1
```

In the example, the error being observed is that the router has a virtual address of 10.18.0.1 for group 1, but it received a virtual address of 10.18.0.2 for group 1 from another router on the same LAN.

Related Commands

Command	Description
debug vrrp all	Displays debugging messages for VRRP errors, events, and state transitions.

debug vrrp events

To display debugging messages about Virtual Router Redundancy Protocol (VRRP) events that are occurring, use the **debug vrrp events** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug vrrp events

no debug vrrp events

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.0(18)ST	This command was introduced.
12.0(22)S	This command was integrated into Cisco IOS Release 12.0(22)S.
12.2(15)T	This command was integrated into Cisco IOS Release 12.2(15)T.
12.2(14)S	This command was integrated into Cisco IOS Release 12.2(14)S.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(31)SB2	This command was integrated into Cisco IOS Release 12.2(31)SB2.
12.2(31)SG	This command was integrated into Cisco IOS Release 12.2(31)SG.
12.2(17d)SXB	This command was integrated into Cisco IOS Release 12.2(17d)SXB.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.
Cisco IOS XE Release 2.1	This command was integrated into Cisco IOS XE Release 2.1.

Examples

The following is sample output from the **debug vrrp events**command:

```
Router# debug vrrp events
May 22 18:48:41.521: VRRP: Grp 1 Event - Advert higher or equal priority
May 22 18:48:44.521: VRRP: Grp 1 Event - Advert higher or equal priority
May 22 18:48:47.521: VRRP: Grp 1 Event - Advert higher or equal priority
```

In the example, the event being observed is that the router received an advertisement from another router for group 1 that has a higher or equal priority to itself.

Related Commands

Command	Description
debug vrrp all	Displays debugging messages for VRRP errors, events, and state transitions.

debug vrrp ha

To display debugging messages for Virtual Router Redundancy Protocol (VRRP) high availability, use the **debug vrrp ha**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug vrrp ha

no debug vrrp ha

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(33)SRC	This command was introduced.
12.2(33)SB2	This command was integrated into Cisco IOS Release 12.2(33)SB2.
Cisco IOS XE Release 2.1	This command was integrated into Cisco IOS XE Release 2.1.
12.2(33)SXI	This command was integrated into Cisco IOS Release 12.2(33)SXI.

Examples

The following examples for the **debug vrrp ha**command display the syncing of VRRP state information from the Active RP to the Standby RP.

The following sample output displays two VRRP state changes on the Active RP:

```
Router# debug vrrp ha
.
.
.
*Nov 14 11:36:50.272 UTC: VRRP: Gi3/2 Grp 42 RF Encode state Backup into sync buffer
*Nov 14 11:36:50.272 UTC: %VRRP-6-STATECHANGE: Gi3/2 Grp 42 state Init -> Backup
*Nov 14 11:36:53.884 UTC: VRRP: Gi3/2 Grp 42 RF Encode state Master into sync buffer
*Nov 14 11:36:53.884 UTC: %VRRP-6-STATECHANGE: Gi3/2 Grp 42 state Backup -> Master
```

The following sample output displays two VRRP state changes on the Standby RP:

```
Router# debug vrrp ha
.
.
.
*Nov 14 11:36:50.392 UTC: STDBY: VRRP: Gi3/2 Grp 42 RF sync state Init -> Backup
*Nov 14 11:36:53.984 UTC: STDBY: VRRP: Gi3/2 Grp 42 RF sync state Backup -> Master
```

Related Commands

Command	Description
debug vrrp error	Displays debugging messages about VRRP error conditions.
debug vrrp events	Displays debugging messages about VRRP events.
debug vrrp state	Displays debugging messages about the VRRP state transitions.

debug vrrp packets

To display summary information about Virtual Router Redundancy Protocol (VRRP) packets being sent or received, use the **debug vrrp packets** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug vrrp packets

no debug vrrp packets

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(18)ST	This command was introduced.
12.0(22)S	This command was integrated into Cisco IOS Release 12.0(22)S.
12.2(15)T	This command was integrated into Cisco IOS Release 12.2(15)T.
12.2(14)S	This command was integrated into Cisco IOS Release 12.2(14)S.

Command History

12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(31)SB2	This command was integrated into Cisco IOS Release 12.2(31)SB2.
12.2(31)SG	This command was integrated into Cisco IOS Release 12.2(31)SG.
12.2(17d)SXB	This command was integrated into Cisco IOS Release 12.2(17d)SXB.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.
Cisco IOS XE Release 2.1	This command was integrated into Cisco IOS XE Release 2.1.

Examples

The following is sample output from the **debug vrrp packets**command. The output is on the master virtual router; the router for group 1 is sending an advertisement with a checksum of 6BE7.

```
Router# debug vrrp packets
VRRP Packets debugging is on
May 22 18:51:03.220: VRRP: Grp 1 sending Advertisement checksum 6BE7
May 22 18:51:06.220: VRRP: Grp 1 sending Advertisement checksum 6BE7
```

In the following example, the router with physical address 10.18.0.3 is advertising a priority of 105 for VRRP group 1:

```
Router# debug vrrp packets
VRRP Packets debugging is on
May 22 18:51:09.222: VRRP: Grp 1 Advertisement priority 105, ipaddr 10.18.0.3
May 22 18:51:12.222: VRRP: Grp 1 Advertisement priority 105, ipaddr 10.18.0.3
```

debug vrrp state

To display debugging messages about the state transitions occurring for Virtual Router Redundancy Protocol (VRRP) groups, use the **debug vrrp state** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug vrrp state

no debug vrrp state

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.0(18)ST	This command was introduced.
12.0(22)S	This command was integrated into Cisco IOS Release 12.0(22)S.
12.2(15)T	This command was integrated into Cisco IOS Release 12.2(15)T.
12.2(14)S	This command was integrated into Cisco IOS Release 12.2(14)S.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(31)SB2	This command was integrated into Cisco IOS Release 12.2(31)SB2.
12.2(31)SG	This command was integrated into Cisco IOS Release 12.2(31)SG.
12.2(17d)SXB	This command was integrated into Cisco IOS Release 12.2(17d)SXB.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.
Cisco IOS XE Release 2.1	This command was integrated into Cisco IOS XE Release 2.1.

Examples

The following is sample output from the **debug vrrp state**command:

```
Router# debug vrrp state
May 22 18:53:23.390: VRRP: Grp 1 changing to V_STATE_INIT
May 22 18:54:26.143: VRRP: Grp 1 changing to V_STATE_BACKUP
May 22 18:54:35.755: VRRP: Grp 1 changing to V_STATE_MASTER
```

Related Commands

Command	Description
debug vrrp all	Displays debugging messages for VRRP errors, events, and state transitions.

debug vrrp vrrs

To enable Virtual Router Redundancy Protocol (VRRP) debugging statements for Virtual Router Redundancy Service (VRRS) interactions, use the **debug vrrp vrrs** command in privileged EXEC mode. To disable VRRP VRRS debugging statements, use the **no** form of this command.

debug vrrp vrrs

no debug vrrp vrrs

Syntax Description

This command has no arguments or keywords.

Command Default

VRRP debugging for VRRS interactions is not enabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS XE Release 2.6	This command was introduced.

Examples

The following is sample output from the **debug vrrp vrrs** command:

```
Router# debug vrrp vrrs
VRRP VRRS debugging is on
The following is sample output from the debug vrrp vrrs
 command when a VRRP group is configured with a name association to ‘name1’:
Router# configure termina
l
Router(config)# interface gigabitethernet0/0/0
Router(config-if)# ip address 10.0.0.1 255.0.0.0
Router(config-if)# vrrp 1 ip 10.0.0.7
Router(config-if)# vrrp 1 name name1
*Feb  5 09:29:47.005: VRRP: Registered VRRS group "name1"
```

The following is sample output when a VRRP group is brought up:

```
Router(config-if)# no shutdown
*Feb  5 09:29:53.237: VRRP: Updated info for VRRS group name1
```

The following is sample output when a name association is changed to a different name:

```
Router(config-if)# vrrp 1 name name2
*Feb  5 09:30:14.153: VRRP: Unregistered VRRS group "name1"
*Feb  5 09:30:14.153: VRRP: Registered VRRS group "name2"
```

The following is sample output when a name association for group is removed:

```
Router(config-if)# no vrrp 1 name
*Feb  5 09:30:22.689: VRRP: Unregistered VRRS group "name2"
```

Related Commands

Command	Description
debug vrrs accounting	Enables debug messages for VRRS accounting.
debug vrrs infra	Enables VRRS infrastructure debug messages.
debug vrrs plugin	Enables VRRS plug-in debug messages.

debug vrrs all

To enable debugging information associated with all elements of Virtual Router Redundancy Service (VRRS), use the **debug vrrs all** command in Privileged EXEC mode.

debug vrrs all [**detail**]

Syntax Description

detail	(Optional) Enables detailed debugging information associated with VRRS pathways and databases.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
15.3(1)S	This command was introduced.
Cisco IOS XE Release 3.8S	This command was introduced.

Usage Guidelines

You must configure the VRRS control groups using the **vrrs** command on interfaces that require a redundant virtual gateway.

Examples

The following example shows how to enable debugging information associated with all elements of VRRS using the **debug vrrs all** command:

```
Device# debug vrrs all

vrrs database client debugging is on
vrrs database error debugging is on
vrrs database event debugging is on
vrrs database server debugging is on
vrrs database tag debugging is on
vrrs pathway event debugging is on
vrrs pathway database debugging is on
vrrs pathway error debugging is on
vrrs pathway mac debugging is on
vrrs pathway address resolution protocol debugging is on
vrrs pathway process debugging is on
vrrs pathway state debugging is on
vrrs pathway address debugging is on
```

Related Commands

Command	Description
debug vrrs log	Enables debugging information associated with VRRS logs.
debug vrrs database	Enables debugging information associated with VRRS databases.
debug vrrs pathway	Enables debugging information associated with VRRS pathways.

debug vrrs accounting

To enable debug messages for Virtual Router Redundancy Service (VRRS) accounting, use the **debug vrrs accounting** command in privileged EXEC mode. To disable VRRS accounting debug messages, use the **no** form of this command.

debug vrrs accounting {**all**| **errors**| **events**}

no debug vrrs accounting command {**all**| **errors**| **events**}

Syntax Description

all	Enables all VRRS accounting debug messages.
errors	Enables VRRS accounting error debug messages.
events	Enables VRRS accounting event debug messages.

Command Default

VRRS accounting debug messages are not displayed.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS XE Release 2.6	This command was introduced.

Examples

The following example turns on all VRRS accounting debug messages:

```
Router# debug vrrs accounting all
00:16:13: VRRS/ACCT/EV: entry create for abc(0x4E8C1F0)
00:16:13: VRRS/ACCT/EV: abc(0x4E8C1F0 12000006) client add ok2(No group)
```

Related Commands

Command	Description
debug vrrp vrrs	Enables VRRP debugging statements for VRRS interactions.
debug vrrs accounting	Enables debug messages for VRRS accounting.
debug vrrs infra	Enables VRRS infrastructure debug messages.
debug vrrs plugin	Enables VRRS plug-in debug messages.

debug vrrs database

To enable debugging information associated with the Virtual Router Redundancy Services (VRRS) database, use the **debug vrrs database** command in Privileged EXEC mode.

debug vrrs database {**all** [**detail**]| {**client**| **error**| **event**| **server**| **tag**} [**Ethernet** *number* [**IPv4** [**verbose**]| **IPv6** [**verbose**]]| **IPv4** [**Ethernet** *number* [**verbose**]| **verbose** [**Ethernet** *number*]]| **IPv6** [**Ethernet** *number* [**verbose**]| **verbose** [**Ethernet** *number*]]] [**detail**]}

Syntax Description

all	Enables debugging information associated with all VRRS databases.
detail	(Optional) Enables detailed debugging information associated with all VRRS databases.
client	Enables debugging information associated with VRRS database clients.
error	Enables debugging information associated with VRRS database errors.
event	Enables debugging information associated with VRRS database events.
server	Enables debugging information associated with VRRS database servers.
tag	Enables debugging information associated with VRRS database tags.
Ethernet *number*	(Optional) Enables debugging information associated with VRRS database for ethernet interfaces.
IPv4	(Optional) Enables debugging information associated with VRRS database for VRRP groups adhering to IPv4 protocol.
verbose	(Optional) Enables debugging information associated with VRRS database for groups adhering to non-protocol events.
IPv6	(Optional) Enables debugging information associated with VRRS database for VRRP groups adhering to IPv6 protocol.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
15.3(1)S	This command was introduced.
Cisco IOS XE Release 3.8S	This command was introduced.

Usage Guidelines

You must configure the VRRS control groups using the **vrrs** command on interfaces that require a redundant virtual gateway.

Examples

The following example shows how to enable debugging information associated with all elements of VRRS database using the **debug vrrs database** command with the **all** keyword:

```
Device# debug vrrs database all

vrrs database client debugging is on
vrrs database error debugging is on
vrrs database event debugging is on
vrrs database server debugging is on
vrrs database tag debugging is on
```

Related Commands

Command	Description
debug vrrs all	Enables debugging information associated with all elements of Virtual Router Redundancy Service (VRRS).
debug vrrs log	Enables debugging information associated with VRRS logs.
debug vrrs pathway	Enables debugging information associated with VRRS pathways.

debug vrrs infra

To enable Virtual Router Redundancy Service (VRRS) infrastructure debug messages, use the **debug vrrs infra** command in privileged EXEC mode. To turn off VRRS infrastructure debugging, use the **no** form of this command.

debug vrrs infra {**all**| **client**| **events**| **server**}

no debug vrrs infra {**all**| **client**| **events**| **server**}

Syntax Description

all	Enables all VRRS infrastructure debug messages.
client	Enables debugging for VRRS infrastructure to VRRS client interactions.
events	Enables debugging for VRRS infrastructure events.
server	Enables debugging for VRRS infrastructure to VRRS server interactions.

Command Default

VRRS debugging is disabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS XE Release 2.6	This command was introduced.

Examples

The following is sample output from the **debug vrrs infra**command:

```
Router# debug vrrs infra all
*Sep 9 16:09:53.848: VRRS: Client 21 is not registered
*Sep 9 16:09:53.848: VRRS: Client 21 unregister failed
*Sep 9 16:09:53.848: VRRS: Client VRRS TEST CLIENT registered, id 21
*Sep 9 16:09:53.848: VRRS: Client 21 add, group VRRP-TEST-1 does not exist, allocating...
*Sep 9 16:09:53.848: VRRS: Client 21 add to VRRP-TEST-1. Vrrs handle F7000001, client handle
 FE720
*Sep 9 16:09:53.848: VRRS: Server VRRP add, group VRRP-TEST-1, state INIT, vrrs handle
F7000001
*Sep 9 16:09:53.876: VRRS: VRRP-TEST-1 group added notification
*Sep 9 16:09:53.876: VRRS: Normal priority clients for group 200000, for all groups[4C0
*Sep 9 16:09:53.876: VRRS: Client 2 add to VRRP-TEST-1. Vrrs handle F7000001, client handle
 22766F0
*Sep 9 16:09:54.356: VRRS: Client 21 remove from VRRP-TEST-1. vrrs handle F7000001
*Sep 9 16:09:54.356: VRRS: Server VRRP delete, group VRRP-TEST-1 vrrs handle F7000001
*Sep 9 16:09:54.360: VRRS: VRRP-TEST-1 group deleted notification
```

```
*Sep 9 16:09:54.360: VRRS: Low priority clients 4
*Sep 9 16:09:54.360: VRRS: Client 2 remove from VRRP-TEST-1. vrrs handle F7000001
*Sep 9 16:09:54.360: VRRS: client remove, no more clients and no server for group VRRP-TEST-1.
 Remov
*Sep 9 16:09:54.860: VRRS: Client 22 is not registered
*Sep 9 16:09:54.860: VRRS: Client 22 unregister failed
*Sep 9 16:09:54.860: VRRS: Client VRRS TEST CLIENT registered, id 22
```

Related Commands

Command	Description
debug vrrp vrrs	Enables VRRP debugging statements for VRRS interactions.
debug vrrs accounting	Enables debug messages for VRRS accounting.
debug vrrs plugin	Enables VRRS plug-in debug messages.

debug vrrs log

debug vrrs log [**detail**]

Syntax Description

detail	(Optional) Enables detailed debugging information associated with VRRS logs.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
15.3(1)S	This command was introduced.
Cisco IOS XE Release 3.8S	This command was introduced.

Usage Guidelines

You must configure the VRRS control groups using the **vrrs** command on interfaces that require a redundant virtual gateway.

Examples

Related Commands

Command	Description
debug vrrs all	Enables debugging information associated with all elements of Virtual Router Redundancy Service (VRRS).
debug vrrs database	Enables debugging information associated with VRRS databases.
debug vrrs pathway	Enables debugging information associated with VRRS pathways.

debug vrrs pathway

debug vrrs pathway {**all** [**detail**]| **process** [**detail**]| **address** [*ipv4-address* [**Ethernet** *number*]| **Ethernet** *number* [*ipv4-address*| **IPv4**| **IPv6**| *ipv6-address*]| **IPv4** [**Ethernet** *number*]| **IPv6** [**Ethernet** *number*]| *ipv6-address* [**Ethernet** *number*]] [**detail**]| {**database**| **error**| **event**| **mac-address**| **protocol**| **state**} [**Ethernet** *number* [**IPv4** [**verbose**]| **IPv6** [**verbose**]]| **IPv4** [**Ethernet** *number* [**verbose**]| **verbose** [**Ethernet** *number*]]| **IPv6** [**Ethernet** *number* [**verbose**]| **verbose** [**Ethernet** *number*]]] [**detail**]}

Syntax Description

all	Enables debugging information associated with all VRRS pathways.
detail	(Optional) Enables detailed debugging information associated with all VRRS pathways.
process	Enables debugging information associated with VRRS pathway processes.
address	Enables debugging information associated with VRRS pathway addresses.
ipv4-address	Enables debugging information associated with IPv4 addresses on VRRS pathways.
Ethernet *number*	(Optional) Enables debugging information associated with VRRS pathways for ethernet interfaces.
IPv4	(Optional) Enables debugging information associated with VRRS pathways for VRRP groups adhering to IPv4 protocol.
IPv6	(Optional) Enables debugging information associated with VRRS pathways for VRRP groups adhering to IPv6 protocol.
ipv6-address	Enables debugging information associated with IPv6 addresses on VRRS pathways.
database	Enables debugging information associated with VRRS pathways for databases.
error	Enables debugging information associated with VRRS pathway errors.
event	Enables debugging information associated with VRRS pathway events.

mac-address	Enables debugging information associated with MAC addresses on VRRS pathways.
protocol	Enables debugging information associated with VRRS pathway protocols.
state	Enables debugging information associated with VRRS pathways for interface states.
verbose	Enables debugging information associated with VRRS pathways for non-protocol events.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
15.3(1)S	This command was introduced.
Cisco IOS XE Release 3.8S	This command was introduced.

Usage Guidelines

You must configure VRRS pathways by defining the First Hop Redundancy Protocol (FHRP) groups and configuring the interfaces that require redundant virtual gateway.

Examples

The following example shows how to enable debugging information associated with all elements of VRRS using the **debug vrrs platform** command:

```
Device# debug vrrs platform

vrrs pathway event debugging is on
vrrs pathway database debugging is on
vrrs pathway error debugging is on
vrrs pathway mac debugging is on
vrrs pathway address resolution protocol debugging is on
vrrs pathway process debugging is on
vrrs pathway state debugging is on
vrrs pathway address debugging is on
```

Related Commands

Command	Description
debug vrrs all	Enables debugging information associated with all elements of Virtual Router Redundancy Service (VRRS).
debug vrrs database	Enables debugging information associated with VRRS databases.

Command	Description
debug vrrs log	Enables debugging information associated with VRRS logs.

debug vrrs plugin

To enable Virtual Router Redundancy Service (VRRS) plug-in debug messages, use the **debug vrrs plugin**command in privileged EXEC mode. To disable VRRS plug-in debug messages, use the **no** form of this command.

debug vrrs plugin {**all**| **arp-packet**| **client**| **database**| **if-state**| **mac**| **process**| **sublock**| **test**}

no debug vrrs plugin {**all**| **arp-packet**| **client**| **database**| **if-state**| **mac**| **process**| **sublock**| **test**}

Syntax Description

all	Enables all VRRS debugs.
arp-packet	Enables debugging for VRRS mac-address gratuitous ARP messages.
client	Enables debugging for VRRS plug-in client interactions with VRRS.
database	Enables debugging for VRRS plug-in database management.
if-state	Enables VRRS events associated specifically with the VRRS interface-state plug-in.
mac	Enables VRRS events associated specifically with the VRRS mac-address plug-in.
process	Enables debugging for the VRRS plug-in events process.
sublock	Enables debugging for VRRS interface subblock management.
test	Enables VRRS plug-in test code monitoring.

Command Default

VRRS plug-in debug messages are not enabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS XE Release 2.6	This command was introduced.

Examples

The following is sample output when a VRRS borrowed MAC address is added to the MAC address filter of an interface enables VRRS plug-in debug messages:

```
Router)# debug vrrs plugin all
Feb 17 19:15:38.052: VRRS-P(mac): GigEth0/0/0.1 Add 0000.12ad.0001 to MAC filter, using
(afilter_add)
Feb 17 19:15:38.053: VRRS-P(mac): Active count increase to (2) for MAC : 0000.12ad.0001
```

The table below describes the significant fields shown in the display.

Table 80: debug vrrs plugin Field Descriptions

Field	Description
VRRS-P	Specifies this debug is related to VRRS plug-ins.
(mac)	Specifies this debug is related to the VRRS mac-address plug-in. Alternately (if-state) may displayed to indicate the debug is related to the VRRS interface-state plugiplug-inn.

Related Commands

Command	Description
debug vrrp vrrs	Enables VRRP debugging statements for VRRS interactions.
debug vrrs accounting	Enables debug messages for VRRS accounting.
debug vrrs infra	Enables VRRS infrastructure debug messages.

debug vsi api

Note Effective with Cisco IOS Release 12.4(20)T, the **debug vsi api**command is not available in Cisco IOS software.

To display information on events associated with the external ATM application programming interface (API) interface to the Virtual Switch Interface (VSI) master, use the **debug vsi api** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug vsi api

no debug vsi api

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.0(5)T	This command was introduced.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.
12.4(20)T	This command was removed.

Usage Guidelines

Use the **debug vsi api** command to monitor the communication between the VSI master and the XmplsATM component regarding interface changes and cross-connect requests.

Examples

The following is sample output from the **debug vsi api** command:

```
Router# debug vsi api
VSI_M: vsi_exatm_conn_req: 0x000C0200/1/35 -> 0x000C0100/1/50
       desired state up, status OK
VSI_M: vsi_exatm_conn_resp: 0x000C0200/1/33 -> 0x000C0100/1/49
       curr state up, status OK
```

The table below describes the significant fields shown in the display.

Table 81: debug vsi api Command Field Descriptions

Field	Description
vsi_exatm_conn_req	The type of connection request (connect or disconnect) that was submitted to the VSI master.
0x000C0200	The logical interface identifier of the primary endpoint, in hexadecimal form.
1/35	The virtual path identifier (VPI) and virtual channel identifier (VCI) of the primary endpoint.
->	The type of traffic flow. A right arrow (->) indicates unidirectional traffic flow (from the primary endpoint to the secondary endpoint). A bidirectional arrow (<->) indicates bidirectional traffic flow.
0x000C0100	Logical interface identifier of the secondary endpoint.
1/50	VPI and VCI of the secondary endpoint.
desired state	The status of a connect request. Up indicates a connect request; Down indicates a disconnect request.
status (in vsi_exatm_conn_req output)	The status of a request. One of following status indications appears: OK INVALID_ARGS NONEXIST_INTF TIMEOUT NO_RESOURCES FAIL OK means only that the request is successfully queued for transmission to the switch; it does not indicate completion of the request.

debug vsi errors

Note Effective with Cisco IOS Release 12.4(20)T, the **debug vsi errors** command is not available in Cisco IOS software.

To display information about errors encountered by the Virtual Switch Interface (VSI) master, use the **debug vsi errors** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug vsi errors [**interface** *interface* [**slave** *number*]]

no debug vsi errors [**interface** *interface* [**slave** *number*]]

Syntax Description

interface *interface*	(Optional) Specifies the interface number.
slave *number*	(Optional) Specifies the slave number (beginning with 0).

Command Default

No default behavior or values.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.0(5)T	This command was introduced.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.
12.4(20)T	This command was removed.

Usage Guidelines

Use the **debug vsi errors** command to display information about errors encountered by the VSI master when parsing received messages, as well as information about unexpected conditions encountered by the VSI master.

If the interface parameter is specified, output is restricted to errors associated with the indicated VSI control interface. If the slave number is specified, output is further restricted to errors associated with the session with the indicated slave.

Note Slave numbers are the same as the session numbers discussed under the **show controllers vsi session** command.

Multiple commands that specify slave numbers allow multiple slaves to be debugged immediately. For example, the following commands display errors associated with sessions 0 and 1 on control interface atm2/0, but for no other sessions.

```
Router#
debug vsi errors interface atm2/0 slave 0
Router#
debug vsi errors interface atm2/0 slave 1
```

Some errors are not associated with any particular control interface or session. Messages associated with these errors are printed, regardless of the **interface** or **slave**options currently in effect.

Examples

The following is sample output from the **debug vsi errors** command:

```
Router# debug vsi errors
VSI Master: parse error (unexpected param-group contents) in GEN ERROR RSP rcvd on ATM2/0:0/51
 (slave 0)
              errored section is at offset 16, for 2 bytes:
 01.01.00.a0 00.00.00.00 00.12.00.38 00.10.00.34
*00.01*00.69 00.2c.00.00 01.01.00.80 00.00.00.08
 00.00.00.00 00.00.00.00 00.00.00.00 0f.a2.00.0a
 00.01.00.00 00.00.00.00 00.00.00.00 00.00.00.00
 00.00.00.00
```

The table below describes the significant fields shown in the display.

Table 82: debug vsi errors Field Descriptions

Field	Description
`parse error`	An error was encountered during the parsing of a message received by the VSI master.
`unexpected param-group contents`	The type of parsing error. In this case, a parameter group within the message contained invalid data.
`GEN ERROR RSP`	The function code in the header of the error message.
`ATM2/0`	The control interface on which the error message was received.
`0/51`	The virtual path identifier (VPI) or virtual channel identifier (VCI) of the virtual circuit (VC) (on the control interface) on which the error message is received.
`slave`	Number of the session on which the error message is received.
`offset <n>`	The number of bytes between the start of the VSI header and the start of that portion of the message in error.
`<n> bytes`	Length of the error section.
`00.01.00.a0 [...]`	The entire error message, as a series of hexadecimal bytes. Note that the error section is between asterisks (*).

debug vsi events

Note Effective with Cisco IOS Release 12.4(20)T, the**debug vsi events** command is not available in Cisco IOS software.

To display information about events that affect entire sessions, as well as events that affect only individual connections, use the **debug vsi events** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug vsi events [**interface interface** [**slave number**]]

no debug vsi events [**interface interface** [**slave number**]]

Syntax Description

interface *interface* >	(Optional) The interface number.
slave > *number*	(Optional) The slave number (beginning with zero).

Command Default

No default behavior or values.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(5)T	This command was introduced.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.

Usage Guidelines

Use the **debug vsi events** command to display information about events associated with the per-session state machines of the Virtual Switch Interface (VSI) master, as well as the per-connection state machines. If you specify an interface, the output is restricted to events associated with the indicated VSI control interface. If you specify the slave number, output is further restricted to events associated with the session with the indicated slave.

Note Slave numbers are the same as the session numbers discussed under the **show controllers vsi session** command.

Multiple commands that specify slave numbers allow multiple slaves to be debugged at once. For example, the following commands restrict output to events associated with sessions 0 and 1 on control interface atm2/0, but for no other sessions. Output associated with all per-connection events are displayed, regardless of the **interface** or **slave** options currently in effect.

```
Router#
debug vsi events interface atm2/0 slave 0
Router#
debug vsi events interface atm2/0 slave 1
```

Examples

The following is sample output from the **debug vsi events** command:

```
Router# debug vsi events
VSI Master: conn 0xC0200/1/37->0xC0100/1/51:
            CONNECTING -> UP
VSI Master(session 0 on ATM2/0):
    event CONN_CMT_RSP, state ESTABLISHED -> ESTABLISHED
VSI Master(session 0 on ATM2/0):
    event KEEPALIVE_TIMEOUT, state ESTABLISHED -> ESTABLISHED
VSI Master(session 0 on ATM2/0):
    event SW_GET_CNFG_RSP, state ESTABLISHED -> ESTABLISHED
debug vsi packets
```

The table below describes the significant fields shown in the display.

Table 83: debug vsi events Field Descriptions

Field	Description
`conn`	The event applies to a particular connection.
`0xC0200`	Logical interface identifier of the primary endpoint, in hexadecimal form.
`1/37`	The virtual path identifier (VPI) or virtual channel identifier (VCI) of the primary endpoint.
`->`	The type of traffic flow. A right arrow (->) indicates unidirectional traffic flow (from the primary endpoint to the secondary endpoint). A bidirectional arrow (<->) indicates bidirectional traffic flow.
`0xC0100`	Logical interface identifier of the secondary endpoint.
`1/51`	VPI or VCI of the secondary endpoint.
`<state1> -> <state2>`	<state1> is a mnemonic for the state of the connection before the event occurred. <state2> represents the state of the connection after the event occurred.
`session`	The number of the session with which the event is associated.
`ATM2/0`	The control interface associated with the session.
`event`	The event that has occurred. This includes mnemonics for the function codes of received messages (for example, CONN_CMT_RSP), as well as mnemonics for other events (for example, KEEPALIVE_TIMEOUT).
`state <state1> -> <state2>`	Mnemonics for the session states associated with the transition triggered by the event. <state1> is a mnemonic for the state of the session before the event occurred; <state2> is a mnemonic for the state of the session after the event occurred.

debug vsi packets

Note Effective with Cisco IOS Release 12.4(20)T, the **debug vsi packets**command is not available in Cisco IOS software.

To display a one-line summary of each Virtual Switch Interface (VSI) message sent and received by the label switch controller (LSC), use the **debug vsi packets** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug vsi packets [**interface interface** [**slave number**]]

no debug vsi packets [**interface interface** [**slave number**]]

Syntax Description

interface *interface*	(Optional) The interface number.
slave > *number*	(Optional) The slave number (beginning with zero).

Command Default

No default behavior or values.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.0(5)T	This command was introduced.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.
12.4(20)T	This command was removed.

Usage Guidelines

If you specify an interface, output is restricted to messages sent and received on the indicated VSI control interface. If you specify a slave number, output is further restricted to messages sent and received on the session with the indicated slave.

Note Slave numbers are the same as the session numbers discussed under the **show controllers vsi session** command.

Multiple commands that specify slave numbers allow multiple slaves to be debugged at once. For example, the following commands restrict output to messages received on atm2/0 for sessions 0 and 1, but for no other sessions.

```
Router# debug vsi packets interface atm2/0 slave 0
Router# debug vsi packets interface atm2/0 slave 1
```

Examples

The following is sample output from the **debug vsi packets** command:

```
Router# debug vsi packets
VSI master(session 0 on ATM2/0): sent msg SW GET CNFG CMD on 0/51
VSI master(session 0 on ATM2/0): rcvd msg SW GET CNFG RSP on 0/51
VSI master(session 0 on ATM2/0): sent msg SW GET CNFG CMD on 0/51
VSI master(session 0 on ATM2/0): rcvd msg SW GET CNFG RSP on 0/51
```

The table below describes the significant fields shown in the display.

Table 84: debug vsi packets Field Descriptions

Field	Description
`session`	Session number identifying a particular VSI slave. Numbers begin with zero. See the **show controllers vsi session** command.
`ATM2/0`	Identifier for the control interface on which the message is sent or received.
`sent`	The message is sent by the VSI master.
`rcvd`	The message is received by the VSI master.
`msg`	The function code from the message header.
`0/51`	The virtual path identifier (VPI) or virtual channel identifier (VCI) of the virtual circuit (VC) (on the control interface) on which the message is sent or received.

debug vsi param-groups

Note Effective with Cisco IOS Release 12.4(20)T, the **debug vsi param-groups**command is not available in Cisco IOS software.

To display the first 128 bytes of each Virtual Switch Interface (VSI) message sent and received by the Multiprotocol Label Switching (MPLS) label switch controller (LSC) (in hexadecimal form), use the **debug vsi param-groups** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug vsi param-groups [**interface interface** [**slave number**]]

no debug vsi param-groups [**interface interface** [**slave number**]]

Syntax Description

interface *interface*	(Optional) The interface number.
slave > *number*	(Optional) The slave number (beginning with zero).

Command Default

No default behavior or values.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.0(5)T	This command was introduced.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.
12.4(20)T	This command was removed.

Usage Guidelines

This command is most commonly used with the **debug vsi packets** command to monitor incoming and outgoing VSI messages.

Note **param-groups** stands for parameter groups. A parameter group is a component of a VSI message.

If you specify an interface, output is restricted to messages sent and received on the indicated VSI control interface.

If you specify a slave, output is further restricted to messages sent and received on the session with the indicated slave.

Note Slave numbers are the same as the session numbers discussed under the **show controllers vsi session** command.

Multiple commands that specify slave numbers allow multiple slaves to be debugged at once. For example, the following commands restrict output for messages received on atm2/0 for sessions 0 and 1, but for no other sessions:

```
Router# debug vsi param-groups interface atm2/0 slave 0

Router# debug vsi param-groups interface atm2/0 slave 1
```

Examples

The following is sample output from the **debug vsi param-groups** command:

```
Router# debug vsi param-groups
Outgoing VSI msg of 12 bytes (not including encap):
 01.02.00.80 00.00.95.c2 00.00.00.00
Incoming VSI msg of 72 bytes (not including encap):
 01.02.00.81 00.00.95.c2 00.0f.00.3c 00.10.00.08
 00.01.00.00 00.00.00.00 01.00.00.08 00.00.00.09
 00.00.00.09 01.10.00.20 01.01.01.00 0c.08.80.00
 00.01.0f.a0 00.13.00.15 00.0c.01.00 00.00.00.00
 42.50.58.2d 56.53.49.31
Outgoing VSI msg of 12 bytes (not including encap):
 01.02.00.80 00.00.95.c3 00.00.00.00
Incoming VSI msg of 72 bytes (not including encap):
 01.02.00.81 00.00.95.c3 00.0f.00.3c 00.10.00.08
 00.01.00.00 00.00.00.00 01.00.00.08 00.00.00.09
 00.00.00.09 01.10.00.20 01.01.01.00 0c.08.80.00
 00.01.0f.a0 00.13.00.15 00.0c.01.00 00.00.00.00
 42.50.58.2d 56.53.49.31
```

The table below describes the significant fields shown in the display.

Table 85: debug vsi param-groups Field Descriptions

Field	Description
Outgoing	The message is sent by the VSI master.
Incoming	The message is received by the VSI master.
bytes	Number of bytes in the message, starting at the VSI header, and excluding the link layer encapsulation.
01.02...	The first 128 bytes of the message, in hexadecimal form.

debug vtemplate

To display cloning information for a virtual access interface from the time it is cloned from a virtual template to the time the virtual access interface comes down when the call ends, use the **debug vtemplate** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug vtemplate

no debug vtemplate

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Examples

The following is sample output from the **debug vtemplate** command when a virtual access interface comes up. The virtual access interface is cloned from virtual template 1.

```
Router# debug vtemplate
VTEMPLATE Reuse vaccess8, New Recycle queue size:50

VTEMPLATE set default vaccess8 with no ip address

Virtual-Access8 VTEMPLATE hardware address 0000.0c09.ddfd
VTEMPLATE vaccess8 has a new cloneblk vtemplate, now it has vtemplate
VTEMPLATE undo default settings vaccess8

VTEMPLATE ************* CLONE VACCESS8 *****************

VTEMPLATE Clone from vtemplate1 to vaccess8
interface Virtual-Access8
no ip address
encap ppp
ip unnumbered Ethernet0
no ip mroute-cache
fair-queue 64 256 0
no cdp enable
ppp authentication chap
end

%LINK-3-UPDOWN: Interface Virtual-Access8, changed state to up
%LINEPROTO-5-UPDOWN: Line protocol on Interface Virtual-Access8, changed state to up
```

The following is sample output from the **debug vtemplate** command when a virtual access interface goes down. The virtual interface is uncloned and returns to the recycle queue.

