Legal and notice information

© Copyright 2014 Hewlett-Packard Development Company, L.P.

No part of this documentation may be reproduced or transmitted in any form or by any means without prior written consent of Hewlett-Packard Development Company, L.P.

The information contained herein is subject to change without notice.

HEWLETT-PACKARD COMPANY MAKES NO WARRANTY OF ANY KIND WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. Hewlett-Packard shall not be liable for errors contained herein or for incidental or consequential damages in connection with the furnishing, performance, or use of this material.

The only warranties for HP products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. HP shall not be liable for technical or editorial errors or omissions contained herein.
## Contents

802.1X debugging commands ................................................................. 6
  debugging dot1x................................................................................. 6

AAA debugging commands .................................................................. 14
  debugging aaa.................................................................................. 14

RADIUS debugging commands ......................................................... 21
  debugging radius.............................................................................. 21

HWTACACS debugging commands .................................................... 28
  debugging hwtacacs......................................................................... 28

LDAP debugging commands ............................................................... 44
  debugging aaa-ldap.......................................................................... 44

SC debugging commands .................................................................... 51
  debugging sc.................................................................................... 51

Local server debugging commands ...................................................... 53
  debugging local-server ........................................................................ 53

RADIUS server debugging commands ................................................. 55
  debugging radius-server.................................................................. 55

ACFP debugging commands ............................................................... 57
  debugging acfp client...................................................................... 57
  debugging acfp server...................................................................... 58

ACSEI debugging commands .............................................................. 61
  ACSEI server debugging commands .................................................. 61
    debugging acsei server................................................................ 61
  ACSEI client debugging commands .................................................. 64
    debugging acsei-client................................................................. 64

Active and standby switchover debugging commands ......................... 69
  debugging ha.................................................................................. 69

Adjacency table debugging commands ............................................... 74
  IPv4 adjacency table debugging commands ...................................... 74
    debugging adjacent-table all...................................................... 74
    debugging adjacent-table entry.................................................. 74
    debugging adjacent-table event.................................................. 76
  IPv6 adjacency table debugging commands ...................................... 77
    debugging ipv6 adjacent-table all............................................... 77
    debugging ipv6 adjacent-table entry......................................... 78
    debugging ipv6 adjacent-table event......................................... 79

Service flow redirection debugging commands .................................. 81
  debugging flow-redirect................................................................. 81

AFT debugging commands ............................................................... 83
  debugging aft................................................................................... 83

ALG debugging commands ............................................................... 88
  debugging alg.................................................................................. 88
<table>
<thead>
<tr>
<th>Category</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>BFD debugging commands</td>
<td>160</td>
</tr>
<tr>
<td>debugging bfd timer</td>
<td>174</td>
</tr>
<tr>
<td>debugging bfd packet</td>
<td>172</td>
</tr>
<tr>
<td>debugging bfd fsm</td>
<td>176</td>
</tr>
<tr>
<td>debugging bfd error</td>
<td>165</td>
</tr>
<tr>
<td>debugging bfd drv</td>
<td>162</td>
</tr>
<tr>
<td>debugging bfd event</td>
<td>166</td>
</tr>
<tr>
<td>debugging bfd all</td>
<td>160</td>
</tr>
<tr>
<td>debugging bfd all</td>
<td>160</td>
</tr>
<tr>
<td>debugging mpls fast-forwarding</td>
<td>133</td>
</tr>
<tr>
<td>debugging mpls ldp</td>
<td>134</td>
</tr>
<tr>
<td>debugging mpls ltid</td>
<td>140</td>
</tr>
<tr>
<td>debugging mpls ltid slot drv</td>
<td>143</td>
</tr>
<tr>
<td>debugging mpls lspc</td>
<td>147</td>
</tr>
<tr>
<td>debugging mpls management</td>
<td>149</td>
</tr>
<tr>
<td>debugging mpls packet</td>
<td>156</td>
</tr>
<tr>
<td>debugging tnlm</td>
<td>159</td>
</tr>
<tr>
<td>Attack detection and protection debugging commands</td>
<td>112</td>
</tr>
<tr>
<td>debugging attack-defense</td>
<td>112</td>
</tr>
<tr>
<td>debugging blacklist</td>
<td>115</td>
</tr>
<tr>
<td>debugging flow-statistics</td>
<td>116</td>
</tr>
<tr>
<td>debugging tcp-proxy</td>
<td>117</td>
</tr>
<tr>
<td>Basic IP routing debugging commands</td>
<td>119</td>
</tr>
<tr>
<td>debugging rm all</td>
<td>119</td>
</tr>
<tr>
<td>debugging rm backup</td>
<td>119</td>
</tr>
<tr>
<td>debugging rm bfd</td>
<td>120</td>
</tr>
<tr>
<td>debugging rm ipv4</td>
<td>121</td>
</tr>
<tr>
<td>debugging rm ipv6</td>
<td>123</td>
</tr>
<tr>
<td>debugging rm jio</td>
<td>126</td>
</tr>
<tr>
<td>debugging rm policy</td>
<td>127</td>
</tr>
<tr>
<td>debugging rm system</td>
<td>130</td>
</tr>
<tr>
<td>debugging rm task</td>
<td>130</td>
</tr>
<tr>
<td>debugging rm timer</td>
<td>131</td>
</tr>
<tr>
<td>ATM debugging commands</td>
<td>105</td>
</tr>
<tr>
<td>debugging atm all</td>
<td>105</td>
</tr>
<tr>
<td>debugging atm error</td>
<td>105</td>
</tr>
<tr>
<td>debugging atm event</td>
<td>107</td>
</tr>
<tr>
<td>debugging atm oam</td>
<td>109</td>
</tr>
<tr>
<td>debugging atm packet</td>
<td>109</td>
</tr>
<tr>
<td>Basic MPLS debugging commands</td>
<td>133</td>
</tr>
<tr>
<td>debugging mpls fast-forwarding</td>
<td>133</td>
</tr>
<tr>
<td>debugging mpls ldp</td>
<td>134</td>
</tr>
<tr>
<td>debugging mpls ltid</td>
<td>140</td>
</tr>
<tr>
<td>debugging mpls ltid slot drv</td>
<td>143</td>
</tr>
<tr>
<td>debugging mpls lspc</td>
<td>147</td>
</tr>
<tr>
<td>debugging mpls management</td>
<td>149</td>
</tr>
<tr>
<td>debugging mpls packet</td>
<td>156</td>
</tr>
<tr>
<td>debugging tnlm</td>
<td>159</td>
</tr>
<tr>
<td>ARP debugging commands</td>
<td>96</td>
</tr>
<tr>
<td>debugging arp packet</td>
<td>96</td>
</tr>
<tr>
<td>debugging arp status</td>
<td>97</td>
</tr>
<tr>
<td>ARP snooping debugging commands</td>
<td>99</td>
</tr>
<tr>
<td>debugging arp-snooping</td>
<td>99</td>
</tr>
<tr>
<td>ARP fast-reply debugging commands</td>
<td>102</td>
</tr>
<tr>
<td>debugging arp fast-reply</td>
<td>102</td>
</tr>
<tr>
<td>ATM debugging commands</td>
<td>105</td>
</tr>
<tr>