```
Router# debug vtemplate
%LINK-3-UPDOWN: Interface Virtual-Access8, changed state to down
VTEMPLATE Free vaccess8

%LINEPROTO-5-UPDOWN: Line protocol on Interface Virtual-Access8, changed state to down
VTEMPLATE clean up dirty vaccess queue, size:1

VTEMPLATE Found a dirty vaccess8 clone with vtemplate
VTEMPLATE ************ UNCLONE VACCESS8 **************
VTEMPLATE Unclone to-be-freed vaccess8 command#7
interface Virtual-Access8
default ppp authentication chap
default cdp enable
default fair-queue 64 256 0
default ip mroute-cache
default ip unnumbered Ethernet0
default encap ppp
default ip address
end

VTEMPLATE set default vaccess8 with no ip address

VTEMPLATE remove cloneblk vtemplate from vaccess8 with vtemplate

VTEMPLATE Add vaccess8 to recycle queue, size=51
```

The table below describes the significant fields shown in the display.

Table 86: debug vtemplate Field Descriptions

Field	Description
VTEMPLATE Reuse vaccess8, New Recycle queue size:50 VTEMPLATE set default vaccess8 with no ip address	Virtual access interface 8 is reused; the current queue size is 50.
Virtual-Access8 VTEMPLATE hardware address 0000.0c09.ddfd	MAC address of virtual interface 8.
VTEMPLATE vaccess8 has a new cloneblk vtemplate, now it has vtemplate	Recording that virtual access interface 8 is cloned from the virtual interface template.
VTEMPLATE undo default settings vaccess8	Removing the default settings.
VTEMPLATE ************* CLONE VACCESS8 ********** *******	Banner: Cloning is in progress on virtual access interface 8.
VTEMPLATE Clone from vtemplate1 to vaccess8 interface Virtual-Access8 no ip address encap ppp ip unnumbered Ethernet0 no ip mroute-cache fair-queue 64 256 0 no cdp enable ppp authentication chap end	Specific configuration commands in virtual interface template 1 that are being applied to the virtual access interface 8.
%LINK-3-UPDOWN: Interface Virtual-Access8, changed state to up	Link status: The link is up.
%LINEPROTO-5-UPDOWN: Line protocol on Interface Virtual-Access8, changed state to up	Line protocol status: The line protocol is up.
%LINK-3-UPDOWN: Interface Virtual-Access8, changed state to down	Link status: The link is down.
VTEMPLATE Free vaccess8	Freeing virtual access interface 8.
%LINEPROTO-5-UPDOWN: Line protocol on Interface Virtual-Access8, changed state to down	Line protocol status: The line protocol is down.

Field	Description
VTEMPLATE clean up dirty vaccess queue, size:1 VTEMPLATE Found a dirty vaccess8 clone with vtemplate VTEMPLATE ************ UNCLONE VACCESS8 **************	Access queue cleanup is proceeding and the template is being uncloned.
VTEMPLATE Unclone to-be-freed vaccess8 command#7 interface Virtual-Access8 default ppp authentication chap default cdp enable default fair-queue 64 256 0 default ip mroute-cache default ip unnumbered Ethernet0 default encap ppp default ip address end	Specific configuration commands to be removed from the virtual access interface 8.
VTEMPLATE set default vaccess8 with no ip address	Default is set again.
VTEMPLATE remove cloneblk vtemplate from vaccess8 with vtemplate	Removing the record of cloning from a virtual interface template.
VTEMPLATE Add vaccess8 to recycle queue, size=51	Virtual access interface is added to the recycle queue.

debug vtemplate subinterface

To display debug messages relating to virtual access subinterfaces, use the debug vtemplate subinterface command in privileged EXEC mode. To disable debugging output, use the no form of this command.

debug vtemplate subinterface

no debug vtemplate subinterface

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(8)B	This command was introduced.
12.2(13)T	This command was integrated into Cisco IOS Release 12.2(13)T.
12.2(15)B	This command was integrated into Cisco IOS Release 12.2(15)B.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(31)SB	This command was integrated into Cisco IOS Release 12.2(31)SB.

Usage Guidelines

The debug messages are displayed if you configure virtual templates with commands that are incompatible with virtual access subinterfaces.

Examples

The following example shows how to display virtual access subinterface debug messages:

```
Router# debug vtemplate subinterface
Virtual Template subinterface debugging is on
Router#
Router#
Sep 19 15:09:41.989:VT[Vt11]:Config prevents subinterface creation
carrier-delay 45
ip rtp priority 2000 2010 500
```

The table below describes the significant fields shown in the display.

Table 87: debug vtemplate subinterface Field Descriptions

Field	Description
VT	Indicates that this is a debug virtual template subinterface message.
[Vt11]:	Indicates that this message concerns virtual template 11.
Config prevents subinterface creation	Indicates that this virtual template cannot support the creation of virtual access subinterfaces.
carrier-delay 45 ip rtp priority 2000 2010 500	These are the commands that make the virtual template incompatible with subinterfaces.

Related Commands

Command	Description
test virtual-template subinterface	Tests a virtual template to determine if it can support virtual access subinterfaces.
virtual-template subinterface	Enables the creation of virtual access subinterfaces.

debug vtsp

Note Effective with release 12.3(8)T, the **debug vtsp**command is replaced by the **debug voip dsm** and **debug voip vtsp**commands. See the **debug voip dsm** and **debug voip vtsp**commands for more information.

To display the state of the gateway and the call events, use the **debug vtsp**command in privileged EXEC mode. To display the machine state during voice telephony service provider (VTSP) event processing, use the **no** form of the command.

debug vtsp {**all**| **dsp**| **error**| **event**| **session**| **stats**| **tone**| **rtp**}

no debug vtsp {**all**| **dsp**| **error**| **event**| **session**| **stats**| **tone**| **rtp**}

Syntax Description

all	All VTSP debugging except stats, tone, and event is enabled.
dsp	Digital signal processor (DSP) message trace is enabled.
error	VTSP error debugging is enabled.
event	State machine debugging is enabled.
session	Session debugging is enabled.
stats	Statistics debugging is enabled.
tone	Tone debugging is enabled.
rtp	Real-Time Transport Protocol (RTP) debugging is enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(3)T	This command was introduced on the Cisco AS5300 universal access servers.
12.0(7)XK	This command was introduced on the Cisco 2600 series router, Cisco 3600 series router, and MC3810 multiservice access concentrators.
12.1(2)T	This command was integrated into Cisco IOS Release 12.1(2)T.

Release	Modification
12.2(11)T	The enhancement of debug capabilities, which affects this command by adding a single call identification header, for Cisco voice gateways was added to the following Cisco routers: Cisco 2600 series, Cisco 3620, Cisco 3640, and Cisco 3660 series; on the following universal gateways: Cisco AS5350, Cisco AS5400, and Cisco AS5850; on the following access servers: Cisco AS5300, and Cisco AS5800; and, on the Cisco MC3810 multiservice access concentrators.
12.3(8)T	This command was replaced by the **debug voip vtsp** command.

Usage Guidelines

The **debug vtsp** command with the **event** keyword must be turned on before the **voice call debug** command can be used.

Examples

The following is sample output for a Cisco AS5300 and Cisco 3640 when the **debug vtsp** all command is entered:

Examples

```
Router# debug vtsp all
!
Voice telephony call control all debugging is on
!
00:10:53: %SYS-5-CONFIG_I: Configured from console by console
00:10:54: %SYS-5-CONFIG_I: Configured from console by console
!
00:11:09:
//-1/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx/VTSP:():-1:-1:-1/vtsp_do_regxrule_translate:
00:11:09:
//-1/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx/VTSP:():-1:-1:-1/vtsp_do_regxrule_translate:
00:11:09: //-1/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx/VTSP:():-1:-1:-1/vtsp_tsp_call_setup_ind:

00:11:09: //-1/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx/VTSP:():-1:-1:-1/vtsp_tsp_fill_setup_ind:
```

Examples

```
3640-orig# debug vtsp all
!
Voice telephony call control all debugging is on
!
3640-orig# show debug
Voice Telephony session debugging is on
Voice Telephony dsp debugging is on
Voice Telephony error debugging is on
!
20:58:16:
//-1/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx/VTSP:():-1:-1:-1/vtsp_tsp_apply_voiceport_xrule:

20:58:16: vtsp_tsp_apply_voiceport_xrule: vtsp_sdb 0x63797720; called_number 0x6294E0F0
called_oct3 128
20:58:16:
//-1/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx/VTSP:():-1:-1:-1/vtsp_tsp_apply_voiceport_xrule:

20:58:16: vtsp_tsp_apply_voiceport_xrule: No called number translation rule configured
20:58:16:
//-1/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx/VTSP:():-1:-1:-1/vtsp_do_regxrule_translate: .
20:58:16:
//-1/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx/VTSP:():-1:-1:-1/vtsp_do_regxrule_translate:
calling_number(original)=
calling_number(xlated)=8880000 called_number(original)= called_number(xlated)=8881111
redirectNumber(original)= redirectNumber(xlated)=
20:58:16: //-1/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx/VTSP:():-1:-1:-1/vtsp_tsp_call_setup_ind:
 (sdb=0x63797720, tdm_info=0x0,
tsp_info=0x63825254, calling_number=8880000 calling_oct3 = 0x0, called_number=8881111
called_oct3 = 0x80, oct3a=0
3640-orig#x80): peer_tag=70
20:58:16: //-1/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx/VTSP:():-1:-1:-1/vtsp_tsp_fill_setup_ind:
 ev.clg.clir is 0
 ev.clg.clid_transparent is 0
 ev.clg.null_orig_clg is 0
 ev.clg.calling_translated is false
//-1/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx/VTSP:(3/0:23):-1:0:0/vtsp_do_call_setup_ind: Call
 ID=101123, guid=63EB9AC8
```

The table below describes the significant fields shown in the display.

Table 88: debug vtsp all Field Descriptions

Field	Description
VTSP:():-1:-1:-1	Identifies the VTSP module, port name, channel number, DSP slot, and DSP channel number.
vtsp_tsp_apply_voiceport_xrule:	Identifies a function name.
called_number	Identifies a called number.
called	Identifies the date the call was made.
peer_tag	Identifies the dial peer number.
guid	Identifies the GUID (hexadecimal address).

Related Commands

Command	Description
debug voip ccapi	Debugs the call control API.
voice call debug	Debugs a voice call by displaying a full GUID or header.

debug vtsp all

To show debugging information for all **debug vtsp** commands, use the **debug vtsp all** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug vtsp all

no debug vtsp all

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(3)T	This command was introduced on the Cisco AS5300.
12.0(7)XK	This command was first supported on the Cisco 2600, Cisco 3600 and Cisco MC3810 series.
12.1(2)T	This command was integrated into Cisco IOS Release 12.1(2)T.
12.2(11)T	The new debug header was added to the following Cisco routers: Cisco 2600 series, Cisco 3620, and Cisco 3640. and Cisco 3660; on the following universal gateways: Cisco AS5350, Cisco AS5400, and Cisco AS5850; on the following universal access servers: Cisco AS5300, and Cisco AS5800; and, on the Cisco MC3810 multiservice access concentrators.

Usage Guidelines

The **debug vtsp all** command enables the following **debug vtsp** commands: **debug vtsp session**, **debug vtsp error**, and **debug vtsp dsp**. For more information or sample output, see the individual commands.

Execution of the **no debug vtsp all** command will turn off all VTSP-level debugging. You should turn off all debugging and then enter the **debug** commands you are interested in one by one. This process helps avoid confusion about which ports you are actually debugging.

Caution Using this command can severely impact network performance and prevent any faxes from succeeding.

Examples

The following example shows the **debug vtsp all** command on a Cisco 3640 modular access router:

```
Router# debug vtsp all
Voice telephony call control all debugging is on
```

At this point, the VTSP is not aware of anything. The format of this message is //callid/GUID/VTSP:(voice-port):T1-channel_number:DSP_number:DSP_channel_number:

- CallEntry ID is -1.
- GUID is xxxxxxxxxx.
- The voice port is blank.
- Channel ID is -1.
- DSP ID is -1.
- DSP channel ID is -1.

```
*Mar  1 08:23:10.869: //-1/xxxxxxxxxxxx/VTSP:():-1:-1:-1/vtsp_do_regxrule_translate:
```

The original and the translated calling number are the same (55555) and the original and the translated called number are the same (888545). These numbers are often the same because if a translation rule is applied, it will be on the dial peers or the ports, both of which comes later than these VTSP messages in the Cisco IOS code execution.

```
*Mar  1 08:23:10.869: //-1/xxxxxxxxxxxx/VTSP:():-1:-1:-1/vtsp_do_regxrule_translate:
calling_number(original)= calling_number(xlated)=55555 called_number(original)=
called_number(xlated)=888545 redirectNumber(original)= redirectNumber(xlated)=
```

The VTSP got a call setup indicator from the TSP layer with called number 888545 and calling number 55555. There is no awareness of the CallEntry ID (-1) or the GUID (xxxxxxxxxxxx).

```
*Mar  1 08:23:10.873: //-1/xxxxxxxxxxxx/VTSP:():-1:-1:-1/vtsp_tsp_call_setup_ind:
(sdb=0x634C90EC, tdm_info=0x0, tsp_info=0x63083950, calling_number=55555 calling_oct3 =
0x80, called_number=888545 called_oct3 = 0x80, oct3a=0x0): peer_tag=10002
*Mar  1 08:23:10.873: //-1/xxxxxxxxxxxx/VTSP:():-1:-1:-1/vtsp_tsp_fill_setup_ind
: ev.clg.clir is 0
 ev.clg.clid_transparent is 0
 ev.clg.null_orig_clg is 0
 ev.clg.calling_translated is false
*Mar  1 08:23:10.873: //-1/xxxxxxxxxxxx/VTSP:():-1:-1:-1/vtsp_do_call_setup_ind: .
*Mar  1 08:23:10.873: //-1/xxxxxxxxxxxx/VTSP:():-1:-1:-1/vtsp_allocate_cdb: ,cdb 0x635FC480
*Mar  1 08:23:10.873: //-1/xxxxxxxxxxxx/VTSP:():-1:-1:-1/vtsp_do_call_setup_ind:
*Mar  1 08:23:10.873:  source route label
```

At this point, the VTSP is not aware of anything. The format of this message is //callid/GUID/VTSP:(voice-port):T1-channel_number:DSP_number:DSP_channel_number:

- CallEntry ID is -1.
- GUID is D2F6429A8A8A.
- The voice port is 1/0:23 where 23 indicates D channel.
- The T1 channel is still unknown at this point (-1).
- The digital signal processor (DSP) is 0.

- The DSP channel is 4.

```
*Mar  1 08:23:10.873: //-1/D2F6429A8A8A/VTSP:(1/0:23):-1:0:4/vtsp_do_call_setup_
ind: Call ID=101002, guid=635FCB08
```

The VTSP learns about the B channel (changed from -1 to 22), and the CallEntry ID is still unknown (-1).

```
*Mar  1 08:23:10.873: //-1/D2F6429A8A8A/VTSP:(1/0:23):22:0:4/vtsp_do_call_setup_ind: type=0,
 under_spec=1615186336, name=, id0=23, id1=0, id2=0, calling=55555,called=888545
subscriber=RegularLinevtsp_do_call_setup_ind: redirect DN =  reason = -1
*Mar  1 08:23:10.877: //-1/xxxxxxxxxxxx/VTSP:():-1:-1:-1/vtsp_do_normal_call_setup_ind: .
```

The VTSP learns the CallEntry ID. The format of this message is //callid/GUID/VTSP:(voice-port):T1-channel_number:DSP_number:DSP_channel_number:

- CallEntry ID is 899 (changed from -1 to 899)
- GUID is D2F6429A8A8A
- The voice port is 1/0:23 where 23 indicates D channel
- The T1 channel is 22
- The DSP is 12
- The DSP channel is 4

```
*Mar  1 08:23:10.877: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_insert_cdb:,cdb
0x635FC480, CallID=899
*Mar  1 08:23:10.877: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_open_voice_and_set_params:
 .
```

In the following outputs, VTSP sets some of the voice parameters for this call:

- Modem capability
- Playout delay
- Dial-peer tag 10003
- Digit timeouts

```
*Mar  1 08:23:10.877: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_modem_proto_from_cdb:
cap_modem_proto 0
*Mar  1 08:23:10.881: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/set_playout_cdb:playout
default
*Mar  1 08:23:10.881:
//899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_dsp_echo_canceller_control: echo_cancel: 1
*Mar  1 08:23:10.885: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_save_dialpeer_tag: tag
 = 10003
*Mar  1 08:23:10.885: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_report_digit_control:
vtsp_report_digit_control: enable=0:
*Mar  1 08:23:10.885: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_report_digit_control:
digit reporting disabled
*Mar  1 08:23:10.885: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_get_digit_timeouts: :
vtsp_get_digit_timeouts
```

VTSP sends out a call-proceeding message to the POTS leg.

```
*Mar  1 08:23:10.885:
//899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_process_event:vtsp:[1/0:23:899,
S_SETUP_INDICATED, E_CC_PROCEEDING]
*Mar  1 08:23:10.885: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/act_proceeding: .
*Mar  1 08:23:10.941: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_get_dialpeer_tag: tag
= 10003
*Mar  1 08:23:10.949: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_get_dialpeer_tag: tag
= 10003
```

VTSP sends out an alerting to the POTS leg; the phone is ringing at this time.

```
*Mar  1 08:23:10.949: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_process_event:
vtsp:[1/0:23:899, S_PROCEEDING, E_CC_ALERT]
*Mar  1 08:23:10.949: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/act_alert: .
*Mar  1 08:23:10.949: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_timer_stop:3019095
*Mar  1 08:23:18.769: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_get_dialpeer_tag: tag
= 10003
```

The phone gets answered here, a bridge is now set up between the two call legs.

```
*Mar  1 08:23:18.769: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_process_event:
vtsp:[1/0:23:899, S_ALERTING, E_CC_BRIDGE]
*Mar  1 08:23:18.769: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/act_bridge: .
```

The call is now connected.

```
*Mar  1 08:23:18.769: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_process_event:
vtsp:[1/0:23:899, S_ALERTING, E_CC_CONNECT]
*Mar  1 08:23:18.769: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/act_alert_connect: .
*Mar  1 08:23:18.773: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_ring_noan_timer_stop:
3019877
```

The VTSP received a capabilities indication event from the CCAPI. The VTSP needs to be aware of this because it handles the DSPs.

```
*Mar  1 08:23:18.773: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_process_event:
vtsp:[1/0:23:899, S_CONNECT, E_CC_CAPS_IND]
*Mar  1 08:23:18.773: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/act_caps_ind: .
*Mar  1 08:23:18.773: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/act_caps_ind: RTP

PT:NTE[101],NTEtx[101],NSE[100],FaxInd[96],FaxAck[97],CiscoDTMF[121],FaxRelay[122],CASsig[123],ClearChan[125],PCMu[0],PCMa[8]Codec[4],TxDynamicPayload[0],
 RxDynamicPayload[0]
*Mar  1 08:23:18.773: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/act_caps_ind: dtmf relay:
mode=32, codec=1
*Mar  1 08:23:18.773: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/act_caps_ind: passthrough:
cap_modem_proto 0, cap_modem_codec 0, cap_modem_redundancy 0, payload100, modem_relay 0,
gw-xid=0
*Mar  1 08:23:18.773: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/act_caps_ind: Encap 1, Vad
2, Codec 0x4, CodecBytes 20,
             FaxRate 2, FaxBytes 20, FaxNsf 0xAD0051
             SignalType 2
             DtmfRelay 32, Modem 0, SeqNumStart 0x1343
*Mar  1 08:23:18.773: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/act_caps_ind:
*Mar  1 08:23:18.777:  FORKING Parameters are forking mask: 0, simple_forking_codec_mask:
0, complex_forking_codec_mask 0
*Mar  1 08:23:18.777: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/act_caps_ind: [ mode:0,init:60,
 min:40, max:200]
```

The VTSP received events regarding capabilities acknowledged from the call control API (CCAPI).

```
*Mar  1 08:23:18.777: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_process_event:
vtsp:[1/0:23:899, S_CONNECT, E_CC_CAPS_ACK]
*Mar  1 08:23:18.777: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/act_caps_ack: .
*Mar  1 08:23:18.777: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/act_caps_ack: passthrough:
cap_modem_proto 0, cap_modem_codec 0, cap_modem_redundancy 0, payload100, modem_relay 0,
gw-xid=0
*Mar  1 08:23:18.777: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/act_caps_ack: Named Telephone
 Event payload: rcv 101, tx 101
*Mar  1 08:23:18.777: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/act_switch_codec:
*Mar  1 08:23:18.777:  DTMF Relay in act_switch_codec is 32
*Mar  1 08:23:18.777: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/set_dsp_encap_config:
*Mar  1 08:23:18.777:  set_dsp_encap_config: logical ssrc 40
*Mar  1 08:23:18.777: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_modem_proto_from_cdb:
cap_modem_proto 0
*Mar  1 08:23:18.777: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/act_switch_codec: codec =
16
*Mar  1 08:23:18.781: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_timer: 3019878
*Mar  1 08:23:18.781: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_process_event:
vtsp:[1/0:23:899, SP_PENDING_CODEC_SWITCH, E_DSPRM_PEND_SUCCESS]
*Mar  1 08:23:18.781: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/act_pend_codec_success: .
```

```
*Mar  1 08:23:18.781: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_timer_stop:3019878
*Mar  1 08:23:18.781: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_open_voice_and_set_params:
 .
*Mar  1 08:23:18.781: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/set_dsp_encap_config:
*Mar  1 08:23:18.781:  set_dsp_encap_config: logical ssrc 40
*Mar  1 08:23:18.781: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_modem_proto_from_cdb:
cap_modem_proto 0
*Mar  1 08:23:18.781: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/set_playout_cdb:playout
default
*Mar  1 08:23:18.781:
//899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_dsp_echo_canceller_control: echo_cancel: 1
*Mar  1 08:23:18.781: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_add_fork:
*Mar  1 08:23:18.785: vtsp_add_fork
*Mar  1 08:23:18.785: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_update_fork_info:
*Mar  1 08:23:18.785: vtsp_update_fork_info: add_fork=0
*Mar  1 08:23:18.785: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_get_xmit_info_node:
*Mar  1 08:23:18.785: vtsp_get_xmit_info_node
*Mar  1 08:23:18.785: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_update_fork_info:
*Mar  1 08:23:18.785:  vtsp_update_fork_info xmit func is 60FC43F0, context is
635BC51Cpeer_call_id: 900, stream_count: 1, update_flag 0
Router#
*Mar  1 08:23:18.785: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_update_fork_info:
*Mar  1 08:23:18.785:  The stream bit-mask is 1
*Mar  1 08:23:18.785: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_update_fork_info:
*Mar  1 08:23:18.785:  The stream type is 0
*Mar  1 08:23:18.785: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_update_fork_info:
*Mar  1 08:23:18.785:  The logical ssrc is 64 for stream 0
*Mar  1 08:23:18.785: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_update_stream_count:
*Mar  1 08:23:18.785:  g711_voice_count=0 g711_avt_count = 0
 g711_voice_avt_count = 0 complex_voice_count = 1
 complex_avt_count = 0 complex_voice_avt_count = 0
```

A digit begin event was detected while in the connect state. Digit 1 is dialed outbound on the POTS legs.

```
*Mar  1 08:23:26.745: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_call_digit_begin:
vtsp_call_digit_begin: digit=1, digit_begin_flags=0x0, rtp_timestamp=0, rtp_expiration=0
*Mar  1 08:23:26.745: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_process_event:
vtsp:[1/0:23:899, S_CONNECT, E_CC_DIGIT_BEGIN]
*Mar  1 08:23:26.745: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/act_digit_begin:act_digit_begin
*Mar  1 08:23:27.045: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_call_digit_end:
vtsp_call_digit_end: digit=1, duration=300
```

A digit end event was detected while in the connect state. The total duration of the digit was 300 ms.

```
*Mar  1 08:23:27.045: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_process_event:
vtsp:[1/0:23:899, S_CONNECT, E_CC_DIGIT_END,]
*Mar  1 08:23:27.045: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/act_digit_end: act_digit_end
```

The call is hung up at this point, VTSP receives a bridge drop event from the CCAPI.

```
*Mar  1 08:23:39.393: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_process_event:
vtsp:[1/0:23:899, S_CONNECT, E_CC_BRIDGE_DROP]
*Mar  1 08:23:39.393: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_remove_stream_node:
*Mar  1 08:23:39.393: vtsp_remove_stream_node
*Mar  1 08:23:39.393: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_get_xmit_info_node:
*Mar  1 08:23:39.393: vtsp_get_xmit_info_node
*Mar  1 08:23:39.393: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_remove_stream_node:
*Mar  1 08:23:39.393:  Stream count is 1 in function vtsp_remove_stream_node
*Mar  1 08:23:39.393: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/act_bdrop: .
*Mar  1 08:23:39.393: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_is_record_active:
*Mar  1 08:23:39.393: vtsp_is_record_active: false
```

VTSP gets a disconnect event from the CCAPI.

```
*Mar  1 08:23:39.397: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_process_event:
vtsp:[1/0:23:899, S_CONNECT, E_CC_DISCONNECT]
*Mar  1 08:23:39.397: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/act_disconnect: .
```

Following the disconnect event from the CCAPI, the timers are stopped.

```
*Mar  1 08:23:39.397: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_ring_noan_timer_stop:
3021940
```

```
*Mar  1 08:23:39.397:
//899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_pcm_tone_detect_timer_stop: 3021940
*Mar  1 08:23:39.397: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_pcm_switchover_timer_stop:
 3021940
*Mar  1 08:23:39.397: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_cm_detect_timer_stop:
3021940
*Mar  1 08:23:39.397:
//899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_modem_relay_mode_timer_stop: 3021940
*Mar  1 08:23:39.397:
//899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_modem_relay_stats_timer_stop: 3021940
*Mar  1 08:23:39.397: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_timer_stop:3021940
*Mar  1 08:23:39.397: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/act_disconnect: cdb 0x635FC480,
 cause 0x10
*Mar  1 08:23:39.401: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_timer: 3021940
```

Statistics are collected for the DSP.

```
*Mar  1 08:23:39.405: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_process_event:
vtsp:[1/0:23:899, S_WAIT_STATS, E_DSP_GET_ERROR]
*Mar  1 08:23:39.405: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/act_get_error: .
*Mar  1 08:23:39.405: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_print_error_stats:
rx_dropped=0 tx_dropped=0
*Mar  1 08:23:39.405: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_print_error_stats:
rx_control=55 tx_control=18 tx_control_dropped=0  dsp_mode_channel_1=0
dsp_mode_channel_2=0c[0]=0c[1]=2c[2]=6c[3]=87c[4]=83c[5]=84c[6]=106c[7]=78c[8]=0c[9]=32639c[10]=32639c[11]=32639c[12]=32639c[13]=32639c[14]=32639c[15]=32639
*Mar  1 08:23:39.409: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_timer_stop:3021941
*Mar  1 08:23:39.409: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_timer: 3021941
*Mar  1 08:23:39.409: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_process_event:
vtsp:[1/0:23:899, S_WAIT_STATS, E_DSP_GET_LEVELS]
*Mar  1 08:23:39.409: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/act_get_levels: .
*Mar  1 08:23:39.413: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/act_stats_complete: .
*Mar  1 08:23:39.413: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_timer_stop:3021941
*Mar  1 08:23:39.413: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_ring_noan_timer_stop:
3021941
*Mar  1 08:23:39.417: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_timer: 3021942
```

The VTSP received a disconnect confirmation from the TSP layer.

```
*Mar  1 08:23:39.417: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_process_event:
vtsp:[1/0:23:899, S_WAIT_RELEASE, E_TSP_DISCONNECT_CONF]
*Mar  1 08:23:39.417: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/act_wrelease_release: .
*Mar  1 08:23:39.417: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_play_busy_timer_stop:
*Mar  1 08:23:39.417: vtsp_play_busy_timer_stop: 3021942
*Mar  1 08:23:39.417: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_timer_stop:3021942
*Mar  1 08:23:39.417: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_do_call_history: .
*Mar  1 08:23:39.417: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_do_call_history:
*Mar  1 08:23:39.417: vtsp_do_call_history :  src carrier id
*Mar  1 08:23:39.417: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_do_call_history:
*Mar  1 08:23:39.421: vtsp_do_call_history : tgt carrier id
*Mar  1 08:23:39.421: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_do_call_history: CoderRate
 16
```

DSP resource manager updates the state.

```
*Mar  1 08:23:39.421: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_process_event:
vtsp:[1/0:23:899, S_CLOSE_DSPRM, E_DSPRM_CLOSE_COMPLETE]
*Mar  1 08:23:39.421: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/act_terminate: .
*Mar  1 08:23:39.421: //899/D2F6429A8A8A/VTSP:(1/0:23):22:12:4/vtsp_free_cdb: ,cdb 0x635FC4803
```

Related Commands

Command	Description
debug vtsp port	Limits VTSP debug output to a specific voice port.
show debug	Displays which debug commands are enabled.
voice call debug	Allows configuration of the voice call debug output.

debug vtsp dsp

To show messages from the digital signal processor (DSP) to the universal access server or router, use the **debug vtsp dsp command**in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug vtsp dsp

no debug vtsp dsp

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(3)T	This command was introduced on the Cisco AS5300 series access servers.
12.0(7)XK	This command was implemented on the Cisco 2600 series, Cisco 3600 series, and Cisco MC3810 multiservice access concentrators.
12.1(2)T	This command was integrated into Cisco IOS Release 12.1(2)T.
12.2(11)T	The new debug header was added to the following Cisco routers: Cisco 2600 series, Cisco 3620, and Cisco 3640. and Cisco 3660; on the following universal gateways: Cisco AS5350, Cisco AS5400, and Cisco AS5850; on the following universal access servers: Cisco AS5300, and Cisco AS5800; and, on the Cisco MC3810 multiservice access concentrators.

Usage Guidelines

On Cisco AS5300 Series Access Servers

The **debug vtsp dsp** command shows messages from the DSP on the voice feature card (VFC) to the router; this command can be useful if you suspect that the VFC is not functional. It is a simple way to check if the VFC is responding to off-hook indications.

On Cisco 2600, 3600, MC3810 Series

The **debug vtsp dsp** command shows messages from the DSP to the router.

Note We recommend that you log output from the **debug vtsp dsp**command to a buffer rather than sending the output to the console; otherwise, the size of the output could severely impact the performance of the gateway.

Examples

The following example shows the VTSP DSP usage on a Cisco 3640 modular access router:

```
Router# debug vtsp dsp
Voice telephony call control dsp debugging is on
Router#
*Mar  1 01:05:18.539: //12/A76D98838014/VTSP:(1/0:23):22:14:2/vtsp_dsp_echo_canceller_control:
 echo_cancel: 1
```

The table below describes the significant fields shown in the display.

Table 89: debug vtsp dsp Field Descriptions

Field	Descriptions
//12	CallEntry ID.
/A76D98838014	GUID.
1/0:23	Controller 1/0, D channel.
:22	B-channel number. This can also be found using the **show voice call summary** command.
:14	DSP number. This can also be found using the **show voice dsp** command.
:2	Channel number on the DSP. This can also be found using the **show voice dsp** command.
echo_cancel: 1	Echo cancel is on.

Related Commands

Command	Description
debug vpm all	Enables all VPM debugging.
debug vtsp port	Limits VTSP debug output to a specific voice port.
show debug	Displays which debug commands are enabled.
voice call debug	Allows configuration of the voice call debug output.

debug vtsp error

To display processing errors in the voice telephony service provider (VTSP), use the **debug vtsp error** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug vtsp error

no debug vtsp error

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(7)XK	This command was first supported on the Cisco 2600, Cisco 3600 and Cisco MC3810 series.
12.1(2)T	This command was integrated into Cisco IOS Release 12.1(2)T.
12.2(11)T	The new debug header was added to the following Cisco routers: Cisco 2600 series, Cisco 3620, and Cisco 3640. and Cisco 3660; on the following universal gateways: Cisco AS5350, Cisco AS5400, and Cisco AS5850; on the following universal access servers: Cisco AS5300, and Cisco AS5800; and, on the Cisco MC3810 multiservice access concentrators.

Usage Guidelines

The **debug vtsp error** command can be used to check for mismatches in interface capabilities.

Note We recommend that you log output from the **debug vtsp error**command to a buffer rather than sending the output to the console; otherwise, the size of the output could severely impact the performance of the gateway.

Related Commands

Command	Description
debug vpm all	Enables all VPM debugging.
debug vtsp port	Limits VTSP debug output to a specific voice port.

Command	Description
show debug	Displays which debug commands are enabled.
voice call debug	Allows configuration of the voice call debug output.

debug vtsp event

To display the state of the gateway and the call events, use the **debug vtsp event** command in privileged EXEC mode. To display the machine state during voice telephony service provider (VTSP) event processing, use the **no** form of this command.

debug vtsp event

no debug vtsp event

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(3)T	This command was introduced on the Cisco AS5300 universal access servers.
12.0(7)XK	This command was first supported on the Cisco 2600, Cisco 3600 and Cisco MC3810 series.
12.1(2)T	This command was integrated into Cisco IOS Release 12.1(2)T.
12.2(11)T	The new debug header was added to the following Cisco routers: Cisco 2600 series, Cisco 3620, and Cisco 3640. and Cisco 3660; on the following universal gateways: Cisco AS5350, Cisco AS5400, and Cisco AS5850; on the following universal access servers: Cisco AS5300, and Cisco AS5800; and, on the Cisco MC3810 multiservice access concentrators.

Usage Guidelines

The **debug vtsp event** command can be used to enable state machine debugging.

Note We recommend that you log output from the **debug vtsp event**command to a buffer rather than sending the output to the console; otherwise, the size of the output could severely impact the performance of the gateway.

Examples

The following shows sample output from the **debug vtsp event** command:

```
Router# debug vtsp event
Voice Telephony event debugging is on
```

The following events are seen when the call is set up.

```
*Mar  1 22:20:39.138: //72/D14258FE806E/VTSP:(1/0:23):22:14:2/vtsp_process_event:
[state:S_SETUP_INDICATED,  event: E_CC_PROCEEDING]
```

When the phone starts ringing, the ALERT event appears.

```
*Mar  1 22:20:39.202: //72/D14258FE806E/VTSP:(1/0:23):22:14:2/vtsp_process_event:
[state:S_PROCEEDING,  event: E_CC_ALERT]
Router#
```

As soon as the call is answered, the bridge comes up and the CONNECT event appears.

```
*Mar  1 22:20:47.798: //72/D14258FE806E/VTSP:(1/0:23):22:14:2/vtsp_process_event:
[state:S_ALERTING,  event: E_CC_BRIDGE]
*Mar  1 22:20:47.802: //72/D14258FE806E/VTSP:(1/0:23):22:14:2/vtsp_process_event:
[state:S_ALERTING,  event: E_CC_CONNECT]
```

The capabilities are exchanged as soon as the connection occurs.

```
*Mar  1 22:20:47.802: //72/D14258FE806E/VTSP:(1/0:23):22:14:2/vtsp_process_event:
[state:S_CONNECT,  event: E_CC_CAPS_IND]
*Mar  1 22:20:47.802: //72/D14258FE806E/VTSP:(1/0:23):22:14:2/vtsp_process_event:
[state:S_CONNECT,  event: E_CC_CAPS_ACK]
*Mar  1 22:20:47.802: //72/D14258FE806E/VTSP:(1/0:23):22:14:2/vtsp_process_event:
[state:SP_PENDING_CODEC_SWITCH,  event: E_DSPRM_PEND_SUCCESS]
```

The following debug outputs are regularly seen as the call progresses. The outputs indicate that collection of Tx/Rx/Delay/Error statistics is occurring.

```
*Mar  1 22:20:49.470: //72/D14258FE806E/VTSP:(1/0:23):22:14:2/vtsp_process_event:
[state:S_CONNECT,  event: E_CC_REQ_PACK_STAT]
*Mar  1 22:20:49.482: //72/D14258FE806E/VTSP:(1/0:23):22:14:2/vtsp_process_event:
[state:S_CONNECT,  event: E_DSP_GET_TX]
*Mar  1 22:20:49.482: //72/D14258FE806E/VTSP:(1/0:23):22:14:2/vtsp_process_event:
[state:S_CONNECT,  event: E_DSP_GET_RX]
*Mar  1 22:20:49.486: //72/D14258FE806E/VTSP:(1/0:23):22:14:2/vtsp_process_event:
[state:S_CONNECT,  event: E_DSP_GET_VP_DELAY]
*Mar  1 22:20:49.486: //72/D14258FE806E/VTSP:(1/0:23):22:14:2/vtsp_process_event:
[state:S_CONNECT,  event: E_DSP_GET_VP_ERROR]
*Mar  1 22:20:51.638: //72/D14258FE806E/VTSP:(1/0:23):22:14:2/vtsp_process_event:
[state:S_CONNECT,  event: E_CC_REQ_PACK_STAT]
*Mar  1 22:20:51.638: //72/D14258FE806E/VTSP:(1/0:23):22:14:2/vtsp_process_event:
[state:S_CONNECT,  event: E_DSP_GET_TX]
*Mar  1 22:20:51.638: //72/D14258FE806E/VTSP:(1/0:23):22:14:2/vtsp_process_event:
[state:S_CONNECT,  event: E_DSP_GET_RX]
*Mar  1 22:20:51.642: //72/D14258FE806E/VTSP:(1/0:23):22:14:2/vtsp_process_event:
[state:S_CONNECT,  event: E_DSP_GET_VP_DELAY]
*Mar  1 22:20:51.642: //72/D14258FE806E/VTSP:(1/0:23):22:14:2/vtsp_process_event:
[state:S_CONNECT,  event: E_DSP_GET_VP_ERROR]
Router#
```

When digits are passed during the conversation, the digit begin and digit end events are seen.

```
*Mar  1 22:21:01.542: //72/D14258FE806E/VTSP:(1/0:23):22:14:2/vtsp_process_event:
[state:S_CONNECT,  event: E_CC_DIGIT_BEGIN]
*Mar  1 22:21:01.842: //72/D14258FE806E/VTSP:(1/0:23):22:14:2/vtsp_process_event:
[state:S_CONNECT,  event: E_CC_DIGIT_END,]
*Mar  1 22:21:01.962: //72/D14258FE806E/VTSP:(1/0:23):22:14:2/vtsp_process_event:
[state:S_CONNECT,  event: E_CC_DIGIT_BEGIN]
*Mar  1 22:21:02.262: //72/D14258FE806E/VTSP:(1/0:23):22:14:2/vtsp_process_event:
[state:S_CONNECT,  event: E_CC_DIGIT_END,]
Router#
```

Once the call is hung up from one side, the bridge_drop and the disconnect events appear.

```
*Mar  1 22:21:10.834: //72/D14258FE806E/VTSP:(1/0:23):22:14:2/vtsp_process_event:
[state:S_CONNECT,  event: E_TSP_DISCONNECT_IND]
*Mar  1 22:21:10.838: //72/D14258FE806E/VTSP:(1/0:23):22:14:2/vtsp_process_event:
[state:S_CONNECT,  event: E_CC_BRIDGE_DROP]
```

```
*Mar  1 22:21:10.838: //72/D14258FE806E/VTSP:(1/0:23):22:14:2/vtsp_process_event:
[state:S_CONNECT,  event: E_CC_DISCONNECT]
```

Following the disconnect event, the signaling state becomes S_WAIT_STATS, during which the DSP stats are collected.

```
*Mar  1 22:21:10.842: //72/D14258FE806E/VTSP:(1/0:23):22:14:2/vtsp_process_event:
[state:S_WAIT_STATS,  event: E_DSP_GET_ERROR]
*Mar  1 22:21:10.846: //72/D14258FE806E/VTSP:(1/0:23):22:14:2/vtsp_process_event:
[state:S_WAIT_STATS,  event: E_DSP_GET_LEVELS]
*Mar  1 22:21:10.854: //72/D14258FE806E/VTSP:(1/0:23):22:14:2/vtsp_process_event:
[state:S_WAIT_STATS,  event: E_DSP_GET_TX]
```

The conference is torn down and the DSP is released.

```
*Mar  1 22:21:10.854: //72/D14258FE806E/VTSP:(1/0:23):22:14:2/vtsp_process_event:
[state:S_WAIT_RELEASE,  event: E_TSP_DISCONNECT_CONF]
*Mar  1 22:21:10.858: //72/D14258FE806E/VTSP:(1/0:23):22:14:2/vtsp_process_event:
[state:S_CLOSE_DSPRM,  event: E_DSPRM_CLOSE_COMPLETE]
```

Related Commands

Command	Description
debug vpm all	Enables all VPM debugging.
debug vtsp error	Displays processing errors in the VTSP.
debug vtsp port	Limits VTSP debug output to a specific voice port.
show debug	Displays which debug commands are enabled.
voice call debug	Allows configuration of the voice call debug output.

debug vtsp port

To observe the behavior of the voice telephony service provider (VTSP) state machine on a specific voice port, use the **debug vtsp port** command in privileged EXEC mode . To disable debugging output, use the **no** form of this command.

For Cisco 2600 and Cisco 3600 Series with Analog Voice Ports

debug vtsp port *slot*/*subunit*/*port*

no debug vtsp port *slot*/*subunit*/*port*

For Cisco 2600 and Cisco 3600 Series with Digital Voice Ports (With T1 Packet Voice Trunk Network Modules)

debug vtsp port *slot*/*port*:*ds0-group*

no debug vtsp port *slot*/*port*:*ds0-group*

For Cisco MC3810 Series with Analog Voice Ports

debug vtsp port *slot*/*port*

no debug vtsp port *slot*/*port*

For Cisco MC3810 Series with Digital Voice Ports

debug vtsp port *slot*/*port*

no debug vtsp port *slot*/*ds0-group*

slot/subunit/port	• *slot* specifies a router slot in which a voice network module (NM) is installed. Valid entries are router slot numbers for the specific platform. • *subunit* specifies a voice interface card (VIC) where the voice port is located. Valid entries are 0 and 1. (The VIC fits into the voice network module.) • *port* specifies an analog voice port number. Valid entries are 0 and 1.

Syntax Description

slot/port:ds0-group	Debugs the digital voice port you specify with the *slot/port:ds0-group* designation. • *slot* specifies a router slot in which the packet voice trunk network module (NM) is installed. Valid entries are router slot numbers for the specific platform. • *port* specifies a T1 or E1 physical port in the voice WAN interface card (VWIC). Valid entries are 0 and 1. (One VWIC fits in an NM.) • *ds0-group* specifies a T1 or E1 logical port number. Valid entries are 0 to 23 for T1 and 0 to 30 for E1.

Syntax Description

slot/port	Debugs the analog voice port you specify with the *slot/port*designation. • *slot* is the physical slot in which the analog voice module (AVM) is installed. The *slot* is always 1 for analog voice ports in the Cisco MC3810 series. • *port* specifies an analog voice port number. Valid entries are 1 to 6.

Syntax Description

slot:ds0-group	Debugs the digital voice port you specify with the *slot:ds0-group* designation. • *slot* specifies the module (and controller). Valid entries are 0 for the MFT (controller 0) and 1 for the DVM (controller 1). • *ds0-group* specifies a T1 or E1 logical voice port number. Valid entries are 0 to 23 for T1 and 0 to 30 for E1.

Command Default

Debug VTSP commands are not limited to a specific port.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(3)XG	This command was introduced on Cisco 2600 and Cisco 3600 series routers.
12.0(3)T	This command was introduced on the Cisco AS5300 series access servers.
12.0(7)XK	This command was first supported on the Cisco MC3810 series.
12.1(2)T	This command was integrated into Cisco IOS Release 12.1(2)T.
12.2(11)T	The new debug header was added to the following Cisco routers: Cisco 2600 series, Cisco 3620, and Cisco 3640. and Cisco 3660; on the following universal gateways: Cisco AS5350, Cisco AS5400, and Cisco AS5850; on the following universal access servers: Cisco AS5300, and Cisco AS5800; and, on the Cisco MC3810 multiservice access concentrators.

Usage Guidelines

Use the **debug vtsp port** command to limit the debug output to a specific voice port. The debug output can be quite voluminous for a single channel. The entire VTSP debug output from a platform with 12 voice ports might create problems. Use this **debug** command with any or all of the other debug modes.

Execution of **no debug vtsp all** will turn off all VTSP-level debugging. It is usually a good idea to turn off all debugging and then enter the **debug** commands you are interested in one by one. This will help to avoid confusion about which ports you are actually debugging.

Note We recommend that you log output from the **debug vtsp port**command to a buffer rather than sending the output to the console; otherwise, the size of the output could severely impact the performance of the gateway.

Related Commands

Command	Description
debug vpm all	Enables all VPM debugging.
show debug	Displays which debug commands are enabled.
voice call debug	Allows configuration of the voice call debug output.

debug vtsp rtp

To show the voice telephony service provider (VTSP) Real-Time Protocol (RTP) packet debugging, use the **debug vtsp rtp**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug vtsp rtp {**both| from-dsp| to-dsp**} **payload** *payload-type codec*

no debug vtsp rtp

Syntax Description

both	Displays packets that are both sent and received from the digital signal processor (DSP).
from-dsp	Displays packets received from the DSP.
to-dsp	Displays packets sent to the DSP.
payload	(Optional) Specifies a specific type of payload.
payload-type	(Optional) Valid payload types are as follows: • **all** --All packets are displayed. No codec is specified. • **equal-to** --Packets in payloads equal to the specified codec are displayed. • **greater-than** --Packets in payloads greater than the specified codec are displayed. • **less-than** --Packets in payloads less than the specified codec are displayed. • **other-than** --Packets in payloads other than the specified codec are displayed. • **other-than-fax-and** --Packets in payloads other than fax relay and the specified codec are displayed. • **other-than-silence-and** --Packets in payloads other than silence and the specified codec are displayed.

codec	(Optional) If a codec needs to be specified for the payload type, valid codecs are as follows: • **0** to **123**--Custom value of the payload. • **g711alaw** --G.711 alaw 64000 bps. • **g711ulaw** --G.711 ulaw 64000 bps. • **g723.1** --G.723.1. • **g726** --G.726. • **g728** --G.728. • **g729a** --G.729a.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(3)T	This command was introduced on the Cisco AS5300 series access servers.
12.0(7)XK	This command was first supported on the Cisco 2600, Cisco 3600, and MC3810 series devices.
12.1(2)T	This command was integrated into Cisco IOS Release 12.1(2)T.
12.2(11)T	The new debug header was added to the following Cisco routers: Cisco 2600 series, Cisco 3620, and Cisco 3640. and Cisco 3660; on the following universal gateways: Cisco AS5350, Cisco AS5400, and Cisco AS5850; on the following universal access servers: Cisco AS5300, and Cisco AS5800; and, on the Cisco MC3810 multiservice access concentrators.

Usage Guidelines

We recommend that you log output from the **debug vtsp rtp** command to a buffer rather than sending the output to the console; otherwise, the size of the output could severely impact the performance of the gateway.

Examples

The following example shows the VTSP RTP debugging:

```
Router# debug vtsp rtp both pay all
Voice telephony RTP Packet debugging enabled for payloads of all types of packets from and
 to DSP
```

The following line shows the payload from the DSP (telephony leg) to the IP leg:

```
*Mar  1 01:10:05.687: //20/4DD959B48020/VTSP:(1/0:23):22:14:2/vtsp_print_rtp_header: s=DSP
 d=VoIP payload 0x12 ssrc 0x40 sequence 0x19E3 timestamp 0xCCDCE092
```

The following line shows the payload from the IP leg to the DSP (telephony leg):

```
*Mar  1 01:10:05.699: //20/4DD959B48020/VTSP:(1/0:23):22:14:2/vtsp_print_rtp_header: s=VoIP
 d=DSP payload 0x12 ssrc 0xAF0534E3 sequence 0x92A timestamp 0x6BE50
```

Related Commands

Command	Description
debug vtsp dsp	Shows messages from the DSP.
voice call debug	Allows configuration of the voice call debug output.

debug vtsp send-nse

To trigger the voice telephony service provider (VTSP) software module to send a triple redundant network services engine (NSE), use the **debug vtsp send-nse** command in privileged EXEC mode. To disable this action, use the **no** form of this command.

debug vtsp send-nse

no debug vtsp send-nse

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(11)T	This command was introduced.

Usage Guidelines

We recommend that you log output from the **debug vtsp send-nse**command to a buffer rather than sending the output to the console; otherwise, the size of the output could severely impact the performance of the gateway.

Related Commands

Command	Description
debug rtpspi all	Debugs all RTP SPI errors, sessions, and in/out functions.
debug rtpspi errors	Debugs RTP SPI errors.
debug rtpspi inout	Debugs RTP SPI in/out functions.
debug rtpspi send-nse	Triggers the RTP SPI to send a triple redundant NSE.
debug sgcp errors	Debugs SGCP errors.
debug sgcp events	Debugs SGCP events.
debug sgcp packet	Debugs SGCP packets.
voice call debug	Allows configuration of the voice call debug output.

debug vtsp session

To trace how the router interacts with the digital signal processor (DSP) based on the signaling indications from the signaling stack and requests from the application, use the **debug vtsp session** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug vtsp session

no debug vtsp session

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(3)T	This command was introduced on the Cisco AS5300 universal access servers.
12.0(7)XK	This command was implemented on the Cisco 2600, Cisco 3600 and Cisco MC3810 series.
12.2(11)T	The new debug header was added to the following Cisco routers: Cisco 2600 series, Cisco 3620, and Cisco 3640. and Cisco 3660; on the following universal gateways: Cisco AS5350, Cisco AS5400, and Cisco AS5850; on the following universal access servers: Cisco AS5300, and Cisco AS5800; and, on the Cisco MC3810 multiservice access concentrators.

Usage Guidelines

The **debug vtsp session** command traces how the router interacts with the DSP based on the signaling indications from the signaling stack and requests from the application. This debug command displays information about how each network indication and application request is handled, signaling indications, and DSP control messages.

This debug level shows the internal workings of the voice telephony call state machine.

Note We recommend that you log output from the **debug vtsp send-nse** command to a buffer rather than sending the output to the console; otherwise, the size of the output could severely impact the performance of the gateway.

Examples

The following shows sample output from the **debug vtsp session**command:

```
Router# debug vtsp session
Voice telephony call control session debugging is on
```

At this point, the VTSP is not aware of anything. The format of this message is //callid/GUID/VTSP:(voice-port):T1-channel_number:DSP_number:DSP_channel_number:

- CallEntry ID is -1.
- GUID is xxxxxxxxxx.
- The voice port is blank.
- Channel ID is -1.
- DSP ID is -1.
- DSP channel ID is -1.

```
*Mar  2 01:20:43.225: //-1/xxxxxxxxxxxx/VTSP:():-1:-1:-1/vtsp_do_regxrule_translate: .
```

The original and the translated calling number are the same (55555) and the original and the translated called number are the same (888545). These numbers are often the same because if a translation rule is applied, it will be on the dial peers or the ports both of which comes later than these VTSP messages in the Cisco IOS code execution.

```
*Mar  2 01:20:43.225: //-1/xxxxxxxxxxxx/VTSP:():-1:-1:-1/vtsp_do_regxrule_translate:
calling_number(original)= calling_number(xlated)=55555 called_number(original)=
called_number(xlated)=888545 redirectNumber(original)= redirectNumber(xlated)=
```

The VTSP got a call setup indicator from the TSP layer with called number 888545 and calling number 55555. There is no awareness of the CallEntry ID (-1) or the GUID (xxxxxxxxxxxx).

```
*Mar  2 01:20:43.225: //-1/xxxxxxxxxxxx/VTSP:():-1:-1:-1/vtsp_tsp_call_setup_ind:
(sdb=0x637AA6C0, tdm_info=0x0, tsp_info=0x630B6050, calling_number=55555 calling_oct3 =
0x80, called_number=888545 called_oct3 = 0x80, oct3a=0x0): peer_tag=10002
*Mar  2 01:20:43.225: //-1/xxxxxxxxxxxx/VTSP:():-1:-1:-1/vtsp_tsp_fill_setup_ind: ev.clg.clir
 is 0
 ev.clg.clid_transparent is 0
 ev.clg.null_orig_clg is 0
 ev.clg.calling_translated is false
*Mar  2 01:20:43.229: //-1/xxxxxxxxxxxx/VTSP:():-1:-1:-1/vtsp_do_call_setup_ind: .
*Mar  2 01:20:43.229: //-1/xxxxxxxxxxxx/VTSP:():-1:-1:-1/vtsp_allocate_cdb: ,cdb 0x637B2A68
*Mar  2 01:20:43.229: //-1/xxxxxxxxxxxx/VTSP:():-1:-1:-1/vtsp_do_call_setup_ind:
*Mar  2 01:20:43.229:  source route label
```

At this point, the VTSP is not aware of the anything. The format of this message is //callid/GUID/VTSP:(voice-port):T1-channel_number:DSP_number:DSP_channel_number:

- CallEntry ID is -1.
- GUID is F90073EB8080.
- The voice port is 1/0:23 where 23 indicates D channel.
- The T1 channel is still unknown at this point (-1).
- The DSP is 0.

- The DSP channel is 2.

```
*Mar  2 01:20:43.229: //-1/F90073EB8080/VTSP:(1/0:23):-1:0:2/vtsp_do_call_setup_ind: Call
ID=98432, guid=637B43F4
```

The VTSP learns that the B channel used changed from -1 to 22.

```
*Mar  2 01:20:43.229: //-1/F90073EB8080/VTSP:(1/0:23):22:0:2/vtsp_do_call_setup_ind: type=0,
 under_spec=1615186336, name=, id0=23, id1=0, id2=0, calling=55555,called=888545
subscriber=RegularLinevtsp_do_call_setup_ind: redirect DN =  reason = -1
*Mar  2 01:20:43.229: //-1/xxxxxxxxxxxx/VTSP:():-1:-1:-1/vtsp_do_normal_call_setup_ind: .
```

The VTSP learns the CallEntry ID. The format of this message is //callid/GUID/VTSP:(voice-port):T1-channel_number:DSP_number:DSP_channel_number:

- CallEntry ID is 84 (changed from -1 to 84).
- GUID is F90073EB8080.
- The voice port is 1/0:23 where 23 indicates D channel.
- The T1 channel is 22.
- The DSP is 14.
- The DSP channel is 2.

```
*Mar  2 01:20:43.233: //84/F90073EB8080/VTSP:(1/0:23):22:14:2/vtsp_insert_cdb: ,cdb
0x637B2A68, CallID=84
*Mar  2 01:20:43.233: //84/F90073EB8080/VTSP:(1/0:23):22:14:2/vtsp_open_voice_and_set_params:
 .
```

In the following outputs VTSP sets some of the voice parameters for this call:

- Modem capability
- Playout-delay
- Dial-peer tag = 10003
- Digit-timeouts

```
*Mar  2 01:20:43.233: //84/F90073EB8080/VTSP:(1/0:23):22:14:2/vtsp_modem_proto_from_cdb:
cap_modem_proto 0
*Mar  2 01:20:43.233: //84/F90073EB8080/VTSP:(1/0:23):22:14:2/set_playout_cdb: playout
default
*Mar  2 01:20:43.237: //84/F90073EB8080/VTSP:(1/0:23):22:14:2/vtsp_save_dialpeer_tag: tag
= 10003
*Mar  2 01:20:43.237: //84/F90073EB8080/VTSP:(1/0:23):22:14:2/vtsp_report_digit_control:
vtsp_report_digit_control: enable=0:
*Mar  2 01:20:43.237: //84/F90073EB8080/VTSP:(1/0:23):22:14:2/vtsp_report_digit_control:
digit reporting disabled
*Mar  2 01:20:43.237: //84/F90073EB8080/VTSP:(1/0:23):22:14:2/vtsp_get_digit_timeouts: :
vtsp_get_digit_timeouts
```

The VTSP sends out a call-proceeding message to the POTS leg.

```
*Mar  2 01:20:43.241: //84/F90073EB8080/VTSP:(1/0:23):22:14:2/vtsp_process_event:
vtsp:[1/0:23:84, S_SETUP_INDICATED, E_CC_PROCEEDING]
*Mar  2 01:20:43.241: //84/F90073EB8080/VTSP:(1/0:23):22:14:2/act_proceeding: .
Router#
*Mar  2 01:20:43.297: //84/F90073EB8080/VTSP:(1/0:23):22:14:2/vtsp_get_dialpeer_tag: tag =
 10003
*Mar  2 01:20:43.301: //84/F90073EB8080/VTSP:(1/0:23):22:14:2/vtsp_get_dialpeer_tag: tag =
 10003
```

VTSP sends out an alerting to the POTS leg; the phone is ringing now.

```
*Mar  2 01:20:43.301: //84/F90073EB8080/VTSP:(1/0:23):22:14:2/vtsp_process_event:
vtsp:[1/0:23:84, S_PROCEEDING, E_CC_ALERT]
*Mar  2 01:20:43.301: //84/F90073EB8080/VTSP:(1/0:23):22:14:2/act_alert: .
*Mar  2 01:20:43.301: //84/F90073EB8080/VTSP:(1/0:23):22:14:2/vtsp_timer_stop: 9124331
Router#
*Mar  2 01:20:52.289: //84/F90073EB8080/VTSP:(1/0:23):22:14:2/vtsp_get_dialpeer_tag: tag =
 10003
```

The phone gets answered here, and a bridge is now set up between the two call legs.

```
*Mar  2 01:20:52.289: //84/F90073EB8080/VTSP:(1/0:23):22:14:2/vtsp_process_event:
vtsp:[1/0:23:84, S_ALERTING, E_CC_BRIDGE]
*Mar  2 01:20:52.289: //84/F90073EB8080/VTSP:(1/0:23):22:14:2/act_bridge: .
```

The call is now connected.

```
*Mar  2 01:20:52.289: //84/F90073EB8080/VTSP:(1/0:23):22:14:2/vtsp_process_event:
vtsp:[1/0:23:84, S_ALERTING, E_CC_CONNECT]
*Mar  2 01:20:52.289: //84/F90073EB8080/VTSP:(1/0:23):22:14:2/act_alert_connect: .
*Mar  2 01:20:52.289: //84/F90073EB8080/VTSP:(1/0:23):22:14:2/vtsp_ring_noan_timer_stop:
9125229
```

Related Commands

Command	Description
debug vpm all	Enables all VPM debugging.
debug vtsp port	Limits VTSP debug output to a specific voice port.
show debug	Displays which debug commands are enabled.

debug vtsp stats

To debug periodic statistical-information-request messages sent and received from the digital signal processor (DSP) during a call, use the **debug vtsp stats** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug vtsp stats

no debug vtsp stats

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(3)T	This command was introduced on the Cisco AS5300 universal access servers.
12.0(7)XK	This command was implemented on the Cisco 2600, Cisco 3600 and Cisco MC3810 series.
12.2(11)T	The new debug header was added to the following Cisco routers: Cisco 2600 series, Cisco 3620, and Cisco 3640. and Cisco 3660; on the following universal gateways: Cisco AS5350, Cisco AS5400, and Cisco AS5850; on the following universal access servers: Cisco AS5300, and Cisco AS5800; and, on the Cisco MC3810 multiservice access concentrators.

Usage Guidelines

The **debug vtsp stats** command generates a collection of DSP statistics for generating Real-Time Transport Protocol (RTCP) packets and a collection of other statistical information.

Note We recommend that you log output from the **debug vtsp stats** command to a buffer rather than sending the output to the console; otherwise, the size of the output could severely impact the performance of the gateway.

Related Commands

Command	Description
debug vpm all	Enables all VPM debugging.

Command	Description
debug vtsp port	Limits VTSP debug output to a specific voice port.
show debug	Displays which debug commands are enabled.
voice call debug	Allows configuration of the voice call debug output.

debug vtsp tone

To display debugging messages showing the types of tones generated by the Voice over IP (VoIP) gateway, use the **debug vtsp tone** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug vtsp tone

no debug vtsp tone

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.1(3)XI	This command was introduced.
12.1(5)T	This command was integrated into Cisco IOS Release 12.1(5)T.
12.2(11)T	The new debug header was added to the following Cisco routers: Cisco 2600 series, Cisco 3620, and Cisco 3640. and Cisco 3660; on the following universal gateways: Cisco AS5350, Cisco AS5400, and Cisco AS5850; on the following universal access servers: Cisco AS5300, and Cisco AS5800; and, on the Cisco MC3810 multiservice access concentrators.

Usage Guidelines

We recommend that you log output from the **debug vtsp tone** command to a buffer rather than sending the output to the console; otherwise, the size of the output could severely impact the performance of the gateway.

Related Commands

Command	Description
debug vtsp dsp	Shows messages from the DSP on the modem to the router.
debug vtsp session	Traces how the router interacts with the DSP, based on the signaling indications from the signaling stack and requests from the application.
voice call debug	Allows configuration of the voice call debug output.

debug vtsp vofr subframe

To display the first 10 bytes (including header) of selected Voice over Frame Relay (VoFR) subframes for the interface, use the **debug vtsp vofr subframe** command in privileged EXEC mode . To disable debugging output, use the **no** form of this command.

debug vtsp vofr subframe *payload* **[from-dsp] [to-dsp]**

no debug vtsp vofr subframe

Syntax Description

payload	Number used to selectively display subframes of a specific payload. Payload types are: **0** : Primary Payload **1**: Annex-A**2**: Annex-B**3**: Annex-D**4**: All other payloads**5**: All payloads **Caution** Options 0 and 5 can cause network instability.
from-dsp	Displays only the subframes received from the digital signal processor (DSP).
to-dsp	Displays only the subframes going to the DSP.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(3)XG, 12.0(4)T	This command was introduced on the Cisco 2600 and Cisco 3600 series.
12.0(7)XK	This command was first supported on the Cisco MC3810 series.
12.1(2)T	This command was integrated into Cisco IOS Release 12.1(2)T.
12.2(11)T	The new debug header was added to the following Cisco routers: Cisco 2600 series, Cisco 3620, and Cisco 3640. and Cisco 3660; on the following universal gateways: Cisco AS5350, Cisco AS5400, and Cisco AS5850; on the following universal access servers: Cisco AS5300, and Cisco AS5800; and, on the Cisco MC3810 multiservice access concentrators.

Usage Guidelines

Each debug output displays the first 10 bytes of the FRF.11 subframe, including header bytes. The **from-dsp** and **to-dsp** options can be used to limit the debugs to a single direction. If not specified, debugs are displayed for subframes when they are received from the DSP and before they are sent to the DSP.

Use extreme caution in selecting payload options 0 and 6. These options may cause network instability.

Note We recommend that you log output from the **debug vtsp vofr subframe**command to a buffer rather than sending the output to the console; otherwise, the size of the output could severely impact the performance of the gateway.

Related Commands

Command	Description
debug vpm all	Enables all VPM debugging.
debug vtsp port	Limits VTSP debug output to a specific voice port.
show debug	Displays which debug commands are enabled.
voice call debug	Allows configuration of the voice call debug output.

debug vwic-mft firmware controller

To display debug output from the multiflex (MFT) Voice/WAN interface card (VWIC) controller firmware, use the **debug vwic-mft firmware controller** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug vwic-mft firmware controller {**t1**| **e1**} *slot/port* {**alarm**| **all**| **config**| **fdl**| **loopback**| **register display**| **status**}

no debug vwic-mft firmware controller {**t1**| **e1**} *slot/port* {**alarm**| **all**| **config**| **fdl**| **loopback**| **register display**| **status**}

Syntax Description

t1	Displays debugging messages for T1 channels.
e1	Displays debugging messages for E1 channels.
slot	Slot number. Refer to the appropriate hardware manual for slot information.
port	Port number. Refer to the appropriate hardware manual for port information. The slash mark is required between the *slot* argument and the *port*argument.
alarm	Displays firmware alarm messages.
all	Displays all debugging messages about the MFT VWIC.
config	Displays firmware output messages about configuration change messages sent by the Cisco IOS software.
fdl	Displays firmware output messages when select facilities data link (FDL) events occur.
loopback	Displays firmware output messages when select loopback events occur.
register display	Displays a full framer register value table.
status	Displays current attributes enabled for the specified controller.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(6)	This command was introduced.
12.3(7)T	This command was integrated into Cisco IOS Release 12.3(7)T.

Usage Guidelines

Use the **debug vwic-mft firmware controller** command in privileged EXEC mode to provide firmware-level information for VWICs when information is required beyond the Cisco IOS T1 and E1 controller statistics. The physical-layer information generated by this command includes alarm conditions, line status, controller issues, and register settings, all of which can be used to help troubleshoot MFT VWIC problems.

All the debugging keywords, except **register display**, enable debugging on both ports of a 2-port card. For example, if T1 0/0 and T1 0/1 are two ports on a 2-port MFT card and any of the keywords except **register display** is enabled, debugging output will be generated for both ports because they share a common firmware system.

The Cisco 1- and 2-port T1/E1 multiflex VWICs support voice and data applications in Cisco 2600, Cisco 3600, and Cisco 3700 series multiservice routers. The multiflex VWIC combines WAN interface card and voice interface card functionality.

Caution Use any debugging command with caution because the volume of output generated can slow or stop the router operations. We recommend that this command be used only under the supervision of a Cisco engineer.

Examples

The following sample output displays firmware output about alarm messages for an MFT VWIC installed in slot 0.

```
Router# debug vwic-mft firmware controller e1 0/0 alarm

vwic-mft firmware output messages for wic slot set to: Alarm
Router#
*Mar  4 13:58:14.702: E1T1 0/1  FW: alm1:0e p:01 ALOS LOS LOF
*Mar  4 13:58:15.194: E1T1 0/1  FW:  CERR: 00
*Mar  4 13:58:15.194: E1T1 0/1  FW:  MERR: 00
*Mar  4 13:58:15.194: E1T1 0/1  FW:  FERR: 00
```

Note The output will vary depending on what the router is configured to do after the **debug** command is entered.

The table below describes the significant fields shown in the display.

Table 90: debug vwic-mft firmware controller alarm Field Descriptions

Field	Description
vwic-mft firmware output messages for wic slot set to	Acknowledges that the command has been entered and indicates the current state.

Field	Description
*Mar 4 13:58:14.702: E1T1 0/1 FW	Time-stamp preface that shows that this is a firmware (FW) message. **Note** The port numbers reported here may differ from the numbers configured using the Cisco IOS software because the error is being reported from the second port where debugging has been enabled by the **alarm** keyword on a 2-port MFT card.
alm1:0e	Actual value of the alarm status register.
p:01	Port number of the local VWIC port that is reporting the condition. Value is either 0 or 1 for each port. **Note** The output shows two port numbers; this is an example of the debugging being enabled for both ports on a 2-port MFT card.
ALOS LOS LOF	Shorthand value of current alarm conditions defined in the register. One of the following: • AIS--Receive Alarm Indication Signal • ALOS--Receive Analog Loss of Signal • LOF--Receive Loss of Frame Alignment • LOS--Receive Loss of Signal • MYEL--Receive Multiframe Yellow Alarm • YEL--Receive Yellow Alarm Register value showing the actual value of the alarm status register.
CERR	Status of the error status register; cyclical redundancy check (CRC) block error.
MERR	Status of the error status register; multiframe alignment signal (MFAS) pattern error (E1 only).
FERR	Status of the error status register; framing error.

Related Commands

show controllers e1	Displays information about E1 links.
show controllers t1	Displays information about T1 links.

debug vxml

Note Effective with release 12.3(8)T, the **debug vxml**command is replaced by the **debug voip application vxml**command. See the **debug voip application vxml**command for more information.

To display debugging messages for VoiceXML features, use the **debug vxml** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug vxml [**all**| **application**| **background**| **error**| **event**| **grammar**| **puts**| **ssml**| **trace**| **warning**]

no debug vxml [**all**| **application**| **background**| **error**| **event**| **grammar**| **puts**| **ssml**| **trace**| **warning**]

Syntax Description

all	(Optional) Displays all VoiceXML debugging messages.
application	(Optional) Displays VoiceXML application states information.
background	(Optional) Displays VoiceXML background messages.
error	(Optional) Displays VoiceXML application error messages.
event	(Optional) Displays VoiceXML asynchronous events.
grammar	(Optional) Enables syntax checking of XML grammar by the VoiceXML interpreter and displays syntax debugging messages.
puts	(Optional) Displays the results of VoiceXML <cisco-puts> and <cisco-putvar) tags.
ssml	(Optional) Enables syntax checking of Speech Synthesis Markup Language (SSML) by the VoiceXML interpreter and displays syntax debugging messages.
trace	(Optional) Displays a trace of all activities for the current VoiceXML document.
warning	(Optional) Displays VoiceXML warning messages.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(2)XB	This command was introduced on the Cisco AS5300, Cisco AS5350, and Cisco AS5400.
12.2(11)T	This command was implemented on the Cisco 3640 and Cisco 3660, and the **background**, **grammar**, and **ssml** keywords were added.
12.3(8)T	This command was replaced by the **debug voip application vxml**command.

Usage Guidelines

- The output of this command is affected by the **debug condition application voice** command. If the **debug condition application voice** command is configured and the <cisco-debug> element is enabled in the VoiceXML document, debugging output is limited to the VoiceXML application named in the **debug condition application voice** command.
- The **debug vxml** command enables all VoiceXML debugging messages except those displayed by the **grammar** and **ssml** keywords. The **debug vxml all** command enables all VoiceXML debugging messages including grammar and SSML.

Caution When the **debug vxml grammar** or **debug vxml ssml** command is enabled, the VoiceXML document could abort if there is a fatal syntax error in its eXtensible Markup Language (XML) grammar or SSML.

Examples

The following example shows output from the **debug vxml application** command:

```
Router# debug vxml application
vxml application debugging is on
Router#
1w5d: //-1//VAPP:/vapp_get_apphandler:
1w5d: vapp_get_apphandler: Script callme
1w5d: //-1//VAPP:/vapp_get_apphandler_core:
1w5d: //-1/000000000000/VAPP:/vapp_InterpInitConfigParams:
1w5d: //-1/000000000000/VAPP:/vapp_evt_handler: State VAPP_ACTIVE got event CC_D
1w5d: //-1/000000000000/VAPP:/vapp_driver: pInterp[660E10FC]:
1w5d: //-1/000000000000/VAPP:/vapp_driver: evtID: 28 vapp record state: 0
1w5d: //-1/000000000000/VAPP:/vapp_evt_setup:
1w5d: //-1//VAPP:/vapp_incoming_cal
doc-rtr54-01#lblock:
1w5d: vapp_incoming_callblock:
1w5d: //39/924083218026/VAPP:/vapp_load_or_run_script:
1w5d: //39/924083218026/VAPP:/vapp_load_or_run_script:
1w5d: The VXML Script with len=1450 starts:
----------------------------------------
<?xml version="1.0" encoding="iso-8859-1"?>
<vxml version="1.0">
<property name="fetchtimeout" value="20s"/>
<var name="phone_num"/>
        <form id="main">
```

```
            <noinput>
                <prompt>
                    <audio src="flas
1w5d: //39/924083218026/VAPP:/vapp_media_play:
1w5d: //39/
Router#924083218026/VAPP:/vapp_media_play: prompt=flash:welcome_test.au:
1w5d: //39/924083218026/VAPP:/vapp_checksessionstate:
1w5d: //39/924083218026/VAPP:/vapp_checkifdone:
1w5d: //39/924083218026/VAPP:/vapp_evt_handler: State VAPP_ACTIVE got event CC_E
1w5d: //39/924083218026/VAPP:/vapp_driver: pInterp[660E10FC]:
1w5d: //39/924083218026/VAPP:/vapp_driver: evtID: 36 vapp record state: 0
1w5d: //39/924083218026/VAPP:/vapp_checksessionstate:
1w5d: //39/924083218026/VAPP:/vapp_checkifdo
Router#ne:
1w5d: //39/924083218026/VAPP:/vapp_evt_handler: State VAPP_ACTIVE got event MSWR
1w5d: //39/924083218026/VAPP:/vapp_driver: pInterp[660E10FC]:
1w5d: //39/924083218026/VAPP:/vapp_driver: evtID: 77 vapp record state: 0
1w5d: //39/924083218026/VAPP:/vapp_media_done: evID=77 status=0, protocol=0, st0
1w5d: //39/924083218026/VAPP:/vapp_media_play:
1w5d: //39/924083218026/VAPP:/vapp_media_play: prompt=flash:enter_dest.au:
1w5d: //39/924083218026/VAPP:/vapp_c
Router#hecksessionstate:
1w5d: //39/924083218026/VAPP:/vapp_checkifdone:
Router#
1w5d: //39/924083218026/VAPP:/vapp_evt_handler: State VAPP_ACTIVE got event MSWR
1w5d: //39/924083218026/VAPP:/vapp_driver: pInterp[660E10FC]:
1w5d: //39/924083218026/VAPP:/vapp_driver: evtID: 77 vapp record state: 0
1w5d: //39/924083218026/VAPP:/vapp_media_done: evID=77 status=0, protocol=0, st0
1w5d: //39/924083218026/VAPP:/vapp_digit_collect:
1w5d: //39/924083218026/VAPP:/vapp_checksessionstate:
1w5d: //39/924083218026/VAPP:/vapp_checkifdone:
Router#
1w5d: //39/924083218026/VAPP:/vapp_evt_handler: State VAPP_ACTIVE got event APPE
1w5d: //39/924083218026/VAPP:/vapp_driver: pInterp[660E10FC]:
1w5d: //39/924083218026/VAPP:/vapp_driver: evtID: 87 vapp record state: 0
1w5d: //39/924083218026/VAPP:/vapp_digit_collection_done:
1w5d: //39/924083218026/VAPP:/vapp_digit_collection_done: digits [5551234], sta]
1w5d: //39/924083218026/VAPP:/vapp_gain_control_default:
1w5d: //39/924083218026/VAPP:/vapp_placecall:
Router#1w5d: //39/924083218026/VAPP:/vapp_checksessionstate:
1w5d: //39/924083218026/VAPP:/vapp_checkifdone:
Router#
1w5d: //39/924083218026/VAPP:/vapp_evt_handler: State VAPP_ACTIVE got event APPE
1w5d: //39/924083218026/VAPP:/vapp_driver: pInterp[660E10FC]:
1w5d: //39/924083218026/VAPP:/vapp_driver: evtID: 84 vapp record state: 0
1w5d: //39/924083218026/VAPP:/vapp_evt_setupdone:
1w5d: //39/924083218026/VAPP:/vapp_checksessionstate:
1w5d: //39/924083218026/VAPP:/vapp_checkifdone:
Router#
1w5d: //39/924083218026/VAPP:/vapp_evt_handler: State VAPP_ACTIVE got event CC_D
1w5d: //39/924083218026/VAPP:/vapp_driver: pInterp[660E10FC]:
1w5d: //39/924083218026/VAPP:/vapp_driver: evtID: 15 vapp record state: 0
1w5d: //39/924083218026/VAPP:/vapp_call_disconnected:
1w5d: //39/924083218026/VAPP:/vapp_connection_destroy:
1w5d: //39/924083218026/VAPP:/vapp_checksessionstate:
1w5d: //39/924083218026/VAPP:/vapp_checkifdone:
1w5d: //39/924083218026/VAPP:/vapp_evt_handler: Sta
Router#te VAPP_ACTIVE got event CC_EV_CONF_DESTROY_DONE
1w5d: //39/924083218026/VAPP:/vapp_driver: pInterp[660E10FC]:
1w5d: //39/924083218026/VAPP:/vapp_driver: evtID: 34 vapp record state: 0
1w5d: //39/924083218026/VAPP:/vapp_leg_disconnect:
1w5d: //39/924083218026/VAPP:/vapp_checksessionstate:
1w5d: //39/924083218026/VAPP:/vapp_checkifdone:
1w5d: //39/924083218026/VAPP:/vapp_evt_handler: State VAPP_ACTIVE got event CC_E
1w5d: //39/924083218026/VAPP:/vapp_driver: pInterp[660E10FC]
Router#:
1w5d: //39/924083218026/VAPP:/vapp_driver: evtID: 16 vapp record state: 0
1w5d: //39/924083218026/VAPP:/vapp_terminate:
1w5d: //39/924083218026/VAPP:/vapp_session_exit_event_name: Exit Event vxml.sese
1w5d: //39/924083218026/VAPP:/vapp_checksessionstate:
1w5d: //39/924083218026/VAPP:/vapp_terminate_initiation:
1w5d: //39/924083218026/VAPP:/vapp_checkifdone:
1w5d: //39/924083218026/VAPP:/vapp_evt_handler: State VAPP_CLEANING got event CE
```