<td>debugging atm all</td>
<td>105</td>
</tr>
<tr>
<td>debugging atm error</td>
<td>105</td>
</tr>
<tr>
<td>debugging atm event</td>
<td>107</td>
</tr>
<tr>
<td>debugging atm oam</td>
<td>109</td>
</tr>
<tr>
<td>debugging atm packet</td>
<td>109</td>
</tr>
<tr>
<td>Attack detection and protection debugging commands</td>
<td>112</td>
</tr>
<tr>
<td>debugging attack-defense</td>
<td>112</td>
</tr>
<tr>
<td>debugging blacklist</td>
<td>115</td>
</tr>
<tr>
<td>debugging flow-statistics</td>
<td>116</td>
</tr>
<tr>
<td>debugging tcp-proxy</td>
<td>117</td>
</tr>
<tr>
<td>Basic IP routing debugging commands</td>
<td>119</td>
</tr>
<tr>
<td>debugging rm all</td>
<td>119</td>
</tr>
<tr>
<td>debugging rm backup</td>
<td>119</td>
</tr>
<tr>
<td>debugging rm bfd</td>
<td>120</td>
</tr>
<tr>
<td>debugging rm ipv4</td>
<td>121</td>
</tr>
<tr>
<td>debugging rm ipv6</td>
<td>123</td>
</tr>
<tr>
<td>debugging rm jio</td>
<td>126</td>
</tr>
<tr>
<td>debugging rm policy</td>
<td>127</td>
</tr>
<tr>
<td>debugging rm system</td>
<td>130</td>
</tr>
<tr>
<td>debugging rm task</td>
<td>130</td>
</tr>
<tr>
<td>debugging rm timer</td>
<td>131</td>
</tr>
<tr>
<td>ATM debugging commands</td>
<td>105</td>
</tr>
<tr>
<td>debugging atm all</td>
<td>105</td>
</tr>
<tr>
<td>debugging atm error</td>
<td>105</td>
</tr>
<tr>
<td>debugging atm event</td>
<td>107</td>
</tr>
<tr>
<td>debugging atm oam</td>
<td>109</td>
</tr>
<tr>
<td>debugging atm packet</td>
<td>109</td>
</tr>
</tbody>
</table>
BGP debugging commands
- debugging bgp .......................................................... 176
- debugging bgp all ..................................................... 179
- debugging bgp bfd .................................................... 181
- debugging bgp detail ................................................ 183
- debugging bgp event ............................................... 185
- debugging bgp graceful-restart .................................. 186
- debugging bgp non-stop-routing ............................... 187
- debugging bgp timer ............................................... 188
- debugging bgp update ............................................. 189
- debugging bgp update ipv4 ...................................... 190
- debugging bgp update ipv6 ...................................... 192
- debugging bgp update ipv6 vpn-instance ................... 194
- debugging bgp update l2vpn .................................... 195
- debugging bgp update label-route ....................... 196
- debugging bgp update mdt .................................... 198
- debugging bgp update peer .................................... 199
- debugging bgp update vpls ................................... 200
- debugging bgp update vpn-instance ....................... 201
- debugging bgp update vpnv4 ............................... 202
- debugging bgp update vpnv6 ............................... 204
- debugging bgp update mdt .................................. 198
- debugging bgp update label-route ....................... 196
- debugging bgp update peer .................................... 199
- debugging bgp update vpls ................................... 200
- debugging bgp update vpn-instance ....................... 201
- debugging bgp update vpnv4 ............................... 202
- debugging bgp update vpnv6 ............................... 204