```
1w5d: //39/924083218
Router#026/VAPP:/vapp_cleaner:
1w5d: //39/924083218026/VAPP:/vapp_checksessionstate:
1w5d: //39/924083218026/VAPP:/vapp_checkifdone:
1w5d: //39/924083218026/VAPP:/vapp_evt_handler: State VAPP_CLEANING got event AE
1w5d: //39/924083218026/VAPP:/vapp_cleaner:
1w5d: //39/924083218026/VAPP:/vapp_cleaner: VxmlDialogDone event=vxml.session.c0
1w5d: //39/924083218026/VAPP:/vapp_popifdone:
1w5d: //39/924083218026/VAPP:/vapp_checkifdone:
1w5d: //39/924083218026/VAPP:/vapp_
Router#cleanup_apphandler:
1w5d: vapp_cleanup_apphandler: Terminate FALSE Terminated TRUE{HAN[VXML_HAN][NU}
1w5d: //39/924083218026/VAPP:/vapp_free_apphandler: {HAN[VXML_HAN][NULL    ]   }
```

The following example shows output from the **debug vxml background** command:

```
Router# debug vxml background
vxml background messages debugging is on
Router#
1w5d: //-1//VAPP:/vapp_init_apphandler:
1w5d: //-1//VXML:/vxml_create: url=flash:call.vxml vapphandle=660E10FC
Router#
1w5d: //-1//VAPP:/vapp_process: Interp Done
```

The following examples show output from the **debug vxml error** command:

```
Router# debug vxml error
```

This example output shows an error when the version header is missing:

```
*May 10 20:08:57.572://7/98119BD78008/VXML:/vxml_vxml_build:tftp://demo/scripts/test.vxml
at line 2:<vxml version> required attribute missing
*May 10 20:08:57.576://7/98119BD78008/VXML:/vxml_create:
*May 10 20:08:57.576:code=ERROR vapp=VAPP_SUCCESS vxml=VXML_ERROR_INVALID
```

This example output shows an error when a field item is not used according to the DTD:

```
*May 10
20:16:23.315://8/A1BCF458800B/VXML:/vxml_start_element_handler:tftp://demo/scripts/test.vxml
 at line 4:Element <field> is not used according to DTD
*May 10 20:16:23.315://8/A1BCF458800B/VXML:/vxml_create:
*May 10 20:16:23.315:code=ERROR vapp=VAPP_SUCCESS vxml=VXML_ERROR_INVALID
```

This example output shows an error when there is a tag mismatch:

```
*May 10 20:17:44.485://10/D21DEAB58011/VXML:/vxml_parse:tftp://demo/scripts/test.vxml at
line 48:mismatched tag
*May 10 20:17:44.485://10/D21DEAB58011/VXML:/vxml_create:
*May 10 20:17:44.485:code=ERROR vapp=VAPP_SUCCESS vxml=VXML_ERROR_INVALID
```

The following example shows output from the **debug vxml event** command:

```
Router# debug vxml event
vxml events debugging is on
Router#
1w5d: //47/000000000000/VXML:/vxml_media_done: status 0 async_status 100000000
Router#
1w5d: //47/000000000000/VXML:/vxml_media_done: status 0 async_status 300000000
Router#
1w5d: //47/000000000000/VXML:/vxml_digit_collection_done: vxmlp 6534C7C8 status0
1w5d: //47/000000000000/VXML:/vxml_digit_collection_done:  digits 5551234
1w5d: //47/000000000000/VXML:/vxml_digit_collection_done:  name v0
Router#
1w5d: //47/000000000000/VXML:/vxml_placecall_done: duration=0 status=0 async_st0
Router#
1w5d: //47/000000000000/VXML:/vxml_user_hangup: duration 3 status=A async_statu0
```

The following example shows output from the **debug vxml grammar** command:

```
Router# debug vxml grammar
vxml xml grammar syntax checking debugging is on
Router#
Feb 11 13:47:25.110: //-1//VAPP:/vapp_get_apphandler:
```

```
*Feb 11 13:47:25.114: vapp_get_apphandler: Script help
*Feb 11 13:47:25.114: //-1//VAPP:/vapp_get_apphandler_core:
*Feb 11 13:47:25.114: //-1/000000000000/VAPP:/vapp_InterpInitConfigParams:
*Feb 11 13:47:25.114: //-1//VAPP:/vapp_init_apphandler:
*Feb 11 13:47:25.114: //-1/000000000000/VAPP:/vapp_evt_handler: State VAPP_ACTIVE got event
 CC_EV_CALL_SETUP_IND
*Feb 11 13:47:25.114: //-1/000000000000/VAPP:/vapp_driver: pInterp[62DD481C]:
*Feb 11 13:47:25.114: //-1/000000000000/VAPP:/vapp_driver: evtID: 28 vapp record state: 0
*Feb 11 13:47:25.114: //-1/000000000000/VAPP:/vapp_evt_setup:
*Feb 11 13:47:25.114: //-1//VAPP:/vapp_incoming_callblock:
*Feb 11 13:47:25.114: vapp_incoming_callblock:
*Feb 11 13:47:25.114: //7/9AC9CCF28008/VAPP:/vapp_load_or_run_script:
*Feb 11 13:47:25.114: //7/9AC9CCF28008/VAPP:/vapp_load_or_run_script:
*Feb 11 13:47:25.114: The VXML Script with len=741 starts:
-------------------------------------
<?xml version = "1.0"?>
<vxml version = "2.0">
<property name="universals" value="all"/>
<form id="check_help">
    <field name="book">
        <grammar version="1.0" mode="voice" xml:lang="en-US">

*Feb 11 13:47:25.114: //-1//VXML:/vxml_create: url=tftp://dirt/lshen/regression/help.vxml
vapphandle=62DD481C
*Feb 11 13:47:25.114: //-1//VXML:/vxml_mem_init:
*Feb 11 13:47:25.118: //7/9AC9CCF28008/VXML:/vxml_rule_build:
tftp://dirt/lshen/regression/help.vxml at line 8: attribute <rule> with invalid value
(wrong_scope)
*Feb 11 13:47:25.118: //7/9AC9CCF28008/VXML:/vxml_create:
*Feb 11 13:47:25.118: code=ERROR vapp=VAPP_SUCCESS vxml=VXML_ERROR_INVALID
*Feb 11 13:47:25.118: //-1//VXML:/vxml_mem_free:
*Feb 11 13:47:25.118: //-1//VXML:/vxml_mem_free1:
*Feb 11 13:47:25.118: //7/9AC9CCF28008/VAPP:/vapp_terminate:
*Feb 11 13:47:25.118: //7/9AC9CCF28008/VAPP:/vapp_session_exit_event_name: Exit Event
vxml.session.complete
*Feb 11 13:47:25.118: //7/9AC9CCF28008/VAPP:/vapp_checksessionstate:
*Feb 11 13:47:25.118: //7/9AC9CCF28008/VAPP:/vapp_terminate_initiation:
*Feb 11 13:47:25.118: //7/9AC9CCF28008/VAPP:/vapp_checkifdone:
*Feb 11 13:47:25.122: //7/9AC9CCF28008/VAPP:/vapp_evt_handler: State VAPP_CLEANING got event
 CC_EV_CALL_MODIFY_DONE
*Feb 11 13:47:25.122: //7/9AC9CCF28008/VAPP:/vapp_cleaner:
*Feb 11 13:47:25.122: //7/9AC9CCF28008/VAPP:/vapp_cleaner: Ignoring Event
CC_EV_CALL_MODIFY_DONE(36) in Cleanup
*Feb 11 13:47:25.122: //7/9AC9CCF28008/VAPP:/vapp_checksessionstate:
*Feb 11 13:47:25.122: //7/9AC9CCF28008/VAPP:/vapp_checkifdone:
*Feb 11 13:47:25.138: //7/9AC9CCF28008/VAPP:/vapp_evt_handler: State VAPP_CLEANING got event
 CC_EV_CALL_DISCONNECT_DONE
*Feb 11 13:47:25.138: 7/7/9AC9CCF28008/VAPP:/vapp_cleaner:
*Feb 11 13:47:25.138: //7/9AC9CCF28008/VAPP:/vapp_checksessionstate:
*Feb 11 13:47:25.138: //7/9AC9CCF28008/VAPP:/vapp_checkifdone:
*Feb 11 13:47:25.138: //7/9AC9CCF28008/VAPP:/vapp_evt_handler: State VAPP_CLEANING got event
 APP_EV_VXMLINTERP_DONE
*Feb 11 13:47:25.138: //7/9AC9CCF28008/VAPP:/vapp_cleaner:
*Feb 11 13:47:25.138: //7/9AC9CCF28008/VAPP:/vapp_cleaner: VxmlDialogDone
event=vxml.session.complete, status 3
*Feb 11 13:47:25.138: //7/9AC9CCF28008/VAPP:/vapp_popifdone:
*Feb 11 13:47:25.138: //7/9AC9CCF28008/VAPP:/vapp_checkifdone:
*Feb 11 13:47:25.138: //-1//VAPP:/vapp_process: Interp Done
*Feb 11 13:47:25.138: //7/9AC9CCF28008/VAPP:/vapp_cleanup_apphandler:
*Feb 11 13:47:25.138: vapp_cleanup_apphandler: Terminate FALSE Terminated
TRUE{HAN[VXML_HAN][NULL    ]  ( )}
*Feb 11 13:47:25.138: //7/9AC9CCF28008/VAPP:/vapp_free_apphandler: {HAN[VXML_HAN][NULL
]     ( )}
```

The following example shows output from the **debug vxml ssml** command:

```
Router# debug vxml ssml
Router#
vxml ssml syntax checking debugging is on
Feb 11 13:55:28.994: //-1//VAPP:/vapp_get_apphandler:
*Feb 11 13:55:28.994: vapp_get_apphandler: Script help
*Feb 11 13:55:28.994: //-1//VAPP:/vapp_get_apphandler_core:
*Feb 11 13:55:28.994: //-1/A93E3F8F800E/VAPP:/vapp_InterpInitConfigParams:
```

```
*Feb 11 13:55:28.998: //-1//VAPP:/vapp_init_apphandler:
*Feb 11 13:55:28.998: //-1/003E3F8F800E/VAPP:/vapp_evt_handler: State VAPP_ACTIVE got event
 CC_EV_CALL_SETUP_IND
*Feb 11 13:55:28.998: //-1/003E3F8F800E/VAPP:/vapp_driver: pInterp[62DD481C]:
*Feb 11 13:55:28.998: //-1/003E3F8F800E/VAPP:/vapp_driver: evtID: 28 vapp record state: 0
*Feb 11 13:55:28.998: //-1/003E3F8F800E/VAPP:/vapp_evt_setup:
*Feb 11 13:55:28.998: //-1//VAPP:/vapp_incoming_callblock:
*Feb 11 13:55:28.998: vapp_incoming_callblock:
*Feb 11 13:55:28.998: //10/BB2F243F8011/VAPP:/vapp_load_or_run_script:
*Feb 11 13:55:28.998: //10/BB2F243F8011/VAPP:/vapp_load_or_run_script:
*Feb 11 13:55:28.998: The VXML Script with len=760 starts:
-------------------------------------
<?xml version = "1.0"?>
<vxml version = "2.0">
<property name="universals" value="all"/>
<form id="check_help">
    <field name="book">
        <grammar version="1.0" mode="voice" xml:lang="en-US">

*Feb 11 13:55:28.998: //-1//VXML:/vxml_create: url=tftp://dirt/lshen/regression/help.vxml
vapphandle=62DD481C
*Feb 11 13:55:28.998: //-1//VXML:/vxml_mem_init:
*Feb 11 13:55:29.002: //10/BB2F243F8011/VXML:/vxml_parse:
tftp://dirt/lshen/regression/help.vxml at line 16: mismatched tag
*Feb 11 13:55:29.002: //10/BB2F243F8011/VXML:/vxml_create:
*Feb 11 13:55:29.002: code=ERROR vapp=VAPP_SUCCESS vxml=VXML_ERROR_INVALID
*Feb 11 13:55:29.002: //-1//VXML:/vxml_mem_free:
*Feb 11 13:55:29.002: //-1//VXML:/vxml_mem_free1:
*Feb 11 13:55:29.002: //10/BB2F243F8011/VAPP:/vapp_terminate:
*Feb 11 13:55:29.002: //10/BB2F243F8011/VAPP:/vapp_session_exit_event_name: Exit Event
vxml.session.complete
*Feb 11 13:55:29.002: //10/BB2F243F8011/VAPP:/vapp_checksessionstate:
*Feb 11 13:55:29.002: //10/BB2F243F8011/VAPP:/vapp_terminate_initiation:
*Feb 11 13:55:29.002: //10/BB2F243F8011/VAPP:/vapp_checkifdone:
*Feb 11 13:55:29.006: //10/BB2F243F8011/VAPP:/vapp_evt_handler: State VAPP_CLEANING got
event CC_EV_CALL_MODIFY_DONE
*Feb 11 13:55:29.006: //10/BB2F243F8011/VAPP:/vapp_cleaner:
*Feb 11 13:55:29.006: //10/BB2F243F8011/VAPP:/vapp_cleaner: Ignoring Event
CC_EV_CALL_MODIFY_DONE(36) in Cleanup
*Feb 11 13:55:29.006: //10/BB2F243F8011/VAPP:/vapp_checksessionstate:
*Feb 11 13:55:29.006: //10/BB2F243F8011/VAPP:/vapp_checkifdone:
*Feb 11 13:55:29.022: //10/BB2F243F8011/VAPP:/vapp_evt_handler: State VAPP_CLEANING got
event CC_EV_CALL_DISCONNECT_DONE
*Feb 11 13:55:29.022: //10/BB2F243F8011/VAPP:/vapp_cleaner:
*Feb 11 13:55:29.022: //10/BB2F243F8011/VAPP:/vapp_checksessionstate:
*Feb 11 13:55:29.022: //10/BB2F243F8011/VAPP:/vapp_checkifdone:
*Feb 11 13:55:29.022: //10/BB2F243F8011/VAPP:/vapp_evt_handler: State VAPP_CLEANING got
event APP_EV_VXMLINTERP_DONE
*Feb 11 13:55:29.022: //10/BB2F243F8011/VAPP:/vapp_cleaner:
*Feb 11 13:55:29.022: //10/BB2F243F8011/VAPP:/vapp_cleaner: VxmlDialogDone
event=vxml.session.complete, status 3
*Feb 11 13:55:29.022: //10/BB2F243F8011/VAPP:/vapp_popifdone:
*Feb 11 13:55:29.022: //10/BB2F243F8011/VAPP:/vapp_checkifdone:
*Feb 11 13:55:29.022: //-1//VAPP:/vapp_process: Interp Done
*Feb 11 13:55:29.022: //10/BB2F243F8011/VAPP:/vapp_cleanup_apphandler:
*Feb 11 13:55:29.022: vapp_cleanup_apphandler: Terminate FALSE Terminated
TRUE{HAN[VXML_HAN][NULL    ]  ( )}
*Feb 11 13:55:29.022: //10/BB2F243F8011/VAPP:/vapp_free_apphandler: {HAN[VXML_HAN][NULL
 ]    ( )}
```

The following example shows output from the **debug vxml trace** command:

```
Router# debug vxml trace
vxml trace debugging is on
Router#
1w5d: //-1//VXML:/vxml_mem_init:
1w5d: //51/359408288031/VXML:/vxml_offramp_mailhdrs_get:
1w5d: //51/359408288031/VXML:/vxml_start: vxmlhandle=65350A7C vapphandle=660E100
1w5d: //51/359408288031/VXML:/vxml_vxml_proc:
1w5d: <vxml> URI(abs):flash:call.vxml scheme=flash path=call.vxml base= URI(abs0
1w5d: <var>: namep=phone_num
1w5d: //-1//VXML:/vxml_stand_alone: scope=document, application = document
1w5d: //51/359
```

```
Router#408288031/VXML:/vxml_form_proc:
1w5d:  <form>: id=main   scope=dialog
1w5d: vxml_form_init current scope: dialog
1w5d:  vxml_counter_reset:
1w5d:  vxml_counter_reset:
1w5d: //51/359408288031/VXML:/vxml_formitem_select: Status=VXML_STATUS_OK,
1w5d: //51/359408288031/VXML:/vxml_formitem_select:  AsyncStatus=VXML_STATUS_OK
1w5d: //51/359408288031/VXML:/vxml_block_proc:
1w5d:   <block>:
1w5d: //51/359408288031/VXML:/vxml_item_attrs_proc:  name=_in6
1w5d: //51/359408288031/VXML:/vxml_expr_eval: exp
Router#r=dialog._in6='defined'
1w5d: //51/359408288031/VXML:/vxml_prompt_proc:
1w5d:    <prompt>: bargein=0 count=1 typeaheadflush=0
1w5d: //51/359408288031/VXML:/vxml_audio_proc:
1w5d:     <audio>: URI(abs):flash:welcome_test.au scheme=flash path=welcome_tes0
1w5d: //51/359408288031/VXML:/vxml_vapp_media_play: bargein=0 timeout=0 typeahe0
1w5d: //51/359408288031/VXML:/vxml_vapp_media_play:
1w5d: //51/359408288031/VXML:/vxml_vapp_me
Router#dia_play: audio=flash:welcome_test.au cachable=1 timeout20
1w5d: //51/359408288031/VXML:/vxml_leave_scope: scope=8
1w5d: //51/359408288031/VXML:/vxml_vapp_vcr_control_disable:
1w5d: //51/359408288031/VXML:/vxml_start: vxmlhandle=65350A7C vapphandle=660E100
1w5d: //51/359408288031/VXML:/vxml_vxml_proc:
1w5d: <vxml> URI(abs):flash:call.vxml scheme=flash path=call.vxml base= URI(abs0
1w5d: //51/359408288031/VXML:/vxm
Router#l_block_proc:
1w5d:   <block>:
1w5d: //51/359408288031/VXML:/vxml_item_attrs_proc:  name=_in6
1w5d: //51/359408288031/VXML:/vxml_form_proc:
1w5d:  <form>: id=main   scope=dialog
1w5d: //51/359408288031/VXML:/vxml_formitem_select: Status=VXML_STATUS_OK,
1w5d: //51/359408288031/VXML:/vxml_formitem_select:  AsyncStatus=VXML_STATUS_OK
1w5d: //51/359408288031/VXML:/vxml_field_proc:
1w5d:   <field>: type=number
1w5d: //51/359408288031/VXML:/vxml_item_attrs_proc:  name=get_phone_num modal=am
Router#pt_counter=1
1w5d: //51/359408288031/VXML:/vxml_prompt_proc:
1w5d:    <prompt>: bargein=1 count=1 typeaheadflush=0
1w5d: //51/359408288031/VXML:/vxml_audio_proc:
1w5d:     <audio>: URI(abs):flash:enter_dest.au scheme=flash path=enter_dest.au0
1w5d: //51/359408288031/VXML:/vxml_vapp_media_play: bargein=1 timeout=0 typeahe0
1w5d: //51/359408288031/VXML:/vxml_vapp_media_play:
1w5d: //51/359408288031/VXML:/vxml_vapp_media_play: audio
Router#=flash:enter_dest.au cachable=1 timeout20
1w5d: //51/359408288031/VXML:/vxml_vapp_vcr_control_disable:
1w5d: //51/359408288031/VXML:/vxml_vapp_digit_collect: termchar # maxDigits 0 t0
1w5d: //51/359408288031/VXML:/vxml_start: vxmlhandle=65350A7C vapphandle=660E100
1w5d: //51/359408288031/VXML:/vxml_vxml_proc:
1w5d: <vxml> URI(abs):flash:call.vxml scheme=flash path=call.vxml base= URI(abs1
Router#.0
1w5d: //51/359408288031/VXML:/vxml_field_proc:
1w5d:   <field>: type=number
1w5d: //51/359408288031/VXML:/vxml_item_attrs_proc:  name=get_phone_num modal=a2
1w5d: //51/359408288031/VXML:/vxml_filled_proc:
1w5d:
1w5d: <filled>: mode=all
1w5d: //51/359408288031/VXML:/vxml_assign_proc:
1w5d:    <assign>: namep=phone_num expr=get_phone_num
1w5d: //51/359408288031/VXML:/vxml_goto_proc:
1w5d:    <goto>: caching=fast fetchhint=invalid fetchtimeout=20 URI:#transfer_mm
Router#entp=transfer_me
1w5d:    vxml_dialog_reset:
1w5d: //51/359408288031/VXML:/vxml_leave_scope: scope=110
1w5d: //51/359408288031/VXML:/vxml_leave_scope: scope=8
1w5d: //51/359408288031/VXML:/vxml_vxml_proc:
1w5d: <vxml> URI(abs):flash:call.vxml scheme=flash path=call.vxml base= URI(abs0
1w5d: //51/359408288031/VXML:/vxml_form_proc:
1w5d:  <form>: id=transfer_me   scope=dialog
1w5d: vxml_form_init current scope: dialog
1w5d:  <var>: namep=myd
Router#ur
1w5d:  vxml_counter_reset:
```

```
1w5d: //51/359408288031/VXML:/vxml_formitem_select: Status=VXML_STATUS_OK,
1w5d: //51/359408288031/VXML:/vxml_formitem_select:  AsyncStatus=VXML_STATUS_OK
1w5d: //51/359408288031/VXML:/vxml_transfer_proc:
1w5d:   <transfer>:
1w5d: //51/359408288031/VXML:/vxml_item_attrs_proc:  name=mycall dest_expr='phoe
Router#ctreason=-1
1w5d: //51/359408288031/VXML:/vxml_vapp_placecall: dest 5551234 timeout 15 maxl0
1w5d: //51/359408288031/VXML:/vxml_vapp_gain_control_default:
1w5d: //51/359408288031/VXML:/vxml_expr_eval: expr=dialog.mycall = 'far_end_dis'
1w5d: //51/359408288031/VXML:/vxml_expr_eval: expr=dialog.mycall$.duration = 2
1w5d: //51/359408288031/VXML:/vxml_start: vxmlhandle=65350A7C vapphandle=660E100
1w5d: //51/359408288031/VXML:/vxml_vxml
Router#_proc:
1w5d: <vxml> URI(abs):flash:call.vxml scheme=flash path=call.vxml base= URI(abs0
1w5d: //51/359408288031/VXML:/vxml_transfer_proc:
1w5d:   <transfer>:
1w5d: //51/359408288031/VXML:/vxml_item_attrs_proc:  name=mycall URI(abs):phone-
Router#1, redirectreason=-1
1w5d: //51/359408288031/VXML:/vxml_form_proc:
1w5d:  <form>: id=transfer_me   scope=dialog
1w5d: //51/359408288031/VXML:/vxml_filled_proc:
1w5d:
1w5d: <filled>: mode=all
1w5d: //51/359408288031/VXML:/vxml_assign_proc:
1w5d:   <assign>: namep=mydur expr=mycall$.duration
1w5d: //51/359408288031/VXML:/vxml_if_proc:
1w5d:   <if>: cond=mycall == 'busy'
1w5d: //51/359408288031/VXML:/vxml_leave_scope: scope=8
1w5d: //51/359408288031/VXML:/vxml_formitem_select: Status=VXML_ST
Router#ATUS_OK,
1w5d: //51/359408288031/VXML:/vxml_formitem_select:  AsyncStatus=VXML_STATUS_OK
1w5d: //51/359408288031/VXML:/vxml_formitem_select: the form is full
1w5d: //51/359408288031/VXML:/vxml_vapp_terminate: vapp_status=0 ref_count 0
1w5d: //-1//VXML:/vxml_mem_free:
1w5d: //-1//VXML:/vxml_mem_free1:
```

Related Commands

Command	Description
debug condition application voice	Displays debugging messages for only the specified VoiceXML application.
debug http client	Displays debugging messages for the HTTP client.
debug voip ivr	Displays debug messages for VoIP IVR interactions.

debug waas

To enable debugging for WAAS Express modules, use the **debug waas** command in privileged EXEC mode. To disable WAAS Express debugging, use the **no** form of this command.

debug waas {{**auto-discovery**| **aoim**| **cce**| **infrastructure**| **lz**| **memory**| **tfo**} {**events**| **errors**| **operations**}| **api**| **mibs**| **dre** {**events**| **errors**| **operations [brief]**| **uplink**}| **management** {**events**| **errors**}}

no debug waas {{**auto-discovery**| **aoim**| **cce**| **infrastructure**| **lz**| **memory**| **tfo**} {**events**| **errors**| **operations**}| **api**| **mibs**| **dre** {**events**| **errors**| **operations [brief]**| **uplink**}| **management** {**events**| **errors**}}

Syntax Description

auto-discovery	Enables debugging for WAAS Express autodiscovery information.
aoim	Enables debugging for peer information and negotiated capabilities information.
cce	Enables debugging for Common Classification Engine (CCE).
infrastructure	Enables debugging for WAAS Express infrastructure.
lz	Enables debugging for Lempel-Ziv (LZ) optimization.
memory	Enables debugging for WAAS Express internal memory usage.
tfo	Enables debugging for Transport Flow Optimization (TFO).
events	Enables debugging for WAAS Express events.
errors	Enables debugging for WAAS Express errors.
operations	Enables debugging for WAAS Express operations.
brief	Displays WAAS connection operations in brief.
api	Enables debugging for WAAS Express public application programming interfaces (APIs).
mibs	Enables debugging for WAAS Express MIBs.
dre	Enables debugging for Data Redundancy Elimination (DRE) optimization.
uplink	Enables debugging for DRE upload.
management	Enables debugging for error and event management.

Command Default

Debugging is disabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
15.1(2)T	This command was introduced.
15.2(3)T	This command was modified. The **api** and **mibs** keywords were added, and the **brief** keyword was removed.

Examples

The following example shows how to enable debugging output for WAAS Express infrastructure operations:

```
Device> enable
Device# debug waas infrastructure operations
```

Related Commands

Command	Description
clear waas	Clears WAAS Express statistics and closed connections information.
show waas alarms	Displays WAAS Express status and alarms.
show waas auto-discovery	Displays information about WAAS Express autodiscovery.
show waas connection	Displays information about WAAS Express connections.
show waas statistics aoim	Displays WAAS Express peer information and negotiated capabilities.
show waas statistics application	Displays WAAS Express policy application statistics.
show waas statistics auto-discovery	Displays WAAS Express autodiscovery statistics.
show waas statistics class	Displays statistics for the WAAS Express class map.
show waas statistics dre	Displays WAAS Express DRE statistics.
show waas statistics errors	Displays WAAS Express error statistics.
show waas statistics global	Displays global WAAS Express statistics.
show waas statistics lz	Displays WAAS Express LZ statistics.
show waas statistics pass-through	Displays WAAS Express connections placed in a pass-through mode.

Command	Description
show waas statistics peer	Displays inbound and outbound statistics for peer WAAS Express devices.
show waas status	Displays the status of WAAS Express.
show waas token	Displays the value of the configuration token used by the WAAS Central Manager.
waas cm-register url	Registers a device with the WAAS Central Manager.

debug waas accelerator cifs-express

To enable debugging for the Common Internet File System (CIFS)-Express accelerator module of WAAS Express, use the **debug waas accelerator cifs-express** command in privileged EXEC mode. To disable CIFS-Express accelerator debugging, use the **no** form of this command.

debug waas accelerator cifs-express [**ads-negative-cache**| **async-write**| **infra**| **read-ahead**] {**debug**| **events**| **errors**| **file remote-file** *file-URL*| **operations**}

no debug waas accelerator cifs-express [**ads-negative-cache**| **async-write**| **infra**| **read-ahead**] {**debug**| **events**| **errors**| **file remote-file** *file-URL*| **operations**}

Syntax Description

ads-negative-cache	(Optional) Enables debugging of alternate data stream negative caching.
async-write	(Optional) Enables debugging of async write operations.
infra	(Optional) Enables debugging of CIFS-Express accelerator infrastructure.
read-ahead	(Optional) Enables debugging of read ahead operations.
debug	Enables debugging of a specific CIFS-Express parameter, such as async write or read ahead.
events	Enables debugging of CIFS-Express parameter events.
errors	Enables debugging of CIFS-Express parameter errors.
file remote-file *file-URL*	Enables debugging of the CIFS-Express accelerator log file. The format to specify the file URL is *ftp://user:pass@remote_ip/filepathname* and can have up to 500 characters.
operations	Enables debugging of CIFS-Express parameter operations.

Command Default

CIFS-Express accelerator debugging is disabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
15.2(3)T	This command was introduced.

Examples

The following example shows how to enable debugging of CIFS-Express accelerator read ahead errors:

```
Device> enable
Device# debug waas accelerator cifs-express read-ahead errors
```

Related Commands

Command	Description
accelerator	Enters a specific WAAS Express accelerator configuration mode based on the accelerator being configured.
debug waas	Enables debugging for WAAS Express modules.
debug waas accelerator http-express	Enables debugging for the HTTP-Express accelerator module of WAAS Express.
debug waas accelerator ssl-express	Enables debugging for the SSL-Express accelerator module of WAAS Express.
show waas accelerator	Displays information about WAAS Express accelerators.

debug waas accelerator http-express

To enable debugging for the HTTP-Express accelerator module of WAAS Express, use the **debug waas accelerator http-express** command in privileged EXEC mode. To disable HTTP-Express accelerator debugging, use the **no** form of this command.

debug waas accelerator http-express {**infrastructure**| **metadatacache**| **parser**| **transaction**} {**events**| **errors**| **operations**}

no debug waas accelerator http-express {**infrastructure**| **metadatacache**| **parser**| **transaction**} {**events**| **errors**| **operations**}

Syntax Description

infrastructure	Enables debugging of HTTP-Express accelerator infrastructure.
metadatacache	Enables debugging of HTTP metadata cache.
parser	Enables debugging of the HTTP-Express accelerator parser.
transaction	Enables debugging of HTTP-Express accelerator transactions.
events	Enables debugging of HTTP-Express parameter events.
errors	Enables debugging of HTTP-Express parameter errors.
operations	Enables debugging of HTTP-Express parameter operations.

Command Default

HTTP-Express accelerator debugging is disabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
15.2(3)T	This command was introduced.

Examples

The following example shows how to enable debugging of HTTP-Express accelerator parser events:

```
Device> enable
Device(config)# debug waas accelerator http-express parser events
```

Related Commands

Command	Description
accelerator	Enters a specific WAAS Express accelerator configuration mode based on the accelerator being configured.
debug waas	Enables debugging for WAAS Express modules.
debug waas accelerator cifs-express	Enables debugging for the CIFS-Express accelerator module of WAAS Express.
debug waas accelerator ssl-express	Enables debugging for the SSL-Express accelerator module of WAAS Express.
show waas accelerator	Displays information about WAAS Express accelerators.

debug waas accelerator ssl-express

To enable debugging for the Secure Sockets Layer (SSL)-Express accelerator module of WAAS Express, use the **debug waas accelerator ssl-express** command in privileged EXEC mode. To disable SSL-Express accelerator debugging, use the **no** form of this command.

debug waas accelerator ssl-express {**events**| **errors**| **operations**| **messages**}

no debug waas accelerator ssl-express {**events**| **errors**| **operations**| **messages**}

Syntax Description

events	Enables debugging of SSL-Express events.
errors	Enables debugging of SSL-Express errors.
operations	Enables debugging of SSL-Express operations.
messages	Enables debugging of SSL protocol messages.

Command Default

Debugging is disabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
15.2(3)T	This command was introduced.

Examples

The following example shows how to enable debugging of SSL-Express accelerator operations:

```
Device> enable
Device(config)# debug waas accelerator ssl-express operations
```

Related Commands

Command	Description
accelerator	Enters a specific WAAS Express accelerator configuration mode based on the accelerator being configured.
debug waas	Enables debugging for WAAS Express modules.
debug waas accelerator cifs-express	Enables debugging for the CIFS-Express accelerator module of WAAS Express.

Command	Description
debug waas accelerator http-express	Enables debugging for the HTTP-Express accelerator module of WAAS Express.
show waas accelerator	Displays information about WAAS Express accelerators.

debug warm-reboot

To display warm reload debug information, use the **debug warm-reboot** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug warm-reboot

no debug warm-reboot

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(11)T	This command was introduced.

Examples

The following is sample output from the **reload warm file** *url* command when the **debug warm-reboot** command is enabled:

```
Router# debug warm-reboot
Router# reload warm file tftp://9.1.0.1/c7200-p-mz.port
Proceed with reload? [confirm]
Loading c7200-p-mz.port from 9.1.0.1 (via Ethernet5/0):!!!
00:05:43:ptr   :63B978E0
00:05:43:magic :A457272
00:05:43:ptr   :63B98020
00:05:43:magic :0
00:05:43:ptr   :63B98380
00:05:43:magic :0
00:05:43:ptr   :63B983A0
00:05:43:magic :FEEDFACE
00:05:43:uncomp_size         :2749E7C
00:05:43:comp_size           :E966F0
00:05:43:comp_checksum       :9BB36053
00:05:43:uncomp_checksum     :56F1754B!!!
[OK - 15323964 bytes]
Decompressing the image :###
00:06:22:Image checksum correct -1#682743213
00:06:22:Compressed Image checksum correct### [OK]
Number 0     source 0x63BD17C4
Number 1     source 0x63C43AD0
Number 2     source 0x63C83AFC
Number 3     source 0x63CC3B28
.
.
.
Number 156   source 0x66384074
Number 157   source 0x663C40A0
Number 158   source 0x664040CC
wrb_copy_and_launch location = 0x664040CC
00:06:39:Found elf header at the expected location
00:06:39:Source elf_hdr->e_shnum = A
00:06:39:Setting up to copy ELF section 1
00:06:39: to image_info section 0
```

```
00:06:39: sh_name = B
00:06:39: sh_type = 1
00:06:39: sh_flags = 7
00:06:39: sh_addr = 80008000
00:06:39: sh_offset = 60
00:06:39: sh_size = 186C000
00:06:39: sh_link = 0
00:06:39: sh_info = 0
00:06:39: sh_addralign = 20
00:06:39: sh_entsize = 0
.
.
.
00:06:40:Setting up to copy ELF section 4
00:06:40: to image_info section A0
00:06:40: sh_name = 1F
00:06:40: sh_type = 1
00:06:40: sh_flags = 10000003
00:06:40: sh_addr = 82750380
00:06:40: sh_offset = 27483E0
00:06:40: sh_size = 18A0
00:06:40: sh_link = 0
00:06:40: sh_info = 0
00:06:40: sh_addralign = 10
00:06:40: sh_entsize = 0
00:06:40:cpu type                   :19
00:06:40:image_info->entry_point   = 80008000
00:06:40:image_info->section_count = A1
00:06:40:image_info->monstack      = 80007FC0
00:06:40:image_info->monra         = BFC014E4
00:06:40:image_info->param0        = 2
00:06:40:image_info->param1        = 0
00:06:40:image_info->param2        = 80005998
00:06:40:image_info->param3        = 80008000
00:06:40:Section
00:06:40:Section
Decompressed Image checksum correct
             Restricted Rights Legend
.
.
.
```

debug wccp

To display information about all (IPv4 and IPv6) Web Cache Communication Protocol (WCCP) services, use the **debug wccp** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug wccp {**default**| **vrf** *vrf-name* {**events**| **packets [control]**}| **events**| **packets** [**bypass**| **control**| **redirect**]| **platform**| **subblocks**}

no debug wccp {**default**| **vrf** *vrf-name* {**events**| **packets [control]**}| **events**| **packets** [**bypass**| **control**| **redirect**]| **platform**| **subblocks**}

Syntax Description

default	Displays information about default WCCP services.
vrf *vrf-name*	Specifies a virtual routing and forwarding (VRF) instance to associate with a service group.
events	Displays information about significant WCCP events.
packets	Displays information about every WCCP packet received or sent by the router.
control	(Optional) Displays information about WCCP control packets.
bypass	(Optional) Displays information about WCCP bypass packets.
redirect	(Optional) Displays information about WCCP redirect packets.
platform	Displays information about the WCCP platform application programming interface (API).
subblocks	Displays information about WCCP subblocks.

Command Default

Debug information is not displayed.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
15.2(3)T	This command was introduced.

Release	Modification
15.1(1)SY1	This command was integrated into Cisco IOS Release 15.1(1)SY1.

Usage Guidelines

When the **vrf** keyword is not used, the command displays debug information about all WCCP services on the router. The **default** keyword is used to specify default WCCP services.

Examples

The following is sample output from the **debug wccp events** command when a Cisco Cache Engine is added to the list of available Web caches:

```
Router# debug wccp events
WCCP-EVNT: Built I_See_You msg body w/1 usable web caches, change # 0000000A
WCCP-EVNT: Web Cache 192.168.25.3 added
WCCP-EVNT: Built I_See_You msg body w/2 usable web caches, change # 0000000B
WCCP-EVNT: Built I_See_You msg body w/2 usable web caches, change # 0000000C
```

The following is sample output from the **debug wccp packets** command. The router is sending keepalive packets to the Cisco Cache Engines at 192.168.25.4 and 192.168.25.3. Each keepalive packet has an identification number associated with it. When the Cisco Cache Engine receives a keepalive packet from the router, it sends a reply with the identification number back to the router.

```
Router# debug wccp packets
WCCP-PKT: Received valid Here_I_Am packet from 192.168.25.4 w/rcvd_id 00003532
WCCP-PKT: Sending I_See_You packet to 192.168.25.4 w/ rcvd_id 00003534
WCCP-PKT: Received valid Here_I_Am packet from 192.168.25.3 w/rcvd_id 00003533
WCCP-PKT: Sending I_See_You packet to 192.168.25.3 w/ rcvd_id 00003535
WCCP-PKT: Received valid Here_I_Am packet from 192.168.25.4 w/rcvd_id 00003534
WCCP-PKT: Sending I_See_You packet to 192.168.25.4 w/ rcvd_id 00003536
WCCP-PKT: Received valid Here_I_Am packet from 192.168.25.3 w/rcvd_id 00003535
WCCP-PKT: Sending I_See_You packet to 192.168.25.3 w/ rcvd_id 00003537
WCCP-PKT: Received valid Here_I_Am packet from 192.168.25.4 w/rcvd_id 00003536
WCCP-PKT: Sending I_See_You packet to 192.168.25.4 w/ rcvd_id 00003538
WCCP-PKT: Received valid Here_I_Am packet from 192.168.25.3 w/rcvd_id 00003537
WCCP-PKT: Sending I_See_You packet to 192.168.25.3 w/ rcvd_id 00003539
```

Related Commands

Command	Description
clear wccp	Clears the counter for packets redirected using WCCP.
ip wccp	Enables support of the specified WCCP service for participation in a service group.
ipv6 wccp	Enables support of the specified WCCP service for participation in a service group.
ip wccp redirect	Enables packet redirection on an outbound or inbound interface using WCCP.
ipv6 wccp redirect	Enables packet redirection on an outbound or inbound interface using WCCP.

Command	Description
show ip interface	Lists a summary of the IP information and status of an interface.
show ipv6 interface	Lists a summary of the IP information and status of an interface.

debug webvpn

To enable the display of debug information for SSL VPN applications and network activity, use the **debug webvpn**command in privileged EXEC mode. To stop debugging messages from being processed and displayed, use the **no** form of this command.

debug webvpn [verbose] [**aaa| acl| cifs**| **citrix [verbose]**| **cookie [verbose]| count| csd| data| dns**| **emweb [state]**| **entry** *context-name* [**source** *ip* [*network-mask*]| **user** *username*]| **http** [**authentication| trace| verbose**]| **package**| **sdps** [**level** *number*]| **sock [flow]| sso| timer| trie**| **tunnel** [**traffic** *acl-number*| **verbose**]| **url-disp**| **webservice [verbose]**]

no debug webvpn [verbose] [**aaa| acl| cifs**| **citrix [verbose]**| **cookie [verbose]| count| csd| data| dns**| **emweb [state]**| **entry** *context-name* [**source** *ip* [*network-mask*]| **user** *username*]| **http** [**authentication| trace| verbose**]| **package**| **sdps** [**level** *number*]| **sock [flow]| sso| timer| trie**| **tunnel** [**traffic** *acl-number*| **verbose**]| **url-disp**| **webservice [verbose]**]

Syntax Description

verbose	(Optional) Detailed information about SSL VPN applications and network activity is displayed in addition to the nondetailed information.
aaa	(Optional) Displays authentication, authorization, and accounting (AAA) event and error messages.
acl	(Optional) Displays information about the Application Layer access control list (ACL).
cifs	(Optional) Displays Microsoft Windows file share access event and error messages.
citrix [**verbose**	(Optional) Displays Citrix application event and error messages. • **verbose** (Optional)--All detailed and nondetailed citrix messages are displayed. If the **verbose** keyword is not used, only the nondetailed messages are displayed.
cookie [**verbose**	(Optional) Displays event and error messages that relate to the cookie that is pushed to the browser of the end user. • **verbose** (Optional)--All detailed and nondetailed cookie messages are displayed. If the **verbose** keyword is not used, only the nondetailed messages are displayed.
count	(Optional) Displays reference count information for a context.

csd	(Optional) Displays Cisco Secure Desktop (CSD) event and error messages.
data	(Optional) Displays data debug messages.
dns	(Optional) Displays domain name system (DNS) event and error messages.
emweb [**state**	(Optional) Displays emweb state debug messages.
entry *context-name* [**source** *ip* [*network-mask*] | **user** *username*	(Optional) Displays information for a specific user or group. • *context-name*-- SSL VPN context name. • **source** *ip* (Optional)--IP address of the user or group. The *network-mask*argument is optional. If not specified, 255.255.255.255 is used. • **user** *username* (Optional)-- Username of the user. **Note** The **entry** keyword can be used with other **debug** commands to single out the debug messages for a particular user or group. If the **debug webvpn** entry is not defined, the debug messages of the feature or function that are turned on are printed for every user.
http [**authentication**| **trace**| **verbose**	(Optional) Displays HTTP debug messages. • **authentication** (Optional)--Displays information for HTTP authentication, such as NT LAN Manager (NTLM). • **trace** (Optional)--Displays HTTP information that involves EmWeb processing. • **verbose** (Optional)--All detailed and nondetailed HTTP messages are displayed. If the **verbose** keyword is not used, only the nondetailed messages are displayed.
package	(Optional) Deploys event and error messages for the software packages that are pushed to the end user.
sdps [**level** *number*]	(Optional) Displays SDPS debug messages. The level is entered as a number from 1 to 5.
sock [**flow**]	(Optional) Displays socket debug messages.

sso	(Optional) Displays information about Single SignOn (SSO) ticket creation, session setup, and response handling.
timer	(Optional) Displays timer debug messages.
trie	(Optional) Displays trie debug messages.
tunnel [**traffic** *acl-number* | **verbose**]	(Optional) Displays tunnel debug messages. • **traffic** *acl-number* (Optional)--Access control list number of the traffic to be displayed. • **verbose** (Optional)--All detailed and nondetailed tunnel messages are displayed. If the **verbose** keyword is not used, only the nondetailed messages are displayed.
url-disp	(Optional) Displays URL debug messages.
webservice [**verbose**]	(Optional) Displays web service event and error messages. • **verbose** (Optional)--All detailed and nondetailed web service messages are displayed. If the **verbose** keyword is not used, only the nondetailed messages are displayed.

Command Default

None

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(14)T	This command was introduced.
12.4(6)T	Support for the SSL VPN enhancements feature was added.

Release	Modification
12.4(11)T	The following keywords were deleted effective with Cisco IOS Release 12.4(11)T: • **port-forward** • **detail** keyword option for the **tunnel** keyword The following keywords and arguments were added effective with Cisco IOS Release 12.4(11)T: • **verbose** • **acl** • **entry** *context-name* [**source** *ip* [*network-mask*] | **user** *username* • **authentication** , **trace**, and **verbose** keyword options for the **http** keyword • **sso** • **verbose** keyword option for the **citrix**, **cookie**, **tunnel**, and **webservice** keywords

Usage Guidelines

This command should be used with caution on a production router or networking device. It is recommended that debugging is enabled only for individual components as necessary. This restriction is intended to prevent the console session from be overwhelmed by large numbers of messages.

The **no** form of this command turns off feature debugging. It does not matter if the **verbose** keyword has been used or not.

If the **no** form of this command is used with the **verbose** keyword option for any keyword, all keyword and argument fields must be an exact match.

Examples

Examples

The following example displays **debug webvpn** output for various SSL VPN sessions:

```
Router# debug webvpn
*Dec 23 07:47:41.368: WV: Entering APPL with Context: 0x64C5F270,
      Data buffer(buffer: 0x64C877D0, data: 0x4F27B638, len: 272,
      offset: 0, domain: 0)
*Dec 23 07:47:41.368: WV: http request: /sslvpn with domain cookie
*Dec 23 07:47:41.368: WV: Client side Chunk data written..
 buffer=0x64C877B0 total_len=189 bytes=189 tcb=0x6442FCE0
*Dec 23 07:47:41.368: WV: sslvpn process rcvd context queue event
*Dec 23 07:47:41.372: WV: sslvpn process rcvd context queue event
*Dec 23 07:47:41.372: WV: Entering APPL with Context: 0x64C5F270,
      Data buffer(buffer: 0x64C877D0, data: 0x4F26D018, len: 277,
      offset: 0, domain: 0)
*Dec 23 07:47:41.372: WV: http request: /webvpn.html with domain cookie
*Dec 23 07:47:41.372: WV: [Q]Client side Chunk data written..
 buffer=0x64C877B0 total_len=2033 bytes=2033 tcb=0x6442FCE0
*Dec 23 07:47:41.372: WV: Client side Chunk data written..
 buffer=0x64C87710 total_len=1117 bytes=1117 tcb=0x6442FCE0
```

Examples

The following example displays information for a specific user (user1 under the context “mycontext”) and for a feature or function:

```
Router# debug webvpn entry mycontext_user_user1
! The above line turns debugging on for user1.
! The following line turns on debugging for a feature (or features) or function (or
functions)--in this case; for authentication, authorization, and accounting (AAA).
Router# debug webvpn aaa
```

The actual output is as follows:

```
*Dec 23 07:56:41.351: WV-AAA: AAA authentication request sent for user: "user1"
*Dec 23 07:56:41.351: WV-AAA: AAA Authentication Passed!
*Dec 23 07:56:41.351: WV-AAA: User "user1" has logged in from "10.107.163.147" to gateway
"sslvpn" context "mycontext"
*Dec 23 07:59:01.535: WV-AAA: User "user1" has logged out from gateway "sslvpn" context
"mycontext"
```

Examples

The following example displays cookie and HTTP information for a group of users under the context “mycontext” having a source IP range from 192.168.1.1. to 192.168.1.255:

```
Router# debug webvpn entry mycontext source 192.168.1.0 255.255.255.0
! The above command line sets up debugging for the group.
!The following command lines turn on debugging for cookie and HTTP information.
Router# debug webvpn cookie
Router# debug webvpn http
```

The actual output is as follows:

```
*Dec 23 08:10:11.191: WV-HTTP: Original client request
GET /webvpn.html HTTP/1.1

*Dec 23 08:10:11.191: WV-HTTP: HTTP Header parsing complete
*Dec 23 08:10:11.191: WV-HTTP: * HTTP request complete
*Dec 23 08:10:11.191: WV-COOKIE: Enter VW context cookie check with Context:0x64C5F470,
          buffer: 0x64C87710, buffer->data: 0x4F26D018, buffer->len: 277,
          cookie: 0x4F26D10A, length: 33
*Dec 23 08:10:11.191: WV-COOKIE: webvpn context cookie received is webvpncontext=00@mycontext

*Dec 23 08:10:11.191: WV-COOKIE: context portion in context cookie is: mycontext

*Dec 23 08:10:11.327: WV-HTTP: Original client request
GET /paramdef.js HTTP/1.1

*Dec 23 08:10:11.327: WV-HTTP: HTTP Header parsing complete
*Dec 23 08:10:11.327: WV-HTTP: * HTTP request complete
```

Examples

The following output example displays information about SSO ticket creation, session setup, and response handling:

```
Router# debug webvpn sso
*Jun 12 20:37:01.052: WV-SSO: Redirect to SSO web agent URL -
http://example.examplecompany.com/vpnauth/
*Jun 12 20:37:01.052: WV_SSO: Set session cookie with SSO redirect
*Jun 12 20:37:01.056: WV-SSO: Set SSO auth flag
*Jun 12 20:37:01.056: WV-SSO: Attach credentials - building auth ticket
*Jun 12 20:37:01.060: WV-SSO: user: [user11], secret: [example123], version: [1.0], login
time: [BCEFC86D], session key: [C077F97A], SHA1 hash :
[B07D0A924DB33988D423AE9F937C1C5A66404819]
*Jun 12 20:37:01.060: WV-SSO: auth_ticket :
user11:1.0@C077F97A@BCEFC86D@B07D0A924DB33988D423AE9F937C1C5A66404819
*Jun 12 20:37:01.060: WV-SSO: Base64 credentials for the auth_ticket:
```

```
dXNlcjExOjEuMEBDMDc3Rjk3QUBCQ0VGQzg2REBCMDdEMEE5MjREQjMzOTg4RDQyM0FFOUY5MzdDMUM1QTY2NDA0ODE5
*Jun 12 20:37:01.060: WV-SSO: Decoded credentials =
user11:1.0@C077F97A@BCEFC86D@B07D0A924DB33988D423AE9F937C1C5A66404819
*Jun 12 20:37:01.060: WV-SSO: Starting SSO request timer for 15-second
*Jun 12 20:37:01.572: WV-SSO: SSO auth response rcvd - status[200]
*Jun 12 20:37:01.572: WV-SSO: Parsed non-SM cookie: SMCHALLENGE
*Jun 12 20:37:01.576: WV-SSO: Parsed SMSESSION cookie
*Jun 12 20:37:01.576: WV-SSO: Sending logon page after SSO auth success
```

debug webvpn dtls

To enable the display of Secure Socket Layer Virtual Private Network (SSL VPN) Datagram Transport Layer Security (DTLS) debug information, use the **debug webvpn dtls** command in privileged EXEC mode. To stop debugging messages from being processed and displayed, use the **no** form of this command.

debug webvpn dtls [**errors**| **events**| **packets**]

no debug webvpn dtls [**errors**| **events**| **packets**]

Syntax Description

errors	(Optional) Displays errors that might have occurred while setting up DTLS tunnel or during data transfer.
events	(Optional) Displays DTLS event messages. Displays events like encryption, decryption, switching, and so on.
packets	(Optional) Displays DTLS packet dump.

Command Default

If no keyword is specified, then all the SSL VPN DTLS debug information displays are enabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
15.1(2)T	This command was introduced.

Usage Guidelines

You can use the **debug webvpn dtls** command to debug any issues related to WebVPN DTLS.

This debug information provides information about the packets that are being processed by WebVPN DTLS and indicates if there are any errors.

Examples

The following example displays the SSL VPN DTLS packet dump information:

```
Router# debug webvpn dtls packets

*Jun 15 10:23:04.495: WV-DTLS: pak (0x67EEF474), dgram (109), length (0) encsize(0)
2E6FBD10:                     17010000 01000000          .......
2E6FBD20: 00004C00 6057A7E2 399F19CF 9915D3F4  ..L.`W'b9..O..St
2E6FBD30: 4FBA7F24 8AEC4EFC 9F4192B5 D334F471  O:.$.lN|.A.5S4tq
2E6FBD40: 02232ADF BB248C8B 54E197F5 713D7886  .#*_;$..Ta.uq=x.
2E6FBD50: 4F71398D 993342BA 90D2A677 96A6ABB9  Oq9¯.3B:.R&w.&+9
2E6FBD60: 8B72F19C 4D454CBB A74D2342 B643FA74  .rq.MEL;'M#B6Czt
2E6FBD70: A627656A E1DDF0A9 ABDAC6FC 7986FC52  &'eja]p)+ZF|y.|R
```

```
2E6FBD80: AD9AF67D C5                          -.v}E
*Jun 15 10:23:04.499: WV-DTLS: pak (0x67EEB7A8), dgram (137), length (0) encsize(0)
2E6FA4D0:                  45000089 FCE80000          E...|h..
2E6FA4E0: FF11761F 1E010132 28010128 01BB0CBA  ..v....2(..(.;.:
2E6FA4F0: 0075ECF8 17010000 01000000 00004C00  .ulx..........L.
2E6FA500: 6057A7E2 399F19CF 9915D3F4 4FBA7F24  `W'b9..O..StO:.$
2E6FA510: 8AEC4EFC 9F4192B5 D334F471 02232ADF  .lN|.A.5S4tq.#*_
2E6FA520: BB248C8B 54E197F5 713D7886 4F71398D  ;$..Ta.uq=x.Oq9.
2E6FA530: 993342BA 90D2A677 96A6ABB9 8B72F19C  .3B:.R&w.&+9.rq.
2E6FA540: 4D454CBB A74D2342 B643FA74 A627656A  MEL;'M#B6Czt&'ej
2E6FA550: E1DDF0A9 ABDAC6FC 7986FC52 AD9AF67D  a]p)+ZF|y.|R-.v}
2E6FA560: C5
```

The following example displays the SSL VPN DTLS event information:

```
Router# debug webvpn dtls events

*Jun 15 10:28:13.731: WV-DTLS-3 Decryption done: context (0x67BF9BA0) pak (0x67634074),
ce_status = (1)
*Jun 15 10:28:14.575: WV-DTLS-3 Decryption done: context (0x67BF9BA0) pak (0x67634074),
ce_status = (1)
*Jun 15 10:28:14.575: WV-DTLS-3 Encryption done: context (0x67BF9BA0) pak (0x65EF4778),
ce_status = (1)
*Jun 15 10:28:14.575: WV-DTLS-2 DTLS: Switching cont pak in process path
*Jun 15 10:28:15.575: WV-DTLS-3 Decryption done: context (0x67BF9BA0) pak (0x67634074),
ce_status = (1)
*Jun 15 10:28:15.579: WV-DTLS-3 Encryption done: context (0x67BF9BA0) pak (0x66B2AAD4),
ce_status = (1)
*Jun 15 10:28:15.579: WV-DTLS-2 DTLS: Switching cont pak in process path
*Jun 15 10:28:16.575: WV-DTLS-3 Decryption done: context (0x67BF9BA0) pak (0x67634074),
ce_status = (1)
*Jun 15 10:28:16.575: WV-DTLS-3 Encryption done: context (0x67BF9BA0) pak (0x65EF4C04),
ce_status = (1)
*Jun 15 10:28:16.575: WV-DTLS-2 DTLS: Switching cont pak in process path
*Jun 15 10:28:17.579: WV-DTLS-3 Decryption done: context (0x67BF9BA0) pak (0x67634074),
ce_status = (1)
*Jun 15 10:28:17.579: WV-DTLS-3 Encryption done: context (0x67BF9BA0) pak (0x66B298A4),
ce_status = (1)
*Jun 15 10:28:17.579: WV-DTLS-2 DTLS: Switching cont pak in process path
*Jun 15 10:28:18.579: WV-DTLS-3 Decryption done: context (0x67BF9BA0) pak (0x67634074),
ce_status = (1)
*Jun 15 10:28:18.579: WV-DTLS-3 Encryption done: context (0x67BF9BA0) pak (0x65EF74F0),
ce_status = (1)
*Jun 15 10:28:18.579: WV-DTLS-2 DTLS: Switching cont pak in process path
*Jun 15 10:28:19.579: WV-DTLS-3 Decryption done: context (0x67BF9BA0) pak (0x67634074),
ce_status = (1)
*Jun 15 10:28:19.583: WV-DTLS-3 Encryption done: context (0x67BF9BA0) pak (0x65EF6BD8),
ce_status = (1)
*Jun 15 10:28:19.583: WV-DTLS-2 DTLS: Switching cont pak in process path
```

Related Commands

Command	Description
dtls port	Configures a desired port for the DTLS to listen.
svc dtls	Enables DTLS support on the Cisco IOS SSL VPN.

debug webvpn license

To display information related to license operations, events, and errors, use the **debug webvpn license**command in privileged EXEC mode. To disable the debugging output, use the **no** form of this command.

debug webvpn license

no debug webvpn license

Syntax Description

This command has no arguments or keywords.

Command Default

Debug messages are not displayed.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
15.0(1)M	This command was introduced.

Examples

The following is sample output from the **debug webvpn license** command when there is no valid license, and a user tries to log in to SSL VPN:

```
*Sep 17 09:36:21.091: %SSLVPN-3-LICENSE_NO_LICENSE: No valid license is available to use
IOS SSLVPN service
*Sep 17 09:36:21.091: WV-License: no valid reserve handle exists, request is not made
*Sep 17 09:36:21.091: WV-AAA: Error! No valid SSLVPN license exists
```

The following is sample output from the **debug webvpn license** command when there is a valid license, and a user tries to log in:

```
*Sep 17 09:40:15.535: WV-License: requested 1 count, granted 1 count, status is : No Error
```

The following is sample output from the **debug webvpn license** command when a user logs out and closes his or her session:

```
*Sep 17 09:41:48.143: WV-License: trying to release 1 count, released 1 count, status is :
 No Error
```

The following is sample output from the **debug webvpn license** command when the currently active license is a temporary (nonpermanent) license, and it has expired; some sessions are still active:

```
*Sep 18 00:28:19.018: WV-License: received licensing event for handle 0x1000004
*Sep 18 00:28:19.018:        Event type : LICENSE_CLIENT_EXPIRED
*Sep 18 00:28:19.018:        Count [usage/max(new max)]: 0/0(0)
*Sep 18 00:28:19.018: WV-License: setting lic expired flag!
*Sep 18 00:28:19.018: %SSLVPN-3-LICENSE_EXPIRED: IOS SSLVPN evaluation/extension license
has expired
*Sep 18 00:28:19.018: WV-License: event handling completed
*Sep 18 00:28:19.078: %LICENSE-2-EXPIRED: License for feature SSL_VPN_Test_Feature 1.0 has
 expired now.. UDI=CISCO2821:FHK1110F0PF
```

The following is sample output from the **debug webvpn license** command when the currently active license is a temporary (nonpermanent) license, and it has expired; some sessions are still active and a new user tries to log in:

```
*Sep 18 00:29:18.078: WV-AAA: AAA authentication request sent for user: "lab"
*Sep 18 00:29:18.078: WV-AAA: AAA Authentication Passed!
*Sep 18 00:29:18.078: %SSLVPN-3-LICENSE_EXPIRED: IOS SSLVPN evaluation/extension license
has expired
*Sep 18 00:29:18.078: WV-License: License expired, no more counts can be requested!
*Sep 18 00:29:18.078: WV-AAA: Error! No valid SSLVPN license exists
```

The following is sample output from the **debug webvpn license** command when a new license having a count higher than the currently active license is installed:

```
*Sep 18 00:39:12.658: WV-License: received licensing event
*Sep 18 00:39:12.658:        Event type : LICENSE_CLIENT_COUNT_CHANGED
*Sep 18 00:39:12.658:        Count [usage/max(new max)]: 0/0(169)
*Sep 18 00:39:12.770: WV-License: reserved extra count (158): No Error
*Sep 18 00:39:12.770: WV-License: reserved count now is 169
*Sep 18 00:39:12.774: WV-License: event handling completed
*Sep 18 00:39:12.774: WV-License: received licensing event for handle 0x1000004
*Sep 18 00:39:12.774:        Event type : LICENSE_CLIENT_COUNT_CHANGED
*Sep 18 00:39:12.774:        Count [usage/max(new max)]: 0/0(169)
```

The above outputs are self-explanatory.

Related Commands

Command	Description
show webvpn license	Displays the available count and the current usage.

debug wlccp ap

Use the debug wlccp ap privileged EXEC command to enable debugging for devices that interact with the access point that provides wireless domain services (WDS).

debug wlccp ap {**mn**| **rm** [**statistics**| **context**| **packet**]| **state**| **wds-discovery**}

Syntax Description

Command	Description
mn	(Optional) Activates display of debug messages related to client devices
rm [**statistics** | **context** | **packet**]	(Optional) Activates display of debug messages related to radio management • **statistics** --shows statistics related to radio management • **context** --shows the radio management contexts • **packet** --shows output related to packet flow
state	(Optional) Activates display of debug messages related to access point authentication to the WDS access point
wds-discovery	(Optional) Activates display of debug messages related to the WDS discovery process

Command Default

Debugging is not enabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(11)JA	This command was first introduced.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.

Usage Guidelines

This command is not supported on bridges.

Examples

This example shows how to begin debugging for LEAP-enabled client devices participating in Cisco Centralized Key Management (CCKM):

```
SOAP-AP# debug wlccp ap mn
```

Related Commands

Command	Description
show debugging	Displays all debug settings and the debug packet headers
show wlccp	Displays WLCCP information

debug wlccp ap rm enhanced-neighbor-list

Use the **debug wlccp ap rm enhanced-neighbor-list** privileged EXEC command to enable internal debugging information and error messages of the Enhanced Neighbor List feature. Use the **no** form of the command to disable the debugging and error messages.

[no] debug wlccp ap rm enhanced-neighbor-list

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging is not enabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.3(8)JA	This command was first introduced.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.

Usage Guidelines

This command is not supported on bridges.

Examples

This example shows how to activate debugging and error messages of the Enhanced Neighbor List feature on the access point:

```
SOAP-AP# debug wlccp ap rm enhanced-neighbor-list
```

Related Commands

Command	Description
show debugging	Displays all debug settings and the debug packet headers
show wlccp	Displays WLCCP information
show wlccp ap rm enhanced-neighbor-list	Displays Enhanced Neighbor List feature related information.

debug wlccp packet

To display the packets being delivered to and from the wireless domain services (WDS) device, use the **debug wlccp packet**command in privileged EXEC mode. To disable the display of packets, use the **no** form of this command.

debug wlccp packet

no debug wlccp packet

Syntax Description

This command has no arguments of keywords.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(11)JA	This command was introduced on Cisco Aironet access points.
12.3(11)T	This command was implemented on the following platforms: Cisco 2600XM, Cisco 2691, Cisco 2811, Cisco 2821, Cisco 2851, Cisco 3700, and Cisco 3800 series routers.

Related Commands

Command	Description
debug wlccp wds	Displays either WDS debug state or WDS statistics messages.
show wlccp wds	Shows information about access points and client devices on the WDS router.
wlccp authentication-server client	Configures the list of servers to be used for 802.1X authentication.
wlccp authentication-server infrastructure	Configures the list of servers to be used for 802.1X authentication for the wireless infrastructure devices.
wlccp wds priority interface	Enables a wireless device such as an access point or a wireless-aware router to be a WDS candidate.

debug wlccp rmlib

Use the debug wlccp rmlib privileged EXEC command to activate display of radio management library functions on the access point that provides wireless domain services (WDS).

debug wlccp rmlib

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(13)JA	This command was first introduced.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.

Usage Guidelines

This command is not supported on bridges.

Examples

This example shows how to activate display of radio management library functions on the access point that provides WDS:

```
SOAP-AP# debug wlccp rmlib
```

Related Commands

Command	Description
show debugging	Displays all debug settings and the debug packet headers
show wlccp	Displays WLCCP information

debug wlccp wds

To display wireless domain services (WDS) debug messages, state messages, and failure statistics, use the **debug wlccp wds**command in privileged EXEC mode. To disable debug output, use the **no** form of this command.

debug wlccp wds {**authenticator**| **state**| **statistics**}

no debug wlccp wds

Syntax Description

authenticator	MAC and Extensible Authentication Protocol (EAP) authentication.
state	WDS state and debug messages.
statistics	WDS failure statistics.

Command Default

No default behavior or values

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(11)JA	This command was introduced.
12.3(11)T	This command was implemented on the following platforms: Cisco 2600XM, Cisco 2691, Cisco 2811, Cisco 2821, Cisco 2851, Cisco 3700, and Cisco 3800 series routers.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.

Examples

The following command displays WDS failure statistics:

```
Router# debug wlccp wds
statistics
```

Related Commands

Command	Description
debug wlccp packet	Displays packet traffic to and from the WDS router.

Command	Description
show wlccp wds	Shows information about access points and client devices on the WDS router.
wlccp authentication-server client	Configures the list of servers to be used for 802.1X authentication.
wlccp authentication-server infrastructure	Configures the list of servers to be used for 802.1X authentication for the wireless infrastructure devices.
wlccp wds priority interface	Enables a wireless device such as an access point or a wireless-aware router to be a WDS candidate.

debug wsma agent

To display debugging information on all Web Services Management Agents (WSMAs), use the **debug wsma agent** command in privileged EXEC mode. To disable the debugging information on all WSMAs, use the **no** form of this command.

debug wsma agent [**config**| **exec**| **filesys**| **notify**]

no debug wsma agent

Syntax Description

config	(Optional) Displays debugging information for the configuration agent.
exec	(Optional) Displays debugging information for the executive agent.
filesys	(Optional) Displays debugging information for the file system agent.
notify	(Optional) Displays debugging information for the notify agent.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.4(24)T	This command was introduced.
12.2(50)SY	This command was integrated into Cisco IOS Release 12.2(50)SY.
15.1(1)SG	This command was integrated into Cisco IOS Release 15.1(1)SG.
IOS XE Release 3.3SG	This command was integrated into Cisco IOS XE Release 3.3SG.
15.1(1)SY	This command was integrated into Cisco IOS Release 15.1(1)SY.

Examples

The following example shows how to display debugging information for a WSMA listener profile:

```
Router# debug wsma agent config

WSMA agent config debugging is on
```

Related Commands

Command	Description
debug wsma profile	Displays debugging information for all WSMA profiles.

debug wsma profile

To display debugging information on all Web Services Management Agent (WSMA) profiles, use the **debug wsma profile** command in privileged EXEC mode. To disable the debugging information on all WSMA profiles, use the **no** form of this command.

debug wsma profile [**listener**| **initiator**]

no debug wsma profile

Syntax Description

listener	Displays debugging information for the listener profile.
initiator	Displays debugging information for the initiator profile.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.4(24)T	This command was introduced.
15.1(1)T	This command was modified. The **initiator** keyword was added.
12.2(50)SY	This command was integrated into Cisco IOS Release 12.2(50)SY.
15.1(1)SG	This command was integrated into Cisco IOS Release 15.1(1)SG.
IOS XE Release 3.3SG	This command was integrated into Cisco IOS XE Release 3.3SG.
15.1(1)SY	This command was integrated into Cisco IOS Release 15.1(1)SY.

Examples

The following example shows how to display debugging information for a WSMA listener profile:

```
Router# debug wsma profile listener

WSMA profile listener debugging is on
```

The following example shows how to display debugging information for a WSMA initiator profile:

```
Router# debug wsma profile initiator

WSMA profile initiator debugging is on
```

Related Commands

Command	Description
debug wsma agent	Displays debugging information for all WSMAs.

debug wsapi

To collect and display traces for the Cisco Unified Communication IOS services application programming interface, use the **debug wsapi** command in privileged EXEC mode. To disable debugging, use the **no** form of this command.

debug wsapiinfrastructure| xcc| xcdr| xsvcall| default| detail| error| event| function| inout| messages

no debug wsapiinfrastructure| xcc| xcdr| xsvcall| default| detail| error| event| function| inout| messages

Syntax Description

infrastructure	Enables debugging traces on the infrastructure.
xcc	Enables debugging traces on the xcc provider.
xcdr	Enables debugging traces on the xcdr provider.
xsvc	Enables debugging traces on the xsvc provider
all	Enables all debugging traces.
default	Enables default debugging traces.
detail	Enables detailed debugging traces.
error	Enables error debugging traces.
event	Enables event debugging traces.
function	Enables function debugging traces.
inout	Enables inout debugging traces.
messages	Enables API message traces.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
15.2(2)T	This command was introduced.

Usage Guidelines

Use this command to enable debugging traces for the Cisco Unified Communicaion IOS services subsystems.

Examples

The following is a sample output from the **debug wsapi infrastructure** command for an XCC registration.

```
Router# debug wsapi infrastructure

23:25:09: //WSAPI/INFRA/wsapi_https_urlhook:
23:25:09: //WSAPI/INFRA: app_name cisco_xcc in url /cisco_xcc in port 8090
23:25:09: //WSAPI/INFRA/wsapi_https_urlhook: Exit
23:25:09: //WSAPI/INFRA/wsapi_https_post_action:
23:25:09: wsapi_https_data_read: <soapenv:Envelope
xmlns:soapenv="http://www.w3.org/2003/05/soap-envelope"><soapenv:Body><RequestXccRegister
xmlns="http://www.cisco.com/schema/cisco_xcc/v1_0"><applicationData><name>myapp</name><url>http://[illegible]/xcc</url></applicationData>[illegible]CREATED
 AUTHORIZE_CALL REDIRECTED ALERTING CONNECTED TRANSFERRED CALL_DELIVERY DISCONNECTED
HANDOFFLEAVE
HANDOFFJOIN</connectionEventsFilter><mediaEventsFilter>MODE_CHANGE DTMF TONE_BUSY TONE_DIAL
 TONE_SECOND_DIAL TONE_RINGBACK TONE_OUT_OF_SERVICE
MEDIA_ACTIVITY</mediaEventsFilter><msgHeader><transactionID>txID001</transactionID></msgHeader><providerData><url>http://10.1.1.1:8090/cisco_xcc</url></providerData></RequestXccRegister></soapenv:Body></soapenv:Envelope>
23:25:09: //WSAPI/INFRA/27/0/wsapi_https_recv:
23:25:09: //WSAPI/INFRA/27/0/txID001/wsapi_ph_request_msg_handle:
23:25:09: //WSAPI/INFRA/27/0/txID001: prov_type 0 msg_type 6 prov_state 1
23:25:09: //WSAPI/INFRA/wsapi_create_common_msg:
23:25:09: //WSAPI/INFRA/wsapi_create_common_msg: Exit
23:25:09: //WSAPI/INFRA/27/0/txID001/wsapi_send_outbound_response:
23:25:09: wsapi_dump_msg: type 8
23:25:09: transactionID txID001
23:25:09: registrationID 50674FC:XCC:myapp:9
23:25:09: ResponseXccRegister:
23:25:09: providerStatus 1
23:25:09: //WSAPI/INFRA/27/0/txID001/wsapi_send_outbound_response: Exit
23:25:09: wsapi_send_ResponseRegister:mem_mgr_mempool_free: mem_refcnt(3CA18B8)=0 - mempool
 cleanup
23:25:09: //WSAPI/INFRA/27/0/txID001/wsapi_https_recv: Exit
23:25:09: wsapi_https_data_write: <?xml version="1.0" encoding="UTF-8"?><SOAP:Envelope
xmlns:SOAP="http://www.w3.org/2003/05/soap-envelope"><SOAP:Body><ResponseXccRegister
xmlns="http://www.cisco.com/schema/cisco_xcc/v1_0"><msgHeader><transactionID>txID001</transactionID><registrationID>50674FC:XCC:myapp:9</registrationID></msgHeader><providerStatus>IN_SERVICE</providerStatus></ResponseXccRegister></SOAP:Body></SOAP:Envelope>
23:25:09: //WSAPI/INFRA/wsapi_https_post_action: Exit
```

The following is a partial debug log from the **debug wsapi xcc all** command for a call..