Bridging debugging commands
- debugging bridge error ........................................ 205
- debugging bridge eth-forwarding ............................ 207
- debugging bridge event ....................................... 209

Call services debugging commands
- debugging voice ss cb .......................................... 213
- debugging voice ss cf .......................................... 215
- debugging voice ss ch .......................................... 226
- debugging voice ss conf ........................................ 240
- debugging voice ss cr .......................................... 242
- debugging voice ss cf .......................................... 244
- debugging voice ss cw .......................................... 255
- debugging voice ss dr .......................................... 262
- debugging voice ss ft .......................................... 264
- debugging voice ss hg .......................................... 271
- debugging voice ss mwi ........................................ 273
- debugging voice ss sa .......................................... 276
- debugging voice ss sm .......................................... 279
- debugging voice ss rc .......................................... 292
- debugging voice ss join ....................................... 295

Call-watch debugging commands
- debugging call-watch event .................................. 298

CFD debugging commands
- debugging cfd .................................................. 301

Cluster management debugging commands
- debugging ndp packet ........................................... 308
- debugging ndp .................................................. 308
- debugging cluster .............................................. 310
- debugging cluster ext ......................................... 314
DLSw debugging commands ................................................................. 388
   debugging dlsw .................................................................................. 388
LLC2 debugging commands ................................................................. 404
   debugging llc2 .................................................................................. 404
SDLC debugging commands .................................................................. 408
   debugging sdlc .................................................................................. 408
DNS debugging commands ................................................................. 413
  debugging dns .................................................................................. 413
  debugging dns proxy ............................................................... 415
DDNS debugging commands ................................................................. 419
  debugging ddns ............................................................................ 419
DVPN debugging commands ................................................................. 423
  debugging dvpn ............................................................................ 423
  debugging vam client ................................................................. 425
  debugging vam server .............................................................. 426
Some information in this chapter is device type specific. Devices in this chapter are categorized depending on their IRF capability and support for interface cards that use independent processors for forwarding traffic, as shown in Table 1.

**Table 1 Device types**

<table>
<thead>
<tr>
<th>Device type</th>
<th>Interface cards with on-card processors</th>
<th>IRF capability</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Distributed devices</td>
<td>Yes</td>
<td>No</td>
<td>HP 6600 routers (except for 6602)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Yes (in standalone mode)</td>
<td>HP 12500 switches HP 10500 switches</td>
</tr>
<tr>
<td>Distributed IRF devices</td>
<td>Yes</td>
<td>Yes (in IRF mode)</td>
<td>HP 12500 switches HP 10500 switches</td>
</tr>
<tr>
<td>Centralized devices</td>
<td>No</td>
<td>No</td>
<td>HP MSR routers HP 6602 router</td>
</tr>
<tr>
<td>Centralized IRF devices</td>
<td>No</td>
<td>Yes</td>
<td>HP 5800 switches HP 5500 switches</td>
</tr>
</tbody>
</table>

The output description tables in this document only contain fields and messages that require an explanation.

**debugging dot1x**

Use `debugging dot1x` to enable 802.1X debugging.

Use `undo debugging dot1x` to disable 802.1X debugging.

**Syntax**

Centralized devices:

```plaintext
debugging dot1x { all | error | event | packet }
undo debugging dot1x { all | error | event | packet }
```

Distributed devices/centralized IRF devices:

```plaintext
debugging dot1x { all | error | event | packet } [ slot slot-number ]
undo debugging dot1x { all | error | event | packet } [ slot slot-number ]
```

Distributed IRF devices:

```plaintext
debugging dot1x { all | error | event | packet } [ chassis chassis-number slot slot-number ]
undo debugging dot1x { all | error | event | packet } [ chassis chassis-number slot slot-number ]
```

**Default**

802.1X debugging is disabled.
Views

User view

Default command level

1: Monitor level

Parameters

all: Specifies all types of 802.1X debugging.
error: Specifies 802.1X error debugging.
event: Specifies 802.1X event debugging.
packet: Specifies 802.1X packet debugging.
slot slot-number: Specifies a card by its slot number. (Distributed devices—In standalone mode.)
slot slot-number: Specifies an IRF member device by its member ID. (Centralized IRF devices.)
chassis chassis-number slot slot-number: Specifies a card on an IRF member device. The chassis-number argument represents the IRF member ID of the device. The slot-number argument represents the slot number of the card. (Distributed devices—In IRF mode.)

Usage guidelines

Table 2 describes output fields and messages for the debugging dot1x error command.