```
Router# debug wsapi xcc all

23:27:20: //WSAPI/XCC/check_xccp_active:177:
23:27:20: //WSAPI/XCC/provider_base_get_state:248:
23:27:20: //WSAPI/XCC/provider_base_get_registration_count:212:
23:27:20: //WSAPI/XCC/check_xccp_active:177:
23:27:20: //WSAPI/XCC/provider_base_get_state:248:
23:27:20: //WSAPI/XCC/provider_base_get_registration_count:212:
23:27:20: //WSAPI/XCC/xccp_sessStore_call_add:271:
23:27:20: //WSAPI/XCC/xccp_sessStore_get_db:145:
23:27:20: //WSAPI/XCC/xccp_session_call_add:353: xcc session successfully added
23:27:20: //WSAPI/XCC/xccp_sessStore_call_add:285: xcc call successfully added
23:27:20: //WSAPI/XCC/check_xccp_active:177:
23:27:20: //WSAPI/XCC/provider_base_get_state:248:
23:27:20: //WSAPI/XCC/provider_base_get_registration_count:212:
23:27:20: //WSAPI/XCC/xccp_create_outbound_msg_space:677:
23:27:20: //WSAPI/XCC/xccp_sessStore_get_callData:225:
23:27:20: //WSAPI/XCC/xccp_sessStore_get_db:145:
23:27:20: //WSAPI/XCC/xccp_session_get_callData:445:
23:27:20: //WSAPI/XCC/check_xccp_active:177:
23:27:20: //WSAPI/XCC/provider_base_get_state:248:
23:27:20: //WSAPI/XCC/provider_base_get_registration_count:212:
23:27:20: //WSAPI/XCC/xccp_notify_events:434:
23:27:20: //WSAPI/XCC/xccp_queue_events:304:
23:27:20: //WSAPI/XCC/provider_base_event_new:335:
23:27:20: //WSAPI/UNKNOWN/event_base_new:267:
23:27:20: //WSAPI/XCC: magic [0xBABE] state[EVENT_STATE_ACTIVE] owner [0x1148C178] evSize[56]
```

```
 debFlag[3] evHdlr[0x894D834] evHdlFree[0x894DB00]
23:27:20: //WSAPI/UNKNOWN/event_base_new:292: event base new succ
23:27:20: //WSAPI/XCC/provider_base_event_new:360: provider base eventNew success
23:27:20: //WSAPI/XCC/provider_base_add_ev_to_q:393:
23:27:20: //WSAPI/XCC/check_xccp_active:177:
23:27:20: //WSAPI/XCC/provider_base_get_state:248:
23:27:20: //WSAPI/XCC/provider_base_get_registration_count:212:
23:27:20: //WSAPI/XCC/xccp_create_outbound_msg_space:677:
23:27:20: //WSAPI/XCC/xccp_sessStore_get_callData:225:
23:27:20: //WSAPI/XCC/xccp_sessStore_get_db:145:
23:27:20: //WSAPI/XCC/xccp_session_get_callData:445:
23:27:20: //WSAPI/XCC/check_xccp_active:177:
23:27:20: //WSAPI/XCC/provider_base_get_state:248:
23:27:20: //WSAPI/XCC/provider_base_get_registration_count:212:
23:27:20: //WSAPI/XCC/xccp_solicit_events:359:
23:27:20: //WSAPI/XCC/xccp_queue_events:304:
23:27:20: //WSAPI/XCC/provider_base_event_new:335:
23:27:20: //WSAPI/UNKNOWN/event_base_new:267:
23:27:20: //WSAPI/XCC: magic [0xBABE] state[EVENT_STATE_ACTIVE] owner [0x1148C178] evSize[56]
 debFlag[3] evHdlr[0x894D834] evHdlFree[0x894DB00]
23:27:20: //WSAPI/UNKNOWN/event_base_new:292: event base new succ
23:27:20: //WSAPI/XCC/provider_base_event_new:360: provider base eventNew success
23:27:20: //WSAPI/XCC/provider_base_add_ev_to_q:393:
23:27:20: //WSAPI/XCC/provider_base_process_events:444:
23:27:20: //WSAPI/XCC/xccp_handle_events:153:
23:27:20: //WSAPI/INFRA/wsapi_send_outbound_message:
23:27:20: //WSAPI/INFRA/wsapi_send_outbound_message_by_provider_info:
23:27:20: //WSAPI/XCC/wsapi_xcc_encode_outbound_msg:
23:27:20: //WSAPI/XCC/wsapi_xcc_encode_outbound_msg: Exit
23:27:20: //WSAPI/INFRA/0/1527/50875A4:319:out_url http://sj22lab-as2:8090/xcc
23:27:20: wsapi_send_outbound_message_by_provider_info: <?xml version="1.0"
encoding="UTF-8"?><SOAP:Envelope
xmlns:SOAP="http://www.w3.org/2003/05/soap-envelope"><SOAP:Body><NotifyXccConnectionData
[illegible]
23:27:20: //WSAPI/INFRA/0/1527/50875A4:319/wsapi_send_outbound_message_by_provider_info:
Exit
.
.
.
```

debug x25

To display information about all X.25 traffic or a specific X.25 service class, use the **debug x25**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug x25 [**only**| **cmns**| **xot**] [**events**| **all**] **[dump]**

no debug x25 [**only**| **cmns**] [**events**| **all**] **[dump]**

Syntax Description

only	(Optional) Displays information about X.25 services only.
cmns	(Optional) Displays information about CMNS services only.
xot	(Optional) Displays information about XOT services only.
events	(Optional) Displays all traffic except Data and Receiver Ready (RR) packets.
all	(Optional) Displays all traffic. This is the default.
dump	(Optional) Displays the encoded packet contents in hexadecimal and ASCII formats.

Command Default

All traffic is displayed.

Command Modes

Privileged EXEC

Command History

Release	Modification
10.0	This command was introduced.
12.0(5)T	For Domain Name System (DNS)-based X.25 routing, additional functionality was added to the **debug x25 events** command to describe the events that occur while the X.25 address is being resolved to an IP address using a DNS server. The **debug domain** command can be used along with **debug x25 events** to observe the whole DNS-based X.25 routing data flow.
12.0(7)T	For the X.25 Closed User Groups (CUGs) feature, functionality was added to the **debug x25 events** command to describe events that occur during CUG activity.

Release	Modification
12.2(8)T	The **debug x25 events** command was enhanced to display events specific to Record Boundary Preservation protocol.
12.3(2)T	The **dump** keyword was added.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelin

Caution The X.25 debug commands can generate large amounts of debugging output. If logging of debug output to the router console is enabled (the default condition), this output may fill the console buffer, preventing the router from processing packets until the contents of the console buffer have been printed.

The **debug x25**, **debug x25 interface**, **debug x25 vc**, and **debug x25 xot** commands all generate the same basic output. The **debug x25 interface**, **debug x25 vc**, and **debug x25 xot** commands narrow the scope of the **debug x25**output to specific virtual circuits or types of traffic.

This command is particularly useful for diagnosing problems encountered when placing calls. The **debug x25 all** output includes data, control messages, and flow control packets for all virtual circuits of the router.

All **debug x25** commands can take either the **events** or the **all** keyword. The keyword **all** is the default and causes all packets meeting the other debug criteria to be reported. The keyword **events** omits reports of any Data or RR flow control packets; the normal flow of data and RR packets is commonly large and less interesting to the user, so event reporting can significantly decrease the processor load induced by debug reporting.

Use the **dump** keyword to display the entire contents, including user data, of X.25 packets. The encoded X.25 packet contents are displayed after the standard packet description. The output includes the offset into the packet and the display of the data in both hexadecimal and ASCII formats.

Caution The X.25 packet information that is reported by using the **dump** keyword may contain sensitive data; for example, clear-text account identities and passwords. The network access policies and router configuration should be controlled appropriately to address this risk.

Examples

The following is sample output from the **debug x25**command, displaying output concerning the functions X.25 restart, call setup, data exchange, and clear:

```
Router# debug x25
Serial0: X.25 I R/Inactive Restart (5) 8 lci 0
  Cause 7, Diag 0 (Network operational/No additional information)
Serial0: X.25 O R3 Restart Confirm (3) 8 lci 0
Serial0: X.25 I P1 Call (15) 8 lci 1
From(6): 170091 To(6): 170090
   Facilities: (0)
   Call User Data (4): 0xCC000000 (ip)
Serial0: X.25 O P3 Call Confirm (3) 8 lci 1
Serial0: X.25 I D1 Data (103) 8 lci 1 PS 0 PR 0
Serial0: X.25 O D1 Data (103) 8 lci 1 PS 0 PR 1
Serial0: X.25 I P4 Clear (5) 8 lci 1
  Cause 9, Diag 122 (Out of order/Maintenance action)
Serial0: X.25 O P7 Clear Confirm (3) 8 lci 1
```

Examples

The following example of the **debug x25**command with the **events** keyword shows output related to the DNS-Based X.25 Routing feature. It shows messages concerning access to the DNS server. In the example, nine alternate addresses for one XOT path are entered into the DNS server database. All nine addresses are returned to the host cache of the router by the DNS server. However, only six addresses will be used during the XOT switch attempt because this is the limit that XOT allows.

```
Router# debug x25 events
00:18:25:Serial1:X.25 I R1 Call (11) 8 lci 1024
00:18:25: From (0): To (4):444
00:18:25: Facilities:(0)
00:18:25: Call User Data (4):0x01000000 (pad)
00:18:25:X.25 host name sent for DNS lookup is "444"
00:18:26:%3-TRUNCATE_ALT_XOT_DNS_DEST:Truncating excess XOT addresses (3)
returned by DNS
00:18:26:DNS got X.25 host mapping for "444" via network
00:18:32:[10.1.1.8 (pending)]:XOT open failed (Connection timed out; remote host not
responding)
00:18:38:[10.1.1.7 (pending)]:XOT open failed (Connection timed out; remote host not
responding)
00:18:44:[10.1.1.6 (pending)]:XOT open failed (Connection timed out; remote host not
responding)
00:18:50:[10.1.1.5 (pending)]:XOT open failed (Connection timed out; remote host not
responding)
00:18:56:[10.1.1.4 (pending)]:XOT open failed (Connection timed out; remote host not
responding)
00:20:04:[10.1.1.3,1998/10.1.1.3,11007]:XOT O P2 Call (17) 8 lci 1
00:20:04: From (0): To (4):444
00:20:04: Facilities:(6)
00:20:04:  Packet sizes:128 128
00:20:04:  Window sizes:2 2
00:20:04: Call User Data (4):0x01000000 (pad)
00:20:04:[10.1.1.3,1998/10.1.1.3,11007]:XOT I P2 Call Confirm (11) 8 lci 1
00:20:04: From (0): To (0):
00:20:04: Facilities:(6)
00:20:04:  Packet sizes:128 128
00:20:04:  Window sizes:2 2
00:20:04:Serial1:X.25 O R1 Call Confirm (5) 8 lci 1024
00:20:04: From (0): To (0):
00:20:04: Facilities:(0)
```

Examples

The following examples show output for the **x25 debug**command with the **events** keyword when record boundary preservation (RBP) has been configured using the **x25 map rbp local** command.

The following display shows establishment of connection:

```
X25 RBP:Incoming connection for port 9999 from 10.0.155.30 port 11001
Serial0/1:X.25 O R1 Call (10) 8 lci 64
  From (5):13133 To (5):12131
  Facilities:(0)
Serial0/1:X.25 I R1 Call Confirm (3) 8 lci 64
```

The following display shows that the X.25 call was cleared by the X.25 host:

```
Serial0/1:X.25 I R1 Clear (5) 8 lci 64
  Cause 0, Diag 122 (DTE originated/Maintenance action)
X25 RBP:X.25 circuit cleared
Serial0/1:X.25 O R1 Clear Confirm (3) 8 lci 64
```

The following display shows that the TCP session has terminated:

```
[10.0.155.30,11000/10.0.155.33,9999]:TCP receive error, End of data transfer
X25 RBP:End of data transfer
Serial0/1:X.25 O R1 Clear (5) 8 lci 64
```

```
Cause 9, Diag 122 (Out of order/Maintenance action)
Serial0/1:X.25 I R1 Clear Confirm (3) 8 lci 64
```

The following examples show output of the **x25 debug**command with the **events** keyword when RBP has been configured using the **x25 pvc rbp local** command.

The following display shows data on the permanent virtual circuit (PVC) before the TCP session has been established:

```
X25 RBP:Data on unconnected PVC
Serial1/0:X.25 O D1 Reset (5) 8 lci 1
  Cause 0, Diag 113 (DTE originated/Remote network problem)
Serial1/0:X.25 I D2 Reset Confirm (3) 8 lci 1
```

The following display shows establishment of connection:

```
X25 RBP:Incoming connection for port 9998 from 2.30.0.30 port 11002
Serial1/0:X.25 O D1 Reset (5) 8 lci 1
  Cause 0, Diag 0 (DTE originated/No additional information)
Serial1/0:X.25 I D2 Reset Confirm (3) 8 lci 1
```

The following display shows termination of connection when the X.25 PVC was reset:

```
Serial1/0:X.25 I D1 Reset (5) 8 lci 1
  Cause 15, Diag 122 (Network operational (PVC)/Maintenance action)
X25 RBP:Reset packet received
Serial1/0:X.25 O D3 Reset Confirm (3) 8 lci 1
```

The following display shows that the TCP session has terminated:

```
[2.30.0.30,11003/2.30.0.33,9998]:TCP receive error, End of data transfer
X25 RBP:End of data transfer
Serial1/0:X.25 O D1 Reset (5) 8 lci 1
  Cause 0, Diag 113 (DTE originated/Remote network problem)
Serial1/0:X.25 I D2 Reset Confirm (3) 8 lci 1
```

The following examples show output of the **x25 debug**command with the **events** keyword when RBP has been configured using the **x25 map rbp remote** command.

The following display shows that the X.25 call was cleared:

```
Serial0/1:X.25 I R1 Clear (5) 8 lci 1024
  Cause 0, Diag 122 (DTE originated/Maintenance action)
X25 RBP:X.25 circuit cleared
Serial0/1:X.25 O R1 Clear Confirm (3) 8 lci 1024
```

The following display shows that the X.25 call was reset:

```
Serial0/1:X.25 I D1 Reset (5) 8 lci 1024
  Cause 0, Diag 122 (DTE originated/Maintenance action)
X25 RBP:Reset packet received
Serial0/1:X.25 O R1 Clear (5) 8 lci 1024
  Cause 9, Diag 122 (Out of order/Maintenance action)
Serial0/1:X.25 I R1 Clear Confirm (3) 8 lci 1024
```

The following examples show output of the **x25 debug**command with the **events** keyword when RBP has been configured using the **x25 pvc rbp remote** command.

The following display shows that the X.25 PVC has been reset:

```
Serial0/0:X.25 I D1 Reset (5) 8 lci 1
  Cause 0, Diag 122 (DTE originated/Maintenance action)
X25 RBP:Reset packet received
Serial0/0:X.25 O D2 Reset Confirm (3) 8 lci 1
```

The following display shows that the connection was terminated when the X.25 interface was restarted:

```
Serial0/0:X.25 I R1 Restart (5) 8 lci 0
  Cause 0, Diag 122 (DTE originated/Maintenance action)
```

```
X25 RBP:X.25 PVC inactive
Serial0/0:X.25 O R2 Restart Confirm (3) 8 lci 0
Serial0/0:X.25 O D1 Reset (5) 8 lci 1
  Cause 1, Diag 113 (Out of order (PVC)/Remote network problem)
Serial0/0:X.25 I D3 Reset Confirm (3) 8 lci 1
```

Examples

The following is sample output for the **debug x25 dump** command. The encoded X.25 packet contents are displayed after the standard packet description. The output includes the offset into the packet and the display of the data in both hexadecimal and ASCII formats.

```
Router# debug x25 dump
Serial1: X.25 O R/Inactive Restart (5) 8 lci 0
   Cause 0, Diag 0 (DTE originated/No additional information)
       0: 1000FB00 00                          ..{..
Serial1: X.25 I R2 Restart (5) 8 lci 0
   Cause 7, Diag 0 (Network operational/No additional information)
       0:                            1000FB               ..{
       3: 0700                                 ..
Serial1: X.25 I R1 Call (13) 8 lci 1
   From (4): 2501 To (4): 2502
   Facilities: (0)
   Call User Data (4): 0xCC000000 (ip)
       0:            10010B 44250225 0100CC00       ...D%.%..L.
      11: 0000                                 ..
Serial1: X.25 O R1 Call Confirm (3) 8 lci 1
       0: 10010F                               ...
Serial1: X.25 I D1 Data (103) 8 lci 1 PS 0 PR 0
       0:            100100 45000064 00000000       ...E..d....
      11: FF01A764 0A190001 0A190002 0800CBFB  ..'d..........K{
      27: 0B1E22CA 00000000 00028464 ABCDABCD  .."J.......d+M+M
      43: ABCDABCD ABCDABCD ABCDABCD ABCDABCD  +M+M+M+M+M+M+M+M
      59: ABCDABCD ABCDABCD ABCDABCD ABCDABCD  +M+M+M+M+M+M+M+M
      75: ABCDABCD ABCDABCD ABCDABCD ABCDABCD  +M+M+M+M+M+M+M+M
      91: ABCDABCD ABCDABCD ABCDABCD           +M+M+M+M+M+M
Serial1: X.25 O D1 Data (103) 8 lci 1 PS 0 PR 1
       0:            100120 45000064 00000000       .. E..d....
      11: FF01A764 0A190002 0A190001 0000D3FB  ..'d..........S{
      27: 0B1E22CA 00000000 00028464 ABCDABCD  .."J.......d+M+M
      43: ABCDABCD ABCDABCD ABCDABCD ABCDABCD  +M+M+M+M+M+M+M+M
      59: ABCDABCD ABCDABCD ABCDABCD ABCDABCD  +M+M+M+M+M+M+M+M
      75: ABCDABCD ABCDABCD ABCDABCD ABCDABCD  +M+M+M+M+M+M+M+M
      91: ABCDABCD ABCDABCD ABCDABCD           +M+M+M+M+M+M
 Serial1: X.25 I R1 Clear (5) 8 lci 1
   Cause 9, Diag 122 (Out of order/Maintenance action)
       0:            100113 097A                    ....z
 Serial1: X.25 O R1 Clear Confirm (3) 8 lci 1
       0:                   100117                  .
```

The table below describes significant fields shown in the displays.

Table 91: debug x25 Field Descriptions

Field	Description
Serial0	Interface on which the X.25 event occurred.
X.25	Type of event this message describes.
I	Letter indicating whether the X.25 packet was input (I) or output (O) through the interface.

Field	Description
R3	State of the service or virtual circuit (VC). Possible values follow: R/Inactive--Packet layer awaiting link layer service R1--Packet layer ready R2--Data terminal equipment (DTE) restart request R3--DCE restart indication P/Inactive--VC awaiting packet layer service P1--Idle P2--DTE waiting for DCE to connect CALL P3--DCE waiting for DTE to accept CALL P4--Data transfer P5--CALL collision P6--DTE clear request P7--DCE clear indication D/Inactive--VC awaiting setup D1--Flow control ready D2--DTE reset request D3--DCE reset indication Refer to Annex B of the *ITU-T Recommendation X.25* for more information on these states.

Field	Description
Restart	The type of X.25 packet. Possible values follow: R Events • Restart • Restart Confirm • Diagnostic P Events • Call • Call Confirm • Clear • Clear Confirm D Events • Reset • Reset Confirm D1 Events • Data • Receiver Not Ready (RNR) • RR (Receiver Ready) • Interrupt • Interrupt Confirm XOT Overhead • PVC Setup Refer to RFC 1613 *Cisco Systems X.25 over TCP (XOT)* for information about the XOT PVC Setup packet type.
(5)	Number of bytes in the packet.
8	Modulo of the virtual circuit. Possible values are 8 and 128.
lci 0	VC number. Refer to Annex A of the *ITU-T Recommendation X.25* for information on VC assignment.

Field	Description
Cause 7	Code indicating the event that triggered the packet. The Cause field can appear only in entries for Clear, Reset, and Restart packets. Possible values for the Cause field can vary, depending on the type of packet. Refer to the appendix "X.25 Cause and Diagnostic Codes" for an explanation of these codes.
Diag 0	Code providing an additional hint of what, if anything, went wrong. The Diag field can appear only in entries for Clear, Diagnostic (as "error 0"), Reset, and Restart packets. Refer to the appendix "X.25 Cause and Diagnostic Codes" for an explanation of these codes.
(Network operational/ No additional information)	The standard explanations of the Cause and Diagnostic codes (*cause*/*diag*).
From (6):170091	Source address. (6) indicates the number of digits in the address that follows. The source address is part of the address block that may be encoded in Call Setup packets.
To (6): 170090	Destination address. (6) indicates the number of digits in the address that follows. The destination address is part of the address block that may be encoded in Call Setup packets.
Facilities:(0)	Indicates that a facilities block is encoded and that it consists of 0 bytes. A breakdown of the encoded facilities (if any) follows.
Call User Data (4):	Indicates that the Call User Data (CUD) field is present and consists of 4 bytes.
0xCC000000 (ip)	Protocol identifier (PID). This subfield of the CUD field is presented in the output as a hexadecimal string followed by the name of the protocol (in this case, IP) that the string represents. Any bytes following the PID are designated "user data" and may be used by an application separately from the PID.

Related Commands

Command	Description
debug x25 interface	Displays information about a specific X.25 or CMNS context or virtual circuit.

Command	Description
debug x25 vc	Displays information about traffic for all virtual circuits that use a given number.
debug x25 xot	Displays information about traffic to or from a specific XOT host.

debug x25 annexg

To display information about Annex G (X.25 over Frame Relay) events, use the **debug x25 annexg**command. To disable debugging output, use the **no** form of this command.

debug x25 annexg

no debug x25 annexg

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0 T	This command was introduced.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

It is generally recommended that the **debug x25 annexg** command be used only when specifically requested by Cisco TAC to obtain information about a problem with an Annex G configuration. The messages displayed by the **debug x25 annexg** command are meant to aid in the diagnosing of internal errors.

Caution The X.25 debug commands can generate large amounts of debugging output. If logging of debug output to the router console is enabled (the default condition), this output may fill the console buffer, preventing the router from processing packets until the contents of the console buffer have been printed.

Examples

The following shows sample output from the **debug x25 annexg** command for a Frame Relay data-link connection identifier (DLCI) configured for Annex G operation:

```
Router# debug x25 annexg

Jul 31 05:23:20.316:annexg_process_events:DLCI 18 attached to interface Serial2/0:0 is
ACTIVE
Jul 31 05:23:20.316:annexg_ctxt_create:Creating X.25 context over Serial2/0:0 (DLCI:18 using
 X.25 profile:OMC), type 10, len 2, addr 00 12
Jul 31 05:23:20.316:annexg_create_lower_layer:Se2/0:0 DLCI 18, payload 1606, overhead 2
Jul 31 05:23:20.320:annexg_restart_tx:sending pak to Serial2/0:0
Jul 31 05:23:23.320:annexg_restart_tx:sending pak to Serial2/0:0
```

The table below describes significant fields shown in the display.

Table 92: debug x25 annexg Field Descriptions

Field	Description
payload	Amount of buffer space available per message before adding Frame Relay and device-specific headers.
overhead	The length of the Frame Relay header and any device-specific header that may be needed.

Related Commands

Command	Description
debug x25	Displays information about all X.25 traffic or a specific X.25 service class.
debug x25 interface	Displays information about specific X.25, Annex G or CMN contexts or virtual circuits that occur on the identified interface.
debug x25 vc	Displays information about traffic for all virtual circuits that have a given number.

debug x25 aodi

To display information about an interface running PPP over an X.25 session, use the **debug x25 aodi**command. To disable debugging output, use the **no** form of this command.

debug x25 aodi

no debug x25 aodi

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
10.0	This command was introduced.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

Use the **debug x25 aodi** command to display interface PPP events running over an X.25 session and to debug X.25 connections between a client and server configured for Always On/Dynamic ISDN (AO/DI).

Examples

The following examples show the normal sequence of events for both the AO/DI client and the server sides:

Examples

```
Router# debug x25 aodi
PPP-X25: Virtual-Access1: Initiating AODI call request
PPP-X25: Bringing UP X.25 AODI VC
PPP-X25: AODI Client Call Confirm Event Received
PPP-X25: Cloning interface for AODI is Di1
PPP-X25: Queuing AODI Client Map Event
PPP-X25: Event:AODI Client Map
PPP-X25: Created interface Vi2 for AODI service
PPP-X25: Attaching primary link Vi2 to Di1
PPP-X25: Cloning Vi2 for AODI service using Di1
PPP-X25: Vi2: Setting the PPP call direction as OUT
PPP-X25: Vi2: Setting vectors for RFC1598 operation on BRI3/0:0 VC 0
PPP-X25: Vi2: Setting the interface default bandwidth to 10 Kbps
PPP-X25: Virtual-Access2: Initiating AODI call request
PPP-X25: Bringing UP X.25 AODI VC
PPP-X25: AODI Client Call Confirm Event Received
```

Examples

```
Router# debug x25 aodi
PPP-X25: AODI Call Request Event Received
PPP-X25: Event:AODI Incoming Call Request
PPP-X25: Created interface Vi1 for AODI service
PPP-X25: Attaching primary link Vi1 to Di1
PPP-X25: Cloning Vi1 for AODI service using Di1
```

```
PPP-X25: Vi1: Setting vectors for RFC1598 operation on BRI3/0:0 VC 1
PPP-X25: Vi1: Setting the interface default bandwidth to 10 Kbps
PPP-X25: Binding X.25 VC 1 on BRI3/0:0 to Vi1
```

debug x25 interface

To display information about the specific X.25, Annex G or Connection Mode Network Service (CMN) contexts or virtual circuits that occur on the identified interface, use the **debug x25 interface**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug x25 interface {*serial-interface*| *cmns-interface* [**mac** *mac-address*]} [**vc** *number*] [**events**| **all**] **[dump]**

no debug x25 interface {*serial-interface*| *cmns-interface* [**mac** *mac-address*]} [**vc** *number*] [**events**| **all**] **[dump]**

Syntax Description

serial-interface	Serial interface number that is configured for X.25 or Annex G service.
cmns-interface	Interface supporting CMNS traffic and, if specified, the MAC address of a remote host. The interface type can be Ethernet, Token Ring, or FDDI.
mac *mac-address*	(Optional) MAC address of the CMNS interface and remote host.
vc *number*	(Optional) Virtual circuit number. Range is from 1 to 4095.
events	(Optional) Displays all traffic except Data and Receiver Ready (RR) packets.
all	(Optional) Displays all traffic. This is the default.
dump	(Optional) Displays the encoded packet contents in hexadecimal and ASCII formats.

Command Default

All traffic is displayed.

Command Modes

Privileged EXEC

Command History

Release	Modification
10.0	This command was introduced.
12.3(2)T	The **dump** keyword was added.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelin

Caution The X.25 debug commands can generate large amounts of debugging output. If logging of debug output to the router console is enabled (the default condition), this output may fill the console buffer, preventing the router from processing packets until the contents of the console buffer have been printed.

The **debug x25**, **debug x25 interface**, **debug x25 vc**, and **debug x25 xot** commands all generate the same basic output. The **debug x25 interface**, **debug x25 vc**, and **debug x25 xot** commands narrow the scope of the **debug x25** output to specific virtual circuits or types of traffic.

The **debug x25 interface** command is useful for diagnosing problems encountered with a single X.25 or CMNS host or virtual circuit.

The keyword **all** is the default and causes all packets meeting the other debug criteria to be reported. The keyword **events** omits reports of any Data or RR flow control packets; the normal flow of data and RR packets is commonly large and less interesting to the user, so event reporting can significantly decrease the processor load induced by debug reporting.

Use the **dump** keyword to display the entire contents, including user data, of X.25 packets. The encoded X.25 packet contents are displayed after the standard packet description. The output includes the offset into the packet and the display of the data in both hexadecimal and ASCII formats.

Caution The X.25 packet information that is reported by using the **dump** keyword may contain sensitive data; for example, clear-text account identities and passwords. The network access policies and router configuration should be controlled appropriately to address this risk.

Examples

The following is sample output from the **debug x25 interface**command:

```
Router# debug x25 interface serial 0
X.25 packet debugging is on
X.25 packet debugging is restricted to interface serial0
Serial0: X.25 I R/Inactive Restart (5) 8 lci 0
  Cause 7, Diag 0 (Network operational/No additional information)
Serial0: X.25 O R3 Restart Confirm (3) 8 lci 0
Serial0: X.25 I P1 Call (15) 8 lci 1
From(6): 170091 To(6): 170090
   Facilities: (0)
   Call User Data (4): 0xCC000000 (ip)
Serial0: X.25 O P3 Call Confirm (3) 8 lci 1
Serial0: X.25 I D1 Data (103) 8 lci 1 PS 0 PR 0
Serial0: X.25 O D1 Data (103) 8 lci 1 PS 0 PR 1
Serial0: X.25 I P4 Clear (5) 8 lci 1
  Cause 9, Diag 122 (Out of order/Maintenance action)
Serial0: X.25 O P7 Clear Confirm (3) 8 lci 1
```

The table below describes the significant fields shown in the display.

Table 93: debug x25 interface Field Descriptions

Field	Description
Serial0	Interface on which the X.25 event occurred.
X.25	Type of event this message describes.

Field	Description
I	Letter indicating whether the X.25 packet was input (I) or output (O) through the interface.
R3	State of the service or virtual circuit (VC). Possible values follow: R/Inactive--Packet layer awaiting link layer service R1--Packet layer ready R2--Data terminal equipment (DTE) restart request R3--DCE restart indication P/Inactive--VC awaiting packet layer service P1--Idle P2--DTE waiting for DCE to connect CALL P3--DCE waiting for DTE to accept CALL P4--Data transfer P5--CALL collision P6--DTE clear request P7--DCE clear indication D/Inactive--VC awaiting setup D1--Flow control ready D2--DTE reset request D3--DCE reset indication Refer to Annex B of the *ITU-T Recommendation X.25* for more information on these states.

Field	Description
Restart	The type of X.25 packet. Possible values follow: R Events • Restart • Restart Confirm • Diagnostic P Events • Call • Call Confirm • Clear • Clear Confirm D Events • Reset • Reset Confirm D1 Events • Data • Receiver Not Ready (RNR) • RR (Receiver Ready) • Interrupt • Interrupt Confirm XOT Overhead • PVC Setup
(5)	Number of bytes in the packet.
8	Modulo of the virtual circuit. Possible values are 8 and 128.
lci 0	VC number. Refer to Annex A of the *ITU-T Recommendation X.25* for information on VC assignment.

Field	Description
Cause 7	Code indicating the event that triggered the packet. The Cause field can appear only in entries for Clear, Reset, and Restart packets. Possible values for the Cause field can vary, depending on the type of packet. Refer to the appendix "X.25 Cause and Diagnostic Codes" for an explanation of these codes.
Diag 0	Code providing an additional hint of what, if anything, went wrong. The Diag field can appear only in entries for Clear, Diagnostic (as "error 0"), Reset, and Restart packets. Refer to the appendix "X.25 Cause and Diagnostic Codes" for an explanation of these codes.
(Network operational/ No additional information)	The standard explanations of the Cause and Diagnostic codes (*cause/diag*).
From (6):170091	Source address. (6) indicates the number of digits in the address that follows. The source address is part of the address block that may be encoded in Call Setup packets.
To (6): 170090	Destination address. (6) indicates the number of digits in the address that follows. The destination address is part of the address block that may be encoded in Call Setup packets.
Facilities:(0)	Indicates that a facilities block is encoded and that it consists of 0 bytes. A breakdown of the encoded facilities (if any) follows.
Call User Data (4):	Indicates that the Call User Data (CUD) field is present and consists of 4 bytes.
0xCC000000 (ip)	Protocol identifier (PID). This subfield of the CUD field is presented in the output as a hexadecimal string followed by the name of the protocol (in this case, IP) that the string represents. Any bytes following the PID are designated "user data" and may be used by an application separately from the PID.

Related Commands

Command	Description
debug x25	Displays information about all X.25 traffic or a specific X.25 service class.

Command	Description
debug x25 vc	Displays information about traffic for all virtual circuits that use a given number.
debug x25 xot	Displays information about traffic to or from a specific XOT host.

debug x25 vc

To display information about traffic for all virtual circuits that have a given number, use the **debug x25 vc**command. To disable debugging output, use the **no**form of this command.

debug x25 vc *number* [**events**| **all**] **[dump]**

no debug x25 vc *number* [**events**| **all**] **[dump]**

Syntax Description

number	Virtual circuit number. Range is from 1 to 4095.
events	(Optional) Displays all traffic except Data and Receiver Ready (RR) packets.
all	(Optional) Displays all traffic. This is the default.
dump	(Optional) Displays the encoded packet contents in hexadecimal and ASCII formats.

Command Default

All traffic is displayed.

Command Modes

Privileged EXEC

Command History

Release	Modification
10.0	This command was introduced.
12.3(2)T	The **dump** keyword was introduced.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelin

Caution The X.25 debug commands can generate large amounts of debugging output. If logging of debug output to the router console is enabled (the default condition), this output may fill the console buffer, preventing the router from processing packets until the contents of the console buffer have been printed.

The **debug x25, debug x25 interface**, **debug x25 vc**, and **debug x25 xot** commands all generate the same basic output. The **debug x25 interface**, **debug x25 vc**, and **debug x25 xot** commands narrow the scope of the **debug x25** output to specific virtual circuits or types of traffic.

Because no interface is specified by the **debug x25 vc** command, traffic on any virtual circuit that has the specified number is reported.

Virtual circuit (VC) zero (vc 0) cannot be specified. It is used for X.25 service messages, such as RESTART packets, not virtual circuit traffic. Service messages can be monitored only when no virtual circuit filter is used.

Use the **dump** keyword to display the entire contents, including user data, of X.25 packets. The encoded X.25 packet contents are displayed after the standard packet description. The output includes the offset into the packet and the display of the data in both hexadecimal and ASCII formats.

Caution The X.25 packet information that is reported by using the **dump** keyword may contain sensitive data; for example, clear-text account identities and passwords. The network access policies and router configuration should be controlled appropriately to address this risk.

Examples

The following shows sample output from the **debug x25 vc** command:

```
Router# debug x25 vc 1 events
X.25 special event debugging is on
X.25 debug output restricted to VC number 1
Router# show debug
X.25 (filtered for VC 1):
  X.25 special event debugging is on
*Jun 18 20:22:29.735 UTC:Serial0:X.25 O R1 Call (13) 8 lci 1
*Jun 18 20:22:29.735 UTC:  From (4):2501 To (4):2502
*Jun 18 20:22:29.735 UTC:  Facilities:(0)
*Jun 18 20:22:29.735 UTC:  Call User Data (4):0xCC000000 (ip)
*Jun 18 20:22:29.739 UTC:Serial0:X.25 I R1 Call Confirm (3) 8 lci 1
*Jun 18 20:22:36.651 UTC:Serial0:X.25 O R1 Clear (5) 8 lci 1
*Jun 18 20:22:36.651 UTC:  Cause 9, Diag 122 (Out of order/Maintenance action)
*Jun 18 20:22:36.655 UTC:Serial0:X.25 I R1 Clear Confirm (3) 8 lci 1
```

The table below describes significant fields shown in the display.

Table 94: debug x25 vc Field Descriptions

Field	Description
Serial0	Interface on which the X.25 event occurred.
X.25	Type of event this message describes.
O	Letter indicating whether the X.25 packet was input (I) or output (O) through the interface.

Field	Description
R1	State of the service or virtual circuit (VC). Possible values follow: R/Inactive--Packet layer awaiting link layer service R1--Packet layer ready R2--Data terminal equipment (DTE) restart request R3--DCE restart indication P/Inactive--VC awaiting packet layer service P1--Idle P2--DTE waiting for DCE to connect CALL P3--DCE waiting for DTE to accept CALL P4--Data transfer P5--CALL collision P6--DTE clear request P7--DCE clear indication D/Inactive--VC awaiting setup D1--Flow control ready D2--DTE reset request D3--DCE reset indication Refer to Annex B of the *ITU-T Recommendation X.25* for more information on these states.

Field	Description
Call	The type of X.25 packet. Possible values follow: R Events • Restart • Restart Confirm • Diagnostic P Events • Call • Call Confirm • Clear • Clear Confirm D Events • Reset • Reset Confirm D1 Events • Data • Receiver Not Ready (RNR) • RR (Receiver Ready) • Interrupt • Interrupt Confirm XOT Overhead • PVC Setup
(5)	Number of bytes in the packet.
8	Modulo of the virtual circuit. Possible values are 8 and 128.
lci 0	VC number. Refer to Annex A of the *ITU-T Recommendation X.25* for information on VC assignment.
From (4):2501	Source address. (4) indicates the number of digits in the address that follows. The source address is part of the address block that may be encoded in Call Setup packets.

Field	Description
To (4): 2502	Destination address. (4) indicates the number of digits in the address that follows. The destination address is part of the address block that may be encoded in Call Setup packets.
Facilities:(0)	Indicates that 0 bytes are being used to encode facilities.
Call User Data (4):	Indicates that the Call User Data (CUD) field is present and consists of 4 bytes.
0xCC000000 (ip)	Protocol identifier (PID). This subfield of the CUD field is presented in the output as a hexadecimal string followed by the name of the protocol (in this case, IP) that the string represents. Any bytes following the PID are designated “user data” and may be used by an application separately from the PID.
Cause 7	Code indicating the event that triggered the packet. The Cause field can appear only in entries for Clear, Reset, and Restart packets. Possible values for the Cause field can vary, depending on the type of packet. Refer to the appendix “X.25 Cause and Diagnostic Codes” for an explanation of these codes.
Diag 0	Code providing an additional hint of what, if anything, went wrong. The Diag field can appear only in entries for Clear, Diagnostic (as “error 0”), Reset, and Restart packets. Refer to the appendix “X.25 Cause and Diagnostic Codes” for an explanation of these codes.
(Network operational/ No additional information)	The standard explanations of the Cause and Diagnostic codes (*cause/diag*).

Related Commands

Command	Description
debug x25	Displays information about all X.25 traffic or a specific X.25 service class.
debug x25 interface	Displays information about a specific X.25 or CMNS context or virtual circuit.
debug x25 xot	Displays information about traffic to or from a specific XOT host.

debug x25 xot

To display information about traffic to or from a specific X.25 over TCP (XOT) host, use the **debug x25 xot** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug x25 xot [**remote** *ip-address* [**port** *number*]] [**local** *ip-address* [**port** *number*]] [**events**| **all**] **[dump]**

no debug x25 xot [**remote** *ip-address* [**port** *number*]] [**local** *ip-address* [**port** *number*]] [**events**| **all**] **[dump]**

Syntax Description

remote *ip-address* [**port** *number*]	(Optional) Remote IP address and, optionally, a port number. Range is from 1 to 65535.
local *ip-address* [**port** *number*]	(Optional) Local host IP address and, optionally, a port number. Range is from 1 to 65535.
events	(Optional) Displays all traffic except Data and Receiver Ready (RR) packets.
all	(Optional) Displays all traffic. This is the default.
dump	(Optional) Displays the encoded packet contents in hexadecimal and ASCII formats.

Command Default

All traffic is displayed.

Command Modes

Privileged EXEC

Command History

Release	Modification
10.0	This command was introduced.
12.3(2)T	The **dump** keyword was added.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelin

Caution The X.25 debug commands can generate large amounts of debugging output. If logging of debug output to the router console is enabled (the default condition), this output may fill the console buffer, preventing the router from processing packets until the contents of the console buffer have been printed.

The **debug x25**, **debug x25 interface**, **debug x25 vc**, and **debug x25 xot** commands all generate the same basic output. The **debug x25 interface**, **debug x25 vc**, and **debug x25 xot** commands narrow the scope of the **debug x25** output to specific virtual circuits or types of traffic.

The **debug x25 xot** output allows you to restrict the debug output reporting to XOT traffic for one or both hosts or host/port combinations. Because each XOT virtual circuit uses a unique TCP connection, an XOT debug request that specifies both host addresses and ports will report traffic only for that virtual circuit. Also, you can restrict reporting to sessions initiated by the local or remote router by specifying 1998 for the remote or local port. (XOT connections are received on port 1998.)

Use the **dump** keyword to display the entire contents, including user data, of X.25 packets. The encoded X.25 packet contents are displayed after the standard packet description. The output includes the offset into the packet and the display of the data in both hexadecimal and ASCII formats.

Caution The X.25 packet information that is reported by using the **dump** keyword may contain sensitive data; for example, clear-text account identities and passwords. The network access policies and router configuration should be controlled appropriately to address this risk.

Examples

The following shows sample output from the **debug x25 xot** command:

```
Router# debug x25 xot
X.25 packet debugging is on
X.25 debug output restricted to protocol XOT
Router# show debug
X.25 (filtered for XOT):
  X.25 packet debugging is on
*Jun 18 20:32:34.699 UTC:[10.0.155.71,11001/10.0.155.70,1998]:XOT I P/Inactive Call (19) 8
 lci 1
*Jun 18 20:32:34.699 UTC:  From (4):2501 To (4):2502
*Jun 18 20:32:34.699 UTC:  Facilities:(6)
*Jun 18 20:32:34.699 UTC:    Packet sizes:128 128
*Jun 18 20:32:34.699 UTC:    Window sizes:2 2
*Jun 18 20:32:34.699 UTC:  Call User Data (4):0xCC000000 (ip)
*Jun 18 20:32:34.707 UTC:[10.0.155.71,11001/10.0.155.70,1998]:XOT O P3 Call Confirm (11) 8
 lci 1
*Jun 18 20:32:34.707 UTC:  From (0): To (0):
*Jun 18 20:32:34.707 UTC:  Facilities:(6)
*Jun 18 20:32:34.707 UTC:    Packet sizes:128 128
*Jun 18 20:32:34.707 UTC:    Window sizes:2 2
*Jun 18 20:32:34.715 UTC:[10.0.155.71,11001/10.0.155.70,1998]:XOT I D1 Data (103) 8 lci 1
PS 0 PR 0
*Jun 18 20:32:34.723 UTC:[10.0.155.71,11001/10.0.155.70,1998]:XOT O D1 Data (103) 8 lci 1
PS 0 PR 1
*Jun 18 20:32:34.731 UTC:[10.0.155.71,11001/10.0.155.70,1998]:XOT I D1 Data (103) 8 lci 1
PS 1 PR 1
*Jun 18 20:32:34.739 UTC:[10.0.155.71,11001/10.0.155.70,1998]:XOT O D1 Data (103) 8 lci 1
PS 1 PR 2
*Jun 18 20:32:34.747 UTC:[10.0.155.71,11001/10.0.155.70,1998]:XOT I D1 Data (103) 8 lci 1
PS 2 PR 2
*Jun 18 20:32:34.755 UTC:[10.0.155.71,11001/10.0.155.70,1998]:XOT O D1 Data (103) 8 lci 1
PS 2 PR 3
*Jun 18 20:32:34.763 UTC:[10.0.155.71,11001/10.0.155.70,1998]:XOT I D1 Data (103) 8 lci 1
PS 3 PR 3
*Jun 18 20:32:34.771 UTC:[10.0.155.71,11001/10.0.155.70,1998]:XOT O D1 Data (103) 8 lci 1
PS 3 PR 4
*Jun 18 20:32:34.779 UTC:[10.0.155.71,11001/10.0.155.70,1998]:XOT I D1 Data (103) 8 lci 1
PS 4 PR 4
*Jun 18 20:32:34.787 UTC:[10.0.155.71,11001/10.0.155.70,1998]:XOT O D1 Data (103) 8 lci 1
PS 4 PR 5
```

The table below describes the significant fields shown in the display.

Table 95: debug x25 xot Field Descriptions

Field	Description
[10.0.155.71,11001/10.0.155.70,1998]	TCP connection identified by the remote IP address, remote TCP port/local IP address, local TCP port. An XOT connection is always placed to port ID 1998, so a remote port ID of 1998 implies that the router initiated the TCP connection, whereas a local port ID of 1998 implies that the router received the TCP connection.
XOT	Type of event this message describes.
I	Letter indicating whether the X.25 packet was input (I) or output (O) through the interface.
P/Inactive	State of the service or virtual circuit (VC). Possible values follow: R/Inactive--Packet layer awaiting link layer service R1--Packet layer ready R2--Data terminal equipment (DTE) restart request R3--DCE restart indication P/Inactive--VC awaiting packet layer service P1--Idle P2--DTE waiting for DCE to connect CALL P3--DCE waiting for DTE to accept CALL P4--Data transfer P5--CALL collision P6--DTE clear request P7--DCE clear indication D/Inactive--VC awaiting setup D1--Flow control ready D2--DTE reset request D3--DCE reset indication Refer to Annex B of the *ITU-T Recommendation X.25* for more information on these states.

Field	Description
Call	The type of X.25 packet. Possible values follow: R Events • Restart • Restart Confirm • Diagnostic P Events • Call • Call Confirm • Clear • Clear Confirm D Events • Reset • Reset Confirm D1 Events • Data • Receiver Not Ready (RNR) • RR (Receiver Ready) • Interrupt • Interrupt Confirm XOT Overhead • PVC Setup
(19)	Number of bytes in the packet.
8	Modulo of the virtual circuit. Possible values are 8 and 128.
lci 1	VC number. Refer to Annex A of the *ITU-T Recommendation X.25* for information on VC assignment.
From (4):2501	Source address. (4) indicates the number of digits in the address that follows. The source address is part of the address block that may be encoded in Call Setup packets.

Field	Description
To (4): 2502	Destination address. (4) indicates the number of digits in the address that follows. The destination address is part of the address block that may be encoded in Call Setup packets.
Facilities:(6)	Indicates that a facilities block is encoded and that it consists of 6 bytes. A breakdown of the encoded facilities follows.
Packet sizes	Encoded packet size facility settings.
Window sizes	Encoded window size facility settings.
Call User Data (4):	Indicates that the Call User Data (CUD) field is present and consists of 4 bytes.
0xCC000000 (ip)	Protocol identifier (PID). This subfield of the CUD field is presented in the output as a hexadecimal string followed by the name of the protocol (in this case, IP) that the string represents. Any bytes following the PID are designated “user data” and may be used by an application separately from the PID.
Cause 7	Code indicating the event that triggered the packet. The Cause field can appear only in entries for Clear, Reset, and Restart packets. Possible values for the Cause field can vary, depending on the type of packet. Refer to the appendix “X.25 Cause and Diagnostic Codes” for an explanation of these codes.
Diag 0	Code providing an additional hint of what, if anything, went wrong. The Diag field can appear only in entries for Clear, Diagnostic (as “error 0”), Reset, and Restart packets. Refer to the appendix “X.25 Cause and Diagnostic Codes” for an explanation of these codes.
(Network operational/ No additional information)	The standard explanations of the Cause and Diagnostic codes (*cause/diag*).

Related Commands

Command	Description
debug x25	Displays information about all X.25 traffic or a specific X.25 service class.

Command	Description
debug x25 interface	Displays information about a specific X.25 or CMNS context or virtual circuit.
debug x25 vc	Displays information about traffic for all virtual circuits that use a given number.

debug x28

To monitor error information and X.28 connection activity, use the **debug x28**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug x28

no debug x28

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Examples

The following is sample output while the packet assembler/disassembler (PAD) initiates an X.28 outgoing call:

```
Router# debug x28
X28 MODE debugging is on
Router# x28
*
03:30:43: X.28 mode session started
03:30:43: X28 escape is exit
03:30:43: Speed for console & vty lines :9600
*call 123456
COM
03:39:04: address ="123456", cud="[none]" 03:39:04: Setting X.3 Parameters for this call...1:1
 2:1 3:126 4:0 5:1 6:2 7:2 8:0 9:0 10:0 11:14 12:1 13:0 14:0 15:0 16:127 17:24 18:18 19:2
20:0 21:0 22:0
Router> exit
CLR CONF
*
*03:40:50: Session ended
* exit
Router#
*03:40:51: Exiting X.28 mode
```

debug xcctsp all

To debug External Call Control Telephony Service Provider (TSP) information, use the **debug xcctsp all**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug xcctsp all

no debug xcctsp all

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(5)T	This command was introduced.
12.0(7)T	Support for this command was extended to the Cisco uBR924 cable modem.

Examples

See the following examples to turn on and off external call control debugging:

```
AS5300-TGW# debug xcctsp all
External call control all debugging is on
AS5300-TGW# no debug xcctsp all
External call control all debugging is off
AS5300-TGW#
```

Related Commands

Command	Description
debug xcctsp error	Enables debugging on external call control errors.
debug xcctsp session	Enables debugging on external call control sessions.

debug xcctsp error

To debug External Call Control Telephony Service Provider (TSP) error information, use the **debug xcctsp error**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug xcctsp error

no debug xcctsp error

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(5)T	This command was introduced.
12.0(7)T	Support for this command was integrated on the Cisco uBR924 cable modem.

Examples

See the following examples to turn on and off error-level debugging:

```
AS5300-TGW# debug xcctsp error
External call control error debugging is on
AS5300-TGW# no debug xcctsp error
External call control error debugging is off
```

Related Commands

Command	Description
debug xcctsp all	Enables debugging on all external call control levels.
debug xcctsp session	Enables debugging on external call control sessions.

debug xcctsp session

To debug External Call Control Telephony Service Provider (TSP) session information, use the **debug xcctsp session**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug xcctsp session

no debug xcctsp session

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(5)T	This command was introduced.
12.0(7)T	Support for this command was integrated on the Cisco uBR924 cable modem.

Examples

See the following examples to turn on and off session-level debugging:

```
AS5300-TGW# debug xcctsp session
External call control session debugging is on
AS5300-TGW# no debug xcctsp session
External call control session debugging is off
AS5300-TGW#
```

Related Commands

Command	Description
debug xcctsp all	Enables debugging on external call control levels.
debug xcctsp error	Enables debugging on external call control errors.

debug xconnect

To debug a problem related to the xconnect configuration, use the **debug xconnect** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug xconnect [rib] {**error| event| checkpoint**}

no debug xconnect [rib] {**error| event| checkpoint**}

Syntax Description

rib	(Optional) Displays events related to the pseudowire Routing Information Base (RIB).
error	Displays errors related to an xconnect configuration.
event	Displays events related to an xconnect configuration processing.
checkpoint	Displays the autodiscovered pseudowire information that is checkpointed to the standby Route Processor (RP).

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.0(23)S	This command was introduced.
12.3(2)T	This command was integrated into Cisco IOS Release 12.3(2)T.
12.2(25)S	This command was integrated into Cisco IOS Release 12.2(25)S.
12.2(27)SBC	This command was integrated into Cisco IOS Release 12.2(27)SBC.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
15.1(1)S	This command was integrated into Cisco IOS Release 15.1(1)S. The **rib** and **checkpoint** keywords were added.
Cisco IOS XE Release 3.8S	This command was integrated into Cisco IOS XE Release 3.8S.

Usage Guidelines

Use this command to display debugging information about xconnect sessions.

Examples

The following is sample output from the **debug xconnect** command for an xconnect session on an Ethernet interface:

```
Router# debug xconnect event
00:01:16: XC AUTH [Et2/1, 5]: Event: start xconnect authorization, state changed from IDLE
 to AUTHORIZING
00:01:16: XC AUTH [Et2/1, 5]: Event: found xconnect authorization, state changed from
AUTHORIZING to DONE
00:01:16: XC AUTH [Et2/1, 5]: Event: free xconnect authorization request, state changed
from DONE to END
```

Related Commands

Command	Description
debug acircuit	Displays events and failures related to attachment circuits.
debug vpdn	Displays errors and events relating to L2TP configuration and the surrounding Layer 2 tunneling infrastructure.

debug xcsp

To display the debugging messages for the External Control Service Provider (XCSP) subsystem, use the **debug xcsp** command in privilegedEXEC mode. To disable debugging output, use the **no** form of this command.

debug xcsp {**all**| **cot**| **event**}

no debug xcsp {**all**| **cot**| **event**}

Syntax Description

all	Provides debug information about XCSP events and continuity testing (COT).
cot	Provides debug information about XCSP and COT. The **cot** keyword is not used with the NAS Package for Media Gateway Control Protocol (MGCP) feature.
event	Provides debug information about XCSP events.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(2)XB	This command was introduced.
12.2(11)T	The command was integrated into Cisco IOS Release 12.2(11)T for the Cisco AS5850.

Usage Guidelines

This command is used with the Network Access Server Package for MGCP. The XCSP subsystem is not configured directly, but information about it may be useful in troubleshooting. The **debug xcsp** command is used to display the exchange of signaling information between the MGCP protocol stack and end applications such as call switching module (CSM) or dialer.

The **cot** keyword is not used with the Network Access Server Package for MGCP feature.

Examples

The following shows sample output from the **debug xcsp all** command and keyword and the **debug xcsp event** command and keyword:

```
Router# debug xcsp all
xcsp all debugging is on
```

```
Router# debug xcsp event
xcsp events debugging is on
01:49:14:xcsp_call_msg:Event  Call Indication , channel state =  Idle  for
slot port channel 7
c5400# 0 23
01:49:14:xcsp_process_sig_fsm:state/event  Idle / Call Indication
01:49:14:xcsp_incall:
01:49:14:xcsp_incall CONNECT_IND:cdn=3000 cgn=1000
01:49:14:xcsp:START guard TIMER
01:49:14:xcsp_fsm:slot 7 port 0 chan 23 oldstate =  Idle  newstate= Connection
in progress mgcpapp_process_mgcp_msg PROCESSED NAS PACKAGE EVENT
01:49:14:Received message on XCSP_CDAPI
01:49:14:process_cdapi_msg :slot/port/channel 7/0/23
01:49:14: process_cdapi_msg:new  slot/port/channel 7/0/23
01:49:14:
c5400#Received CONN_RESP:callid=0x7016
01:49:14:process_cdapi:Event CONN_RESP, channel state = 8 for slot port
channel 7 0 23
01:49:14:xcsp_process_sig_fsm:state/event  Connection in progress / In Call
accept
 mgcpapp_xcsp_alert:
 mgcpapp_xcsp_get_chan_cb -Found - Channel state  Connection in progress
200 58  Alert
I:630AED90
<---:Ack send SUCCESSFUL
01:49:14:xcsp_fsm:slot 7 p
c5400#ort 0 chan 23 oldstate =  Connection in progress  newstate= Connection in
progress
01:49:14:Received message on XCSP_CDAPI
01:49:14:process_cdapi_msg :slot/port/channel 7/0/23
01:49:14: process_cdapi_msg:new  slot/port/channel 7/0/23
01:49:14: Received CALL_CONN:callid=0x7016
01:49:14:process_cdapi:Event CONN_, channel state = 8 for slot port channel 7
0 23
01:49:14:xcsp_process_sig_fsm:state/event  Connection in progress / in call
connect
 mgcpapp_xcsp_connect:
 mgcpapp_xc
c5400#sp_get_chan_cb -Found - Channel state In Use
01:49:14:STOP TIMER
01:49:14:xcsp_fsm:slot 7 port 0 chan 23 oldstate =  Connection in progress
newstate=In Use
c5400#
01:50:23:Received message on XCSP_CDAPI
01:50:23:process_cdapi_msg :slot/port/channel 7/0/23
01:50:23: process_cdapi_msg:new  slot/port/channel 7/0/23
01:50:23: Received CALL_DISC_REQ:callid=0x7016
01:50:23:process_cdapi:Event DISC_CONN_REQ, channel state = 7 for slot port
channel 7 0 23
01:50:23:xcsp_process_sig_fsm:state/event In Use / release Request
 mgcpapp_xcsp_disconnect
 mgcpapp_xcsp_get_chan_cb -Fou
c5400#nd - Channel state In Use
01:50:23:send_mgcp_msg, MGCP Packet sent --->
01:50:23:RSIP 1 *@c5400 MGCP 1.0
RM:restart
DLCX 4 S7/DS1-0/23 MGCP 1.0
C:3
I:630AED90
E:801 /NAS User request
01:50:23:xcsp_fsm:slot 7 port 0 chan 23 oldstate = In Use  newstate=Out
Release in progress
 xcsp_restart Serial7/0:22 vc = 22
 xcsp_restart Put idb Serial7/0:22 in down state
01:50:23:MGCP Packet received -
200 4 bye
 Data call ack received callp=0x62AEEA70mgcpapp_xcsp
c5400#_ack_recv:mgcpapp_xcsp_get_chan_cb -Found - Channel state Out Release in
progress
mgcpapp_xcsp_ack_recv ACK 200 rcvd:transaction id = 4 endpt=S7/DS1-0/23
01:50:23:xcsp_call_msg:Event  Release confirm , channel state = Out Release in
progress for slot port channel 7 0 23
01:50:23:xcsp_process_sig_fsm:state/event Out Release in progress/ Release
```

```
confirm
01:50:23:STOP TIMER
01:50:23:xcsp_fsm:slot 7 port 0 chan 23 oldstate = Out Release in progress
newstate= Idle
```

Related Commands

Command	Description
show vrm vdevices	Displays the status of a router port under the control of the XCSP subsystem.
show xcsp slot	Displays the status of a router slot under the control of the XCSP subsystem.

debug xdsl application

To monitor the xDSL if the digital subscriber line (DSL) does not come up, use the debug xdsl application command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug xdsl application

no debug xdsl application

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(4)XD	This command was introduced on Cisco 2600 series and Cisco 3700 series routers.
12.3(4)XG	Support was added for the Cisco 1700 series routers.
12.3(7)T	This command was integrated into Cisco IOS Release 12.3(7)T on Cisco 2600 series, Cisco 3631, and Cisco 3700 series routers.
12.3(11)T	Support was added for the Cisco 2800 and Cisco 3800 series routers.
12.3(14)T	Support was added for the Cisco 1800 series routers.

Usage Guidelines

The debug xdsl application command details what occurs during the Cisco IOS SHDSL process events and signal-to-noise ratio sampling of the SHDSL chip. This information can be used more for software debugging in analyzing the internal events.

Examples

The following is sample output from the **debug xdsl application** command:

```
Router# debug xdsl application

xDSL application debugging is on
Router#
```

The following lines show that the application is starting on the router and waiting for a response:

```
00:47:40: DSL 0/0 process_get_wakeup
00:47:41: DSL 0/0 process_get_wakeup
00:47:42: DSL 0/0 process_get_wakeup
00:47:43: DSL 0/0 process_get_wakeup
00:47:44: DSL 0/0 process_get_wakeup
00:47:45: DSL 0/0 process_get_wakeup
00:47:46: DSL 0/0 process_get_wakeup
00:47:47: DSL 0/0 process_get_wakeup
00:47:48: DSL 0/0 process_get_wakeup
```

```
00:47:49: DSL 0/0 process_get_wakeup
00:47:49: DSL 0/0 process_get_wakeup
```

The following lines show that the controller link comes up:

```
00:47:49: DSL 0/0 xdsl_background_process: XDSL link up boolean event received
00:47:49:  DSL 0/0 controller Link up! line rate: 1600 Kbps
```

The following lines show that the DSL controller comes up:

```
00:47:49: DSL 0/0 xdsl_controller_reset: cdb-state=up
00:47:49: %CONTROLLER-5-UPDOWN: Controller DSL 0/0, changed state to up
00:47:49:  Dslsar data rate 1600
00:47:49: DSL 0/0 TipRing 1, Xmit_Power Val 75, xmit_power 7.5
00:47:49: DSL 0/0 Mode 2, BW 1600, power_base_value 135, power_backoff 6
00:47:50: DSL 0/0 process_get_wakeup
00:47:51: DSL 0/0 process_get_wakeup
00:47:52: DSL 0/0 process_get_wakeup
00:47:53: DSL 0/0 process_get_wakeup
00:47:54: DSL 0/0 process_get_wakeup
00:47:55: DSL 0/0 process_get_wakeup
00:47:56: DSL 0/0 process_get_wakeup
```

The following lines show signal-to-noise ratio sampling:

```
00:47:56: DSL 0/0   SNR Sampling: 42 dB
00:47:57: DSL 0/0 process_get_wakeup
00:47:57: DSL 0/0   SNR Sampling: 41 dB
00:47:58: DSL 0/0 process_get_wakeup
00:47:58: DSL 0/0   SNR Sampling: 40 dB
00:47:59: DSL 0/0 process_get_wakeup
00:47:59: DSL 0/0   SNR Sampling: 40 dB
00:48:00: DSL 0/0 process_get_wakeup
00:48:00: DSL 0/0   SNR Sampling: 39 dB
00:48:01: DSL 0/0 process_get_wakeup
00:48:01: DSL 0/0   SNR Sampling: 39 dB
00:48:02: DSL 0/0 process_get_wakeup
00:48:02: DSL 0/0   SNR Sampling: 38 dB
00:48:03: DSL 0/0 process_get_wakeup
00:48:03: DSL 0/0   SNR Sampling: 38 dB
00:48:04: DSL 0/0 process_get_wakeup
00:48:04: DSL 0/0   SNR Sampling: 38 dB
00:48:05: DSL 0/0 process_get_wakeup
00:48:05: DSL 0/0   SNR Sampling: 37 dB
00:48:06: DSL 0/0 process_get_wakeup
00:48:06: DSL 0/0   SNR Sampling: 37 dB
00:48:07: DSL 0/0 process_get_wakeup
00:48:07: DSL 0/0   SNR Sampling: 36 dB
```

The following lines show that the link comes up:

```
00:48:07: %LINK-3-UPDOWN: Interface ATM0/0, changed state to up
00:48:08: DSL 0/0 process_get_wakeup
00:48:08: DSL 0/0   SNR Sampling: 36 dB
```

The following lines show that the line protocol comes up:

```
00:48:08: %LINEPROTO-5-UPDOWN: Line protocol on Interface ATM0/0, changed state to up
00:48:09: DSL 0/0 process_get_wakeup
00:48:09: DSL 0/0   SNR Sampling: 36 dB
00:48:10: DSL 0/0 process_get_wakeup
00:48:10: DSL 0/0   SNR Sampling: 36 dB
00:48:11: DSL 0/0 process_get_wakeup
00:48:11: DSL 0/0   SNR Sampling: 35 dB
00:48:12: DSL 0/0 process_get_wakeup
00:48:12: DSL 0/0   SNR Sampling: 36 dB
00:48:13: DSL 0/0 process_get_wakeup
00:48:13: DSL 0/0   SNR Sampling: 36 dB
00:48:14: DSL 0/0 process_get_wakeup
00:48:14: DSL 0/0   SNR Sampling: 36 dB
00:48:15: DSL 0/0 process_get_wakeup
00:48:15: DSL 0/0   SNR Sampling: 36 dB
00:48:16: DSL 0/0 process_get_wakeup
```

```
00:48:16: DSL 0/0   SNR Sampling: 36 dB
00:48:17: DSL 0/0 process_get_wakeup
00:48:17: DSL 0/0   SNR Sampling: 35 dB
00:48:18: DSL 0/0 process_get_wakeup
00:48:18: DSL 0/0   SNR Sampling: 35 dB
00:48:19: DSL 0/0 process_get_wakeup
```

Related Commands

Command	Description
debug xdsl driver	Monitors what is happening when downloading and installing the drivers.
debug xdsl eoc	Monitors what is in the embedded operations channel messages.
debug xdsl error	Monitors the errors of the xDSL process and firmware.

debug xdsl driver

To display what is happening when the drivers are downloaded and installed, use the **debug xdsl driver** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug xdsl driver

no debug xdsl driver

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(4)XD	This command was introduced on Cisco 2600 series and Cisco 3700 series routers.
12.3(4)XG	Support was added for Cisco 1700 series routers.
12.3(7)T	This command was integrated into Cisco IOS Release 12.3(7)T on Cisco 2600 series, Cisco 3631, and Cisco 3700 series routers.
12.3(11)T	Support was added for the Cisco 2800 and Cisco 3800 series routers.
12.3(14)T	Support was added for Cisco 1800 series routers.

Usage Guidelines

Use the **debug xdsl driver** command to monitor what is happening when downloading the firmware. This debugging command displays the Globespan DSL Driver details and provides framer interrupt information and line training failure information. This information can help you understand the problems faced while downloading the firmware, why the line went down, and so forth.

Examples

The following is sample output from the **debug xdsl driver**command:

```
Router# debug xdsl driver
xDSL driver debugging is on
```

The following lines show that the DSP interrupt download is running:

```
*Mar 12 08:01:04.772: DSL 0/2  dsp interrupt-download next block for line-0
*Mar 12 08:01:04.780: DSL 0/2 framer intr_status 0xC0
*Mar 12 08:01:05.072: DSL 0/2  dsp interrupt-download next block for line-0
*Mar 12 08:01:05.080: DSL 0/2 framer intr_status 0xC0
*Mar 12 08:01:06.484: DSL 0/2  dsp interrupt-download next block for line-0
*Mar 12 08:01:06.492: DSL 0/2 framer intr_status 0xC0
*Mar 12 08:01:08.092: DSL 0/2  dsp interrupt-download next block for line-0
*Mar 12 08:01:08.096: DSL 0/2 framer intr_status 0xC0
*Mar 12 08:01:19.180: DSL 0/2  dsp interrupt-download next block for line-0
```

```
*Mar 12 08:01:19.184: DSL 0/2 framer intr_status 0xC0
*Mar 12 08:01:19.480: DSL 0/2  dsp interrupt-download next block for line-0
*Mar 12 08:01:19.484: DSL 0/2 framer intr_status 0xC0
*Mar 12 08:01:19.680: DSL 0/2  dsp interrupt-download next block for line-0
```

The following lines show that the DSP interrupt has been disabled and that the framer interrupt has been enabled:

```
*Mar 12 08:01:19.680: DSL 0/2 DSP interrupt disabled
*Mar 12 08:01:19.680: DSL 0/2  Download completed for line-0
*Mar 12 08:01:19.680: DSL 0/2 Framer interrupt enabled
*Mar 12 08:01:19.680: DSL 0/2 framer intr_status 0xC0
*Mar 12 08:01:19.680:  DSL 0/2 controller Link up! line rate: 2304 Kbps
```

The following lines show that the digital subscriber line (DSL) controller has come up on slot 0 and port 2:

```
*Mar 12 08:01:19.680: %CONTROLLER-5-UPDOWN: Controller DSL 0/2, changed state to up
*Mar 12 08:01:19.680:  Dslsar data rate 2304
*Mar 12 08:01:22.528: %LINK-3-UPDOWN: Interface ATM0/2, changed state to up
*Mar 12 08:01:23.528: %LINEPROTO-5-UPDOWN: Line protocol on Interface ATM0/2, changed state
 to up
```

The following lines show that the framer interrupt status is running:

```
*Mar 12 08:01:23.812: DSL 0/2 framer intr_status 0xC4
*Mar 12 08:01:23.816: DSL 0/2 framer intr_status 0xC4
*Mar 12 08:01:23.904: DSL 0/2 framer intr_status 0xC1
*Mar 12 08:01:28.612: DSL 0/2 framer intr_status 0xC4
*Mar 12 08:01:28.616: DSL 0/2 framer intr_status 0xC4
*Mar 12 08:01:28.708: DSL 0/2 framer intr_status 0xC1
*Mar 12 08:01:28.804: DSL 0/2 framer intr_status 0xC1
*Mar 12 08:01:33.412: DSL 0/2 framer intr_status 0xC4
*Mar 12 08:01:33.420: DSL 0/2 framer intr_status 0xC4
*Mar 12 08:01:33.508: DSL 0/2 framer intr_status 0xC1
*Mar 12 08:01:33.604: DSL 0/2 framer intr_status 0xC1
*Mar 12 08:01:33.700: DSL 0/2 framer intr_status 0xC1
*Mar 12 08:01:38.212: DSL 0/2 framer intr_status 0xC4
*Mar 12 08:01:38.220: DSL 0/2 framer intr_status 0xC4
*Mar 12 08:01:38.308: DSL 0/2 framer intr_status 0xC1
```

Related Commands

Command	Description
debug xdsl application	Monitors the xDSL if the DSL does not come up.
debug xdsl eoc	Monitors what is in the embedded operations channel messages.
debug xdsl error	Monitors the errors of the xDSL process and firmware.

debug xdsl eoc

To display the flow of the embedded operations channel (EOC) messages received, processed, and transmitted, use the **debug xdsl eoc** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug xdsl eoc

no debug xdsl eoc

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(4)XD	This command was introduced on Cisco 2600 series and Cisco 3700 series routers.
12.3(4)XG	This command was integrated into the Cisco IOS Release 12.3(4)XG on the Cisco 1700 series routers.
12.3(7)T	This command was integrated into Cisco IOS Release 12.3(7)T on Cisco 2600 series, Cisco 3631, and Cisco 3700 series routers.
12.3(11)T	This command was implemented on Cisco 2800 series and Cisco 3800 series routers.
12.3(14)T	This command was implemented on Cisco 1800 series routers.

Usage Guidelines

Use the **debug xdsl eoc** command to review the contents of the embedded operations channel messages.

Examples

The following is sample output from the **debug xdsl eoc** command:

```
Router# debug xdsl eoc

xDSL EOC debugging is on
Router#
```

The following lines show the embedded operations channel message being received and copied to the buffer. The xdsl_background_process is performed. The data_transparency_remove is performed.

```
00:02:55:   Incoming EOC received
00:02:55:   Copy the EOC to buffer
00:02:55:   Incoming EOC received
00:02:55:   Copy the EOC to buffer
00:02:55:  End of EOC received, Notify task
00:02:55: xdsl_background_process:
```

```
00:02:55:  Rx EOC remove transparency:: 12 C  A  63
00:02:55: data_transparency_remove: Done, eoc packet size = 4
```

The following lines show that the packet of the embedded operations channel messages was received and verified as good. The data_transparency_add is performed.

```
00:02:55:   Good eoc packet received
00:02:55:  incoming request eocmsgid: 12
00:02:55:  Tx Converted EOC message:: 21 8C 0  28 0  0  0  0  0  0  0  1  1  713
00:02:55: data_transparency_add: eoc packet size - before 15, after 15
```

The following lines show another embedded operations channel message coming in and being copied to the buffer. The xdsl_background_process is run on this message as before.

```
00:02:55:  size of eoc status response :: 13
00:02:56:   Incoming EOC received
00:02:56:   Copy the EOC to buffer
00:02:56:   Incoming EOC received
00:02:56:   Copy the EOC to buffer
00:02:56:  End of EOC received, Notify task
00:02:56: xdsl_background_process:
00:02:56:  Rx EOC remove transparency:: 12 C  A  63
00:02:56: data_transparency_remove: Done, eoc packet size = 4
```

Related Commands

Command	Description
debug xdsl application	Displays status of the xDSL if the DSL does not activate as expected.
debug xdsl driver	Diaplays status when the drivers are downloaded and installed.
debug xdsl error	Displays the errors of the xDSL process and firmware.

debug xdsl error

To display the errors of xDSL process and firmware, use the **debug xdsl error** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug xdsl error

no debug xdsl error

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(4)XD	This command was introduced on Cisco 2600 series and Cisco 3700 series routers.
12.3(4)XG	Support was added for the Cisco 1700 series routers.
12.3(7)T	This command was integrated into Cisco IOS Release 12.3(7)T on Cisco 2600 series, Cisco 3631, and Cisco 3700 series routers.
12.3(11)T	Support was added for the Cisco 2800 and Cisco 3800 series routers.
12.3(14)T	Support was added for Cisco 1800 series routers.

Usage Guidelines

Use the **debug xdsl error** command to display the errors during driver initialization and any Globespan firmware API failures.

Examples

The following is sample output from the **debug xdsl error** command. When the debug is enabled, a message indicates that DSL error debugging is on.

```
Router# debug xdsl error
```

```
xDSL error debugging is on
Router#
```

Related Commands

Command	Description
debug xdsl application	Monitors the xDSL if the DSL does not come up.
debug xdsl driver	Monitors what is happening when downloading and installing the drivers.

Command	Description
debug xdsl eoc	Monitors what is in the embedded operations channel messages.

debug xmpp profile

To debug issues related to Cisco Open Plug-n-Play (PnP) agent infrastructure, use the **debug xmpp profile** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug xmpp [**initiator** | **listener**]

no debug xmpp [**initiator** | **listener**]

Syntax Description

initiator	Enables debugging of Extensible Messaging and Presence Protocol (XMPP) Extension Protocols (XEP) profile initiator.
listener	Enables debugging of XEP profile listener.

Command Default

Disabled

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
15.4(2)T	This command was introduced.
Cisco IOS XE Release 3.12S	This command was integrated into Cisco IOS XE Release 3.12S.
15.2(2)E	This command was integrated into Cisco IOS Release 15.2(2)E.

Examples

The following example shows how to debug infrastructure issues in a PnP agent:

```
Device> enable
Device# debug xmpp initiator
XEP profile initiator debugging is on
Device# debug xmpp listener
XEP profile listener debugging is on
```

Related Commands

Command	Description
debug pnp	Debugs traces in PnP agent.

debug zone

To display zone security event debugs, use the **debug zone**command in privileged EXEC mode. To disable the debugging messages, use the **no** form of this command.

debug zone security {**events**| **object-creation**| **object-deletion**}

no debug zone security {**events**| **object-creation**| **object-deletion**}

Syntax Description

security	Displays security events debug messages.
events	Displays zone security events debug messages.
object-creation	Displays zone security object creation debug messages.
object-deletion	Displays zone security object deletion debug messages.

Command Default

By default, debugging is not enabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.4(6)T	This command was introduced.

Examples

If the **debug zone security events**command is enabled and a zone event occurs, firewall generates debug messages. An event can be a zone or zone pair creation and deletion.

```
Router# show debug
zone:
Zone security Events debugging is on
Router# configure terminal
Router(config)# zone security public
Router(config-sec-zone)#
*Jan 29 05:04:52.967: ZONE_SEC:zone added
Router(config-sec-zone)# zone security private
Router(config-sec-zone)#
*Jan 29 05:05:02.999: ZONE_SEC:zone added
Router(config-sec-zone)# exit

Router(config)# zone-pair security pu2pr source public destination private

Router(config-sec-zone-pair)#
*Jan 29 05:05:37.575: ZONE_SEC:zone-pair added
```

```
*Jan 29 05:05:37.575: ZONE_SEC:allocating zone-pair
Router(config)# no zone-pair security pu2pr source public destination private
Router(config)#
*Jan 29 05:08:00.667: ZONE_SEC:zone-pair deleting...
Router(config)# no zone security public
Router(config)#
*Jan 29 05:08:12.135: ZONE_SEC:zone deleting..
Router(config)# no zone security private
Router(config)#
*Jan 29 05:08:18.243: ZONE_SEC:zone deleting..
```

If the **debug zone security object-creation** and the **debug zone security object-deletion** commands are enabled and when zones or zone pairs are created or deleted, firewall generates debug messages.

```
Router# show debugging

zone:
Zone security Object Creations debugging is on
Zone security Object Deletions debugging is on
Router# configure terminal
Router(config)# zone security public

Router(config-sec-zone)#
*Jan 29 05:09:28.207: ZONE_SEC: zone public created
Router(config-sec-zone)# exit
Router(config)# zone security private
Router(config-sec-zone)#
*Jan 29 05:09:50.831: ZONE_SEC: zone private created
Router(config-sec-zone)# exit
Router(config)# zone-pair security zp source public destination private
Router(config-sec-zone-pair)#
*Jan 29 05:10:22.063: ZONE_SEC: zone-pair zp created
Router(config-sec-zone-pair)# service-policy type inspect pmap
Router(config-sec-zone-pair)#
*Jan 29 05:10:36.787: ZONE_SEC: zone-pair FW_INT_REV_zp_3748291079 created
Router(config-sec-zone-pair)# no service-policy type inspect pmap
Router(config-sec-zone-pair)# exit
Router(config)# no zone-pair security zp source public destination private
Router(config)#
*Jan 29 05:11:04.043: ZONE_SEC: zone-pair zp deleted
Router(config)# no zone security public
Router(config)#
*Jan 29 05:11:10.875: ZONE_SEC: zone public deleted
Router(config)# no zone security private
Router(config)#
*Jan 29 05:11:16.931: ZONE_SEC: zone private deleted
Router(config)# end
Router#
```

Related Commands

Command	Description
zone security	Creates a security zone.
zone-pair security	Creates a zone pair.

show memory debug incremental

To display information about memory leaks after a starting time has been established, use the **show memory debug incremental** command in privileged EXEC mode.

show memory debug incremental {**allocations**| **leaks**[**lowmem**| **summary**]| **status**}

Syntax Description

allocations	Displays all memory blocks that were allocated after issuing the **set memory debug incremental starting-time** command.
leaks	Displays only memory that was leaked after issuing the **set memory debug incremental starting-time** command.
lowmem	(Optional) Forces the memory leak detector to work in low memory mode, making no memory allocations.
summary	(Optional) Reports summarized memory leaks based on allocator_pc and size of the memory block.
status	Displays all memory blocks that were allocated after issuing the **set memory debug incremental starting-time** command.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(7)T	This command was introduced.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.4T	The summary keyword was added.

Usage Guidelines

The **show memory debug incremental allocations** command displays all the memory blocks that were allocated after the **set memory debug incremental starting-time** command was entered. The displayed memory blocks are just memory allocations, they are not necessarily leaks.

The **show memory debug incremental leaks** command provides output similar to the **show memory debug leaks** command, except that it displays only memory that was leaked after the **set memory debug incremental starting-time** command was entered.

The **show memory debug incremental leaks lowmem** command forces memory leak detection to work in low memory mode. The amount of time taken for analysis is considerably greater than that of normal mode. The output for this command is similar to the **show memory debug leaks** command, except that it displays only memory that was leaked after the **set memory debug incremental starting-time** command was entered. You can use this command when you already know that normal mode memory leak detection will fail (perhaps by an unsuccessful previous attempt to invoke normal mode memory leak detection).

The **show memory debug incremental leaks summary** command displays a summarized report of the memory that was leaked after the **set memory debug incremental starting-time** command was entered, ordered by allocator process call address (Alloc_pc) and by memory block size.

The **show memory debug incremental status** command displays whether a starting point for incremental analysis has been set and the elapsed time since then.

Note All show memory debug commands must be used on customer networks only to diagnose the router for memory leaks when memory depletion is observed. These CLI's will have high CPU utilization and might result in time sensitive protocols to flap. These CLI's are recommended for customer use, only in the maintenance window when the router is not in a scaled condition.

Note All memory leak detection commands invoke normal mode memory leak detection, except when the low memory option is specifically invoked by use of the **lowmem** keyword. In normal mode, if memory leak detection determines that there is insufficient memory to proceed in normal mode, it will display an appropriate message and switch to low memory mode.

Examples

Examples

The following example shows output from the **show memory debug incremental**command when entered with the **allocations** keyword:

```
Router# show memory debug incremental allocations
Address    Size   Alloc_pc  PID  Name
62DA4E98      176 608CDC7C  44   CDP Protocol
62DA4F48       88 608CCCC8  44   CDP Protocol
62DA4FA0       88 606224A0  3    Exec
62DA4FF8       96 606224A0  3    Exec
635BF040       96 606224A0  3    Exec
63905E50      200 606A4DA4  69   Process Events
```

Examples

The following example shows output from the **show memory debug incremental** command when entered with the **leaks** and **summary** keywords:

```
Router# show memory debug incremental leaks summary
Adding blocks for GD...
                 PCI memory
Alloc PC      Size     Blocks      Bytes    What
                 I/O memory
Alloc PC      Size     Blocks      Bytes    What
                 Processor memory
Alloc PC      Size          Blocks       Bytes      What
0x60874198  0000000052  0000000001  0000000052    Exec
0x60874198  0000000060  0000000001  0000000060    Exec
0x60874198  0000000100  0000000001  0000000100    Exec
```

```
0x60874228  0000000052  0000000004  0000000208    Exec
0x60874228  0000000060  0000000002  0000000120    Exec
0x60874228  0000000100  0000000004  0000000400   Exec
```

Examples

The following example shows output from the **show memory debug incremental** command entered with the **status** keyword:

```
Router# show memory debug incremental status
Incremental debugging is enabled
Time elapsed since start of incremental debugging: 00:00:10
```

Related Commands

Command	Description
set memory debug incremental starting-time	Sets the current time as the starting time for incremental analysis.
show memory debug leaks	Displays detected memory leaks.

set memory debug incremental starting-time

To set the current time as the starting time for incremental analysis, use the **set memory debug incremental starting-time** command in privileged EXEC mode.

set memory debug incremental starting-time [none]

Syntax Description

none	(Optional) Resets the defined start time for incremental analysis.

Command Default

No default behavior or values.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(8)T1	This command was introduced.
12.2(25)S	This command was integrated into Cisco IOS Release 12.2(25)S.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

For incremental analysis, a starting point can be defined by using the **set memory debug incremental starting-time** command. When a starting time is set, only memory allocated after that starting time will be considered for reporting as leaks.

Examples

The following example shows the command used to set the starting time for incremental analysis to the time when the command was issued:

```
Router# set memory debug incremental starting-time
```

Related Commands

Command	Description
show memory debug incremental allocation	Displays all memory blocks that were allocated after the issue of the **set memory debug incremental starting-time** command.

Command	Description
show memory debug incremental leaks	Displays only memory that was leaked after the issue of the **set memory debug incremental starting-time** command.
show memory debug incremental leaks lowmem	Forces incremental memory leak detection to work in low memory mode. Displays only memory that was leaked after the issue of the **set memory debug incremental starting-time** command.
show memory debug incremental status	Displays if the starting point of incremental analysis has been defined and the time elapsed since then.
show memory debug leaks	Displays detected memory leaks.

show memory debug leaks

To display detected memory leaks, use the **show memory debug leaks** command in privileged EXEC mode. This command does not have a **no** form.

Cisco IOS software

show memory debug leaks [**chunks**| **largest**| **lowmem**| **summary**]

Cisco Catalyst 4500e Series Switches running IOS XE software

show memory debug leaks all [**detailed**| **totals**]

Syntax Description

all	Displays the information about leak block of the internal memory .
detailed	(Optional) Displays the detailed information about memory debug leak.
chunks	(Optional) Displays the memory leaks in chunks.
largest	(Optional) Displays the top ten leaking allocator_pcs based on size, and the total amount of memory they have leaked.
lowmem	(Optional) Forces the memory leak detector to work in low memory mode, making no memory allocations.
summary	(Optional) Reports summarized memory leaks based on allocator_pc and size of the memory block.
totals	(Optional) Displays summary report with the total number of each process.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS 12.3(8)T1	This command was introduced.
Cisco IOS 12.2(25)S	This command was integrated into Cisco IOS Release 12.2(25)S.
Cisco IOS 12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Release	Modification
Cisco IOS XE 3.1.0.SG	This command was introduced on the Cisco Catalyst 4500e Serfies Switches to display per-process memory leak amounts.
Cisco IOS 15.2(2)E	This command was integrated into Cisco IOS Release 15.2(2)E.

Usage Guidelines

If optional keywords are not specified, the **show memory debug leaks** command invokes normal mode memory leak detection and does not look for memory leaks in chunks.

The **show memory debug leaks chunks** command invokes normal mode memory leak detection and looks for leaks in chunks as well.

The **show memory debug leaks largest** command displays the top ten leaking allocator_pcs and the total amount of memory that they have leaked. Additionally, each time when this command is invoked, it remembers the report of the previous invocation and compares it with the report of the current invocation. If there are new entries in the current report, they are tagged as “inconclusive.” If the same entry appears in the report of the previous invocation and the report of the current invocation, the inconclusive tag is not added. It is beneficial to run memory leak detection more than once and to consider only the consistently reported leaks.

The **show memory debug leaks lowmem** command forces memory leak detection to work in low memory mode. The amount of time taken for analysis is considerably greater than that of normal mode. The output for this command is similar to the **show memory debug leaks** command. You can use this command when you know that the normal mode memory leak detection fails (perhaps by an unsuccessful previous attempt to invoke normal mode memory leak detection).

The **show memory debug leaks summary** command reports memory leaks based on allocator_pc and then on the size of the block.

The **show memory debug leaks all detailed** command provides the details of memory leaks for a particular process.

The **show memory debug leaks all totals** command provides the summary report with the total number of memory leaks of each running process.

Note All **show memory debug** commands must be used on customer networks only to diagnose the router for memory leaks when memory depletion is observed. These CLIs have high CPU utilization and might result in time sensitive protocols to flap. These CLIs are recommended for customer use, only in the maintenance window when the router is not in a scaled condition.

Note The command **show memory debug leak lowmem** is extremely CPU intensive and can result in CPUHOG/WATCHDOG crash. This command must be used only when the router has reached an unusable state due to memory exhaustion. Its use on high end platforms such as ISR and above can potentially crash the box. Use of this command outside of these limitations can cause a console hang of one hour in some cases. As an alternative, use the **show memory debug leak** command.

Examples

Examples

The following example shows output from the **show memory debug leaks** command:

```
Device# show memory debug leaks
Adding blocks for GD...
                 PCI memory
Address    Size   Alloc_pc  PID  Name
                 I/O memory
Address    Size   Alloc_pc  PID  Name
                 Processor memory
Address    Size   Alloc_pc  PID  Name
62DABD28       80 60616750  -2   Init
62DABD78       80 606167A0  -2   Init
62DCF240       88 605B7E70  -2   Init
62DCF298       96 605B7E98  -2   Init
62DCF2F8       88 605B7EB4  -2   Init
62DCF350       96 605B7EDC  -2   Init
63336C28      104 60C67D74  -2   Init
63370D58       96 60C656AC  -2   Init
633710A0      304 60C656AC  -2   Init
63B2BF68       96 60C659D4  -2   Init
63BA3FE0    32832 608D2848  104  Audit Process
63BB4020    32832 608D2FD8  104  Audit Process
```

The table below describes the significant fields shown in the display.

Table 96: show memory debug leaks Field Descriptions

Field	Description
Address	Hexadecimal address of the leaked block.
Size	Size of the leaked block (in bytes).
Alloc_pc	Address of the system call that allocated the block.
PID	The process identifier of the process that allocated the block.
Name	The name of the process that allocated the block.

Examples

The following example shows output from the **show memory debug leaks chunks** command:

```
Router# show memory debug leaks chunks
Adding blocks for GD...
                 PCI memory
Address    Size   Alloc_pc  PID  Name
Chunk Elements:
Address  Size  Parent   Name
                 I/O memory
Address    Size   Alloc_pc  PID  Name
Chunk Elements:
Address  Size  Parent   Name
                 Processor memory
Address    Size   Alloc_pc  PID  Name
62DABD28       80 60616750  -2   Init
62DABD78       80 606167A0  -2   Init
```

```
62DCF240       88 605B7E70  -2   Init
62DCF298       96 605B7E98  -2   Init
62DCF2F8       88 605B7EB4  -2   Init
62DCF350       96 605B7EDC  -2   Init
63336C28      104 60C67D74  -2   Init
63370D58       96 60C656AC  -2   Init
633710A0      304 60C656AC  -2   Init
63B2BF68       96 60C659D4  -2   Init
63BA3FE0    32832 608D2848  104  Audit Process
63BB4020    32832 608D2FD8  104  Audit Process
Chunk Elements:
Address  Size  Parent   Name
62D80DA8    16 62D7BFD0 (Managed Chunk )
62D80DB8    16 62D7BFD0 (Managed Chunk )
62D80DC8    16 62D7BFD0 (Managed Chunk )
62D80DD8    16 62D7BFD0 (Managed Chunk )
62D80DE8    16 62D7BFD0 (Managed Chunk )
62E8FD60   216 62E8F888 (IPC Message He)
```

The table below describes the significant fields shown in the display.

Note **show memory debug leaks chunks** command is a debug command and CPU intensive. Hence, the trace-backs is thrown on running the command as expected.

Table 97: show memory debug leaks chunks Field Descriptions

Field	Description
Address	Hexadecimal address of the leaked block.
Size	Size of the leaked block (in bytes).
Alloc_pc	Address of the system call that allocated the block.
PID	The process identifier of the process that allocated the block.
Name	The name of the process that allocated the block.
Size	(Chunk Elements) Size of the leaked element (bytes).
Parent	(Chunk Elements) Parent chunk of the leaked chunk.
Name	(Chunk Elements) The name of the leaked chunk.

Examples

The following example shows output from the **show memory debug leaks largest** command:

```
Router# show memory debug leaks largest
Adding blocks for GD...
                 PCI memory
Alloc_pc    total leak size
                 I/O memory
Alloc_pc    total leak size
                 Processor memory
Alloc_pc    total leak size
```

```
608D2848    32776     inconclusive
608D2FD8    32776     inconclusive
60C656AC    288       inconclusive
60C67D74    48        inconclusive
605B7E98    40        inconclusive
605B7EDC    40        inconclusive
60C659D4    40        inconclusive
605B7E70    32        inconclusive
605B7EB4    32        inconclusive
60616750    24        inconclusive
```

The following example shows output from the second invocation of the **show memory debug leaks largest** command:

```
Router# show memory debug leaks largest
Adding blocks for GD...
                 PCI memory
Alloc_pc    total leak size
                 I/O memory
Alloc_pc    total leak size
                 Processor memory
Alloc_pc    total leak size
608D2848    32776
608D2FD8    32776
60C656AC    288
60C67D74    48
605B7E98    40
605B7EDC    40
60C659D4    40
605B7E70    32
605B7EB4    32
60616750    24
```

Examples

The following example shows output from the **show memory debug leaks summary** command:

```
Router# show memory debug leaks summary
Adding blocks for GD...
                 PCI memory
Alloc PC        Size     Blocks      Bytes    What
                 I/O memory
Alloc PC        Size     Blocks      Bytes    What
                 Processor memory
Alloc PC        Size     Blocks      Bytes    What
0x605B7E70 0000000032 0000000001 0000000032    Init
0x605B7E98 0000000040 0000000001 0000000040    Init
0x605B7EB4 0000000032 0000000001 0000000032    Init
0x605B7EDC 0000000040 0000000001 0000000040    Init
0x60616750 0000000024 0000000001 0000000024    Init
0x606167A0 0000000024 0000000001 0000000024    Init
0x608D2848 0000032776 0000000001 0000032776    Audit Process
0x608D2FD8 0000032776 0000000001 0000032776    Audit Process
0x60C656AC 0000000040 0000000001 0000000040    Init
0x60C656AC 0000000248 0000000001 0000000248    Init
0x60C659D4 0000000040 0000000001 0000000040    Init
0x60C67D74 0000000048 0000000001 0000000048    Init
```

The table below describes the significant fields shown in the display.

Table 98: show memory debug leaks summary Field Descriptions

Field	Description
Alloc_pc	Address of the system call that allocated the block.
Size	Size of the leaked block.

Field	Description
Blocks	Number of blocks leaked.
Bytes	Total amount of memory leaked.
What	Name of the process that owns the block.

Examples

Examples

The following example shows output from the **show memory debug leaks all detailed** command:

```
Device# show memory debug leaks all detailed
Process PID : 4644    Process Name : platformmgr
Address   Size    Alloc PC                                             TID         Name
1FEA5E30  20     0000000000000000000000000000000000000000-0+1F5E51BC  4644       XOS_MEM_XDT
```

The table below describes the significant fields shown in the display.

Table 99: show memory debug leaks all detailed Field Descriptions

Field	Description
Address	Hexadecimal address of the leaked block.
Size	Size of the leaked block (in bytes).
Alloc_pc	Address of the system call that allocated the block.
PID	The process identifier of the process that allocated the block
TID	The Task identifier for a particular process identifier.
Name	Name of the process that owns the block.

Examples

The following example shows output from the **show memory debug leaks all totals** command:

```
Device# show memory debug leaks all totals
Process      Num Leak  Total Leak      Num Leak  Total Leak      Total Leak
Name         Mem Blks  Mem Blks(b)     Chunk El  Chunk Elem(b)   (bytes)

slproc       7         232             0         0               232
platformmgr  1         20              0         0               20
ns_oir_prox  8         240             0         0               240
profiled     13        1516            0         0               1516
obfld        5         1464            9         152             1616
consoled     13        1516            0         0               1516
csprovider   13        1552            0         0               1552
system_mgr_  13        1560            0         0               1560
plogd        7         1736            25        968             2704
```

```
psdprov      13        1532          0         0              1532
pdsd         16        1564          0         0              1564
gold_slave   9         376           0         0              376
osinfo-prov  13        1576          0         0              1576
oscore_p     13        1520          0         0              1520
netd         10        256           1         28             284
mem_mgmt     82        4620          1         40             4660
mgmte_tap    9         376           0         0              376
licensed     133       16052         95        2660           18712
```

The table below describes the significant fields shown in the display.

Table 100: show memory debug leaks all totals Field Descriptions

Field	Description
Process Name	Name of the leak process.
Num Leak Mem Blks	Number of memory blocks affected by leak for a particular process.
Total Leak Mem Blks(b)	Amount of memory affected by leaks in bytes for a particular process.
Num Leak Chunk El	Number of chunk blocks affected by leak for a particular process.
Total Leak Chunk Elem(b)	Amount of chunk block memory affected by leaks in bytes for a particular process.
Total Leak (bytes)	Total amount of memory leaked for a particular process.

Related Commands

Command	Description
set memory debug incremental starting-time	Sets the current time as the starting time for incremental analysis.
show memory debug incremental allocation	Displays all memory blocks that were allocated after the issue of the **set memory debug incremental starting-time** command.
show memory debug incremental leaks	Displays only memory that was leaked after the issue of the **set memory debug incremental starting-time** command.
show memory debug incremental leaks lowmem	Forces incremental memory leak detection to work in low memory mode. Displays only memory that was leaked after the issue of the **set memory debug incremental starting-time** command.

Command	Description
show memory debug incremental status	Displays if the starting point of incremental analysis has been defined and the time elapsed since then.

show memory debug references

To display debug information on references, use the **show memory debug references**command in user EXEC or privileged EXEC mode.

show memory debug references [**dangling** [*start-address start-address*]]

Syntax Description

dangling	(Optional) Displays the possible references to free memory.
start-address	(Optional) Address numbers <0-4294967295> that determine the address range.

Command Modes

User EXEC Privileged EXEC

Command History

Release	Modification
12.0	This command was introduced.

Usage Guidelines

All show memory debug commands must be used on customer networks only to diagnose the router for memory leaks when memory depletion is observed. These CLI's will have high CPU utilization and might result in time sensitive protocols to flap. These CLI's are recommended for customer use, only in the maintenance window when the router is not in a scaled condition.

Examples

The following is sample output from the **show memory debug references**command:

```
Router# show memory debug references 2 3
Address  Reference Cont_block Cont_block_name
442850BC        2  44284960   bss
44285110        3  44284960   bss
4429C33C        2  44284960   bss
4429C34C        2  44284960   bss
4429C35C        3  44284960   bss
.
.
.
```

The following is sample output from the **show memory debug references dangling**command:

```
Router# show memory debug references dangling
Address  Reference Free_block Cont_block Cont_block_name
442D5774 458CE5EC  458CE5BC   44284960   bss
442D578C 46602998  46602958   44284960   bss
442D58A0 465F9BC4  465F9B94   44284960   bss
442D58B8 4656785C  4656781C   44284960   bss
442D5954 45901E7C  45901E4C   44284960   bss
.
```

.
.

The table below describes the significant fields shown in the displays.

Table 101: show memory debug references Field Descriptions

Field	Description
Address	Hexadecimal address of the block having the given or dangling reference.
Reference	Address which is given or dangling.
Free_block	Address of the free block which now contains the memory referenced by the dangling reference.
Cont_block	Address of the control block which contains the block having the reference.
Cont_block_name	Name of the control block.

show memory debug unused

To display debug information on leaks that are accessible, but are no longer needed, use the **show memory debug unused**command in user EXEC or privileged EXEC mode.

show memory debug unused

Syntax Description

This command has no arguments or keywords.

Command Modes

User EXEC Privileged EXEC

Command History

Release	Modification
12.0	This command was introduced.

Examples

The following is sample output from the **show memory debug unused**command:

```
Router# show memory debug unused
Address  Alloc_pc PID  size     Name
654894B8 62BF31DC -2   44       *Init*
6549A074 601F7A84 -2   4464     XDI data
6549B218 601F7274 -2   4500     XDI data
6549DFB0 6089DDA4 42   84       Init
65509160 6089DDA4 1    84       *Init*
6550A260 6089DDA4 2    84       *Init*
6551FDB4 6089DDA4 4    84       *Init*
6551FF34 627EFA2C -2   24       *Init*
65520B3C 6078B1A4 -2   24       Parser Mode Q1
65520B88 6078B1C8 -2   24       Parser Mode Q2
65520C40 6078B1A4 -2   24       Parser Mode Q1
65520C8C 6078B1C8 -2   24       Parser Mode Q2
65520D44 6078B1A4 -2   24       Parser Mode Q1
65520D90 6078B1C8 -2   24       Parser Mode Q2
65520E48 6078B1A4 -2   24       Parser Mode Q1
65520E94 6078B1C8 -2   24       Parser Mode Q2
65520F4C 6078B1A4 -2   24       Parser Mode Q1
65520F98 6078B1C8 -2   24       Parser Mode Q2
65521050 6078B1A4 -2   24       Parser Mode Q1
6552109C 6078B1C8 -2   24       Parser Mode Q2
65521154 6078B1A4 -2   24       Parser Mode Q1
655211A0 6078B1C8 -2   24       Parser Mode Q2
.
.
.
```

The table below describes the significant fields shown in the display.

Table 102: show memory debug unused Field Descriptions

Field	Description
Address	Hexadecimal address of the block.

Field	Description
Alloc_pc	Address of the program counter that allocated the block.
PID	Process identifier of the process that allocated the block.
size	Size of the unused block (in bytes).
Name	Name of the process that owns the block.

show crypto debug-condition

To display crypto debug conditions that have already been enabled in the router, use the **show crypto debug-condition**command in privileged EXEC mode.

show crypto debug-condition [peer] [connid] [spi] [fvrf] [**gdoi-group** *groupname*] [**isakmp profile** *profile-name*] **[ivrf]** [**local** *ip-address*] **[unmatched]** [**username** *username*]

Syntax Description

peer	(Optional) Displays debug conditions related to the peer. Possible conditions can include peer IP address, subnet mask, hostname, username, and group key.
connid	(Optional) Displays debug conditions related to the connection ID.
spi	(Optional) Displays debug conditions related to the security parameter index (SPI).
fvrf	(Optional) Displays debug conditions related to the front-door virtual private network (VPN) routing and forwarding (FVRF) instance.
gdoi-group *groupname*	(Optional) Displays debug conditions related to the Group Domain of Interpretation (GDOI) group filter. • The *groupname*value is the name of the GDOI group.
isakmp profile *profile-name*	(Optional) Displays debug conditions related to the Internet Security Association Key Management Protocol (ISAKMP) profile filter. • The *profile-name*value is the name of the profile filter.
ivrf	(Optional) Displays debug conditions related to the inside VRF (IVRF) instance.
local *ip-address*	(Optional) Displays debug conditions related to the local address debug condition filters. • The *ip-address* is the IP address of the local crypto endpoint.

unmatched	(Optional) Displays debug messages related to the Internet Key Exchange (IKE), IP Security (IPsec), or the crypto engine, depending on what was specified via the **debug crypto condition unmatched** [**engine** | **gdoi-group**| **ipsec** | **isakmp**] command.
username *username*	(Optional) Displays debug messages related to the AAA Authentication (Xauth) or public key infrastructure (PKI) and authentication, authorization, and accounting (AAA) username filter.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.3(2)T	This command was introduced.
12.2(18)SXD	This command was integrated into Cisco IOS Release 12.2(18)SXD.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.4(11)T	The **gdoi-group** *groupname* **, isakmp profile** *profile-name* **, local** *ip-address* **,**and **username** *username* keywords and arguments were added.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.

Usage Guidelines

You can specify as many filter values as specified via the **debug crypto condition** command. (You cannot specify a filter value that you did not use in the **debug crypto condition** command.)

Examples

The following example shows how to display debug messages when the peer IP address is 10.1.1.1, 10.1.1.2, or 10.1.1.3 and when the connection ID 2000 of crypto engine 0 is used. This example also shows how to enable global debug crypto CLIs and enable the **show crypto debug-condition** command to verify conditional settings.

```
Router#
debug crypto condition connid 2000 engine-id 1
Router#
debug crypto condition peer ipv4 10.1.1.1
Router#
debug crypto condition peer ipv4 10.1.1.2
Router#
debug crypto condition peer ipv4 10.1.1.3
Router#
debug crypto condition unmatched
! Verify crypto conditional settings.
```

```
Router#
show crypto debug-condition
Crypto conditional debug currently is turned ON
IKE debug context unmatched flag:ON
IPsec debug context unmatched flag:ON
Crypto Engine debug context unmatched flag:ON
IKE peer IP address filters:
10.1.1.1  10.1.1.2   10.1.1.3
Connection-id filters:[connid:engine_id]2000:1,
! Enable global crypto CLIs to start conditional debugging.
Router#
debug crypto isakmp
Router#
debug crypto ipsec
Router#
debug crypto engine
```

The following example shows how to disable all crypto conditional settings via the **reset** keyword:

```
Router#
debug crypto condition reset
! Verify that all crypto conditional settings have been disabled.
Router#
show crypto debug-condition
Crypto conditional debug currently is turned OFF
IKE debug context unmatched flag:OFF
IPsec debug context unmatched flag:OFF
Crypto Engine debug context unmatched flag:OFF
```

Related Commands

Command	Description
debug crypto condition	Defines conditional debug filters.
debug crypto condition unmatched	Displays crypto conditional debug messages when context information is unavailable to check against debug conditions.

show debugging

To display information about the types of debugging that are enabled for your router, use the **show debugging** command in privileged EXEC mode.

show debugging

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
11.1	This command was introduced.
12.3(7)T	The output of this command was enhanced to show TCP Explicit Congestion Notification (ECN) configuration.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(31)SB2	This command was integrated into Cisco IOS Release 12.2(31)SB2.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.
12.4(20)T	The output of this command was enhanced to show the user-group debugging configuration.
Cisco IOS XE 3.3SE	This command was implemented in Cisco IOS XE Release 3.3SE.

Examples

The following is sample output from the **show debugging** command. In this example, the remote host is not configured or connected.

```
Router# show debugging
!
TCP:
  TCP Packet debugging is on
  TCP ECN debugging is on
!
Router# telnet 10.1.25.234
!
Trying 10.1.25.234 ...
!
00:02:48: 10.1.25.31:11001 <---> 10.1.25.234:23 out ECN-setup SYN
00:02:48: tcp0: O CLOSED 10.1.25.234:11001 10.1.25.31:23 seq 1922220018
        OPTS 4 ECE CWR SYN  WIN 4128
00:02:50: 10.1.25.31:11001 <---> 10.1.25.234:23 congestion window changes
00:02:50: cwnd from 1460 to 1460, ssthresh from 65535 to 2920
```

```
00:02:50: tcp0: R SYNSENT 10.1.25.234:11001 10.1.25.31:23 seq 1922220018
        OPTS 4 ECE CWR SYN  WIN 4128
00:02:54: 10.1.25.31:11001 <---> 10.1.25.234:23 congestion window changes
00:02:54: cwnd from 1460 to 1460, ssthresh from 2920 to 2920
00:02:54: tcp0: R SYNSENT 10.1.25.234:11001 10.1.25.31:23 seq 1922220018
        OPTS 4 ECE CWR SYN  WIN 4128
00:03:02: 10.1.25.31:11001 <---> 10.1.25.234:23 congestion window changes
00:03:02: cwnd from 1460 to 1460, ssthresh from 2920 to 2920
00:03:02: tcp0: R SYNSENT 10.1.25.234:11001 10.1.25.31:23 seq 1922220018
        OPTS 4 ECE CWR SYN  WIN 4128
00:03:18: 10.1.25.31:11001 <---> 10.1.25.234:23 SYN with ECN disabled
00:03:18: 10.1.25.31:11001 <---> 10.1.25.234:23 congestion window changes
00:03:18: cwnd from 1460 to 1460, ssthresh from 2920 to 2920
00:03:18: tcp0: O SYNSENT 10.1.25.234:11001 10.1.25.31:23 seq 1922220018
        OPTS 4 SYN  WIN 4128
00:03:20: 10.1.25.31:11001 <---> 10.1.25.234:23 congestion window changes
00:03:20: cwnd from 1460 to 1460, ssthresh from 2920 to 2920
00:03:20: tcp0: R SYNSENT 10.1.25.234:11001 10.1.25.31:23 seq 1922220018
        OPTS 4 SYN  WIN 4128
00:03:24: 10.1.25.31:11001 <---> 10.1.25.234:23 congestion window changes
00:03:24: cwnd from 1460 to 1460, ssthresh from 2920 to 2920
00:03:24: tcp0: R SYNSENT 10.1.25.234:11001 10.1.25.31:23 seq 1922220018
        OPTS 4 SYN  WIN 4128
00:03:32: 10.1.25.31:11001 <---> 10.1.25.234:23 congestion window changes
00:03:32: cwnd from 1460 to 1460, ssthresh from 2920 to 2920
00:03:32: tcp0: R SYNSENT 10.1.25.234:11001 10.1.25.31:23 seq 1922220018
        OPTS 4 SYN  WIN 4128
!Connection timed out; remote host not responding
```

The following is sample output from the **show debugging** command when user-group debugging is configured:

```
Router# show debugging
!
usergroup:
 Usergroup Deletions debugging is on
 Usergroup Additions debugging is on
 Usergroup Database debugging is on
 Usergroup API debugging is on
!
```

The following is sample output from the **show debugging** command when SNAP debugging is configured:

```
Router# show debugging
Persistent variable debugging is currently All
SNAP Server Debugging ON
SNAP Client Debugging ON
Router#
```

The table below describes the significant fields in the output.

Table 103: show debugging Field Descriptions

Field	Description
OPTS 4	Bytes of TCP expressed as a number. In this case, the bytes are 4.
ECE	Echo congestion experience.
CWR	Congestion window reduced.
SYN	Synchronize connections--Request to synchronize sequence numbers, used when a TCP connection is being opened.

Field	Description
WIN 4128	Advertised window size, in bytes. In this case, the bytes are 4128.
cwnd	Congestion window (cwnd)--Indicates that the window size has changed.
ssthresh	Slow-start threshold (ssthresh)--Variable used by TCP to determine whether or not to use slow-start or congestion avoidance.
usergroup	Statically defined usergroup to which source IP addresses are associated.

show debugging condition

To display the current state of debugging conditions, use the **show debugging condition**command in privileged EXEC mode.

show debugging condition [*condition-id*| **all**| **next-call** {**gprs**| **pdp**| **summary**}]

Syntax Description

condition-id	(Optional) Number of the condition for which you want to display its current state. The range is from 1 to 1000.
all	(Optional) Displays the current state for all conditions.
next-call	(Optional) Displays existing debug next-call conditions or Packet Data Protocol (PDP) with next-call debug conditions.
gprs	(Optional) Displays the information of all the (General Packet Radio System) GPRS under the next call debug condition.
pdp	(Optional) Displays the information of all the PDPs under the next call debug condition.
summary	(Optional) Displays existing debug next call conditions.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.4(22)YE	This command was introduced.
12.4(24)T	This command was integrated into a release earlier than Cisco IOS Release 12.4(24)T. The **gprs**, **pdp**, and **summary** keywords are not supported in T releases.

Usage Guidelin

Note The syntax of the command depends on your platform and release. The **next-call**, **gprs**, **pdp** and **summary** keywords are not supported in Cisco IOS Release 12.4(24)T and earlier releases.

Configure the **debug condition** command to enable conditional debgging.

Examples

The following is sample output from the **show debugging condition** command. The field descriptions are self-explanatory:

```
Router# show debugging condition
Condition 1: interface Fa0/0 (1 flags triggered)
        Flags: Fa0/0
Condition 2: interface Fa0/1 (1 flags triggered)
        Flags: Fa0/1
Condition 3: interface Et3/0 (1 flags triggered)
        Flags: Et3/0
Condition 4: username user1 (0 flags triggered)
```

Related Commands

Command	Description
debug condition	Limits output for some debug commands based on specified conditions.

voice call debug

To debug a voice call, use the **voice call debug** command in global configuration mode. To disable the **short-header** setting and return tothe **full-guid** setting, use the **no** form of this command.

{**voice call debug full-guid| short-header**}

{**no voice call debug full-guid| short-header**}

Syntax Description

full-guid	Displays the GUID in a 16-byte header. **Note** When the no version of this command is input with the full-guid keyword, the short 6-byte version displays. This is the default.
short-header	Displays the CallEntry ID in the header without displaying the GUID or module-specific parameters.

Command Default

The short 6-byte header displays.

Command Modes

Global configuration

Command History

Release	Modification
12.2(11)T	The new debug header was added to the following platforms: Cisco 2600 series, Cisco 3620, Cisco 3640, Cisco 3660 series, Cisco AS5300, Cisco AS5350, Cisco AS5400, Cisco AS5800, Cisco AS5850, and Cisco MC3810.
12.2(15)T	The header-only keyword was replaced by the short-header keyword.

Usage Guidelines

Despite its nontraditional syntax (trailing rather than preceding "debug"), this is a normal **debug** command.

You can control the contents of the standardized header. Display options for the header are as follows:

- Short 6-byte GUID
- Full 16-byte GUID
- Short header which contains only the CallEntry ID

The format of the GUID headers is as follows: //CallEntryID/GUID/Module-Dependent-List/Function-name:.

The format of the short header is as follows: //CallEntryID/Function-name:.

When the voice call debug short-header command is entered, the header displays with no GUID or module-specific parameters. When the no voice call debug short-header command is entered, the header, the

6-byte GUID, and module-dependent parameter output displays. The default option is displaying the 6-byte GUID trace.

Note Using the no form of this command does not turn off debugging.

Examples

The following is sample output when the full-guid keyword is specified:

```
Router# voice call debug full-guid
!
00:05:12: //1/0E2C8A90-BC00-11D5-8002-DACCFDCEF87D/VTSP:(0:D):0:0:4385/vtsp_insert_cdb:
00:05:12: //-1/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx/CCAPI/cc_incr_if_call_volume: 00:05:12:
 //1/0E2C8A90-BC00-11D5-8002-DACCFDCEF87D/VTSP:(0:D):0:0:4385/vtsp_open_voice_and_set_params:
00:05:12:
//1/0E2C8A90-BC00-11D5-8002-DACCFDCEF87D/VTSP:(0:D):0:0:4385/vtsp_modem_proto_from_cdb:
00:05:12: //1/0E2C8A90-BC00-11D5-8002-DACCFDCEF87D/VTSP:(0:D):0:0:4385/set_playout_cdb:
00:05:12:
//1/0E2C8A90-BC00-11D5-8002-DACCFDCEF87D/VTSP:(0:D):0:0:4385/vtsp_dsp_echo_canceller_control:
```

Note The "//-1/" output indicates that CallEntryID for the CCAPI module is not available.

The table below describes significant fields shown in the display.

Table 104: voice call debug full-guid Field Descriptions

Field	Description
VTSP:(0:D):0:0:4385	VTSP module, port name, channel number, DSP slot, and DSP channel number.
vtsp_insert_cdb	Function name.
CCAPI	CCAPI module.

The following is sample output when the short-header keyword is specified:

```
Router(config)# voice call debug short-header
!
00:05:12: //1/vtsp_insert_cdb:
00:05:12: //-1/cc_incr_if_call_volume:
00:05:12: //1/vtsp_open_voice_and_set_params:
00:05:12: //1/vtsp_modem_proto_from_cdb:
00:05:12: //1/set_playout_cdb:
00:05:12: //1/vtsp_dsp_echo_canceller_control:
```

Note The "//-1/" output indicates that CallEntryID for CCAPI is not available.

Related Commands

Command	Description
debug rtsp api	Displays debug output for the RTSP client API.
debug rtsp client session	Displays debug output for the RTSP client data.
debug rtsp error	Displays error message for RTSP data.
debug rtsp pmh	Displays debug messages for the PMH.
debug rtsp socket	Displays debug output for the RTSP client socket data.
debug voip ccapi error	Traces error logs in the CCAPI.
debug voip ccapi inout	Traces the execution path through the CCAPI.
debug voip ivr all	Displays all IVR messages.
debug voip ivr applib	Displays IVR API libraries being processed.
debug voip ivr callsetup	Displays IVR call setup being processed.
debug voip ivr digitcollect	Displays IVR digits collected during the call.
debug voip ivr dynamic	Displays IVR dynamic prompt play debug.
debug voip ivr error	Displays IVR errors.
debug voip ivr script	Displays IVR script debug.
debug voip ivr settlement	Displays IVR settlement activities.
debug voip ivr states	Displays IVR states.
debug voip ivr tclcommands	Displays the TCL commands used in the script.
debug voip rawmsg	Displays the raw VoIP message.
debug vtsp all	Enables **debug vtsp session**, **debug vtsp error**, and **debug vtsp dsp**.
debug vtsp dsp	Displays messages from the DSP.
debug vtsp error	Displays processing errors in the VTSP.
debug vtsp event	Displays the state of the gateway and the call events.

Command	Description
debug vtsp port	Limits VTSP debug output to a specific voice port.
debug vtsp rtp	Displays the voice telephony RTP packet debugging.
debug vtsp send-nse	Triggers the VTSP software module to send a triple redundant NSE.
debug vtsp session	Traces how the router interacts with the DSP.
debug vtsp stats	Debugs periodic statistical information sent and received from the DSP
debug vtsp vofr subframe	Displays the first 10 bytes of selected VoFR subframes for the interface.
debug vtsp tone	Displays the types of tones generated by the VoIP gateway.