Table 2 Output from the debugging dot1x error command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UCIB index: UCIBIndex</td>
<td>User control information block index.</td>
</tr>
<tr>
<td>Wrong PortIfIndex.</td>
<td>Illegal port logical index.</td>
</tr>
<tr>
<td>(SmartOn) Received an Error msg(ErrCode)…</td>
<td>The SmartOn feature received an error message. ErrCode represents the error code.</td>
</tr>
<tr>
<td>Check Reauth author info failed!!!</td>
<td>802.1X failed to examine re-authentication information.</td>
</tr>
<tr>
<td>specify VLAN ID to PVID failed.</td>
<td>802.1X failed to specify a VLAN ID as the PVID of the port.</td>
</tr>
<tr>
<td>DOT1X SuccessTrans Received an Error Code ulParam2(ErrCode) of MSG:GuestVlanUnAsgnResp…</td>
<td>In success state, 802.1X failed to clear the guest VLAN information and received an error message. ErrCode represents the error code.</td>
</tr>
<tr>
<td>DOT1X SuccessTrans Received an Error Code ulParam2(ErrCode) of MSG:VlanAsgnResp…</td>
<td>In success state, 802.1X failed to assign a VLAN to the user and received an error message. ErrCode represents the error code.</td>
</tr>
<tr>
<td>DOT1X SuccessTrans Received an Error Code ulParam2(ErrCode) of MSG:VlanUNAsgnResp…</td>
<td>In success state, 802.1X failed to remove users from the assigned VLAN and received an error message. ErrCode represents the error code.</td>
</tr>
<tr>
<td>DOT1X SuccessTrans Received an Error msg(ErrCode)…</td>
<td>802.1X received a message that should not be received in success state.</td>
</tr>
<tr>
<td>RealyState: when Relayed, receive a relay msg again</td>
<td>802.1X received an 802.1X request again from the client.</td>
</tr>
<tr>
<td>DOT1X UnauthorTrans recv SC AsgnVlan2Port Resp: ulParam2=ErrCode</td>
<td>802.1X failed to assign the VLAN. ErrCode represents the error code.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td>DOT1X UnauthorTrans Receive an Error msg(ErrCode)...</td>
<td>802.1X received a message that should not be received in unauthorized state.</td>
</tr>
<tr>
<td>Bad msg.</td>
<td>The message had invalid parameters.</td>
</tr>
<tr>
<td>SlaveBoard/IO: DOT1X Config By Port failed</td>
<td>802.1X failed to configure ports on the standby MPU or interface cards.</td>
</tr>
<tr>
<td>MAIN/Slave: RPC get DOT1X Config By Port res: failed</td>
<td>The active MPU or the standby MPU failed to obtain the port-specific 802.1X configuration from interface cards.</td>
</tr>
<tr>
<td>MasterBoard: DOT1X Config By Port failed</td>
<td>The active MPU failed to obtain the port-specific 802.1X configuration on interface cards.</td>
</tr>
<tr>
<td>Failed to get this port name string because it doesn’t exist or isn’t up.</td>
<td>802.1X failed to get a port name because the port did not exist or was down.</td>
</tr>
<tr>
<td>DOT1X set LogicState failed</td>
<td>802.1X failed to set the control state of the logical port.</td>
</tr>
<tr>
<td>DOT1XSOP unAsgn GuestVlan from port failed</td>
<td>DOT1XSOP failed to cancel a guest VLAN assignment.</td>
</tr>
<tr>
<td>Warning:DOT1X Que is almost full, CUT user msg is dropped.</td>
<td>CUT user messages were dropped because the 802.1X queue was getting full.</td>
</tr>
<tr>
<td>Failed to del 1X drv-MAC in DOT1X Backup Deal.</td>
<td>802.1X failed to delete the drive MAC address during backup.</td>
</tr>
<tr>
<td>Fail to get port MAC Max LearnStatus on port :PortIndex ,Errcode is :ErrCode</td>
<td>802.1X failed to get the maximum number of MAC addresses allowed on the specified port.</td>
</tr>
<tr>
<td>Failed to move MBUF to buffer.</td>
<td>The device failed to move BPDUs to contiguous blocks of memory.</td>
</tr>
<tr>
<td>Failed to cut ethernet head</td>
<td>The device failed to ignore the Ethernet header of packets.</td>
</tr>
<tr>
<td>Failed to process MGV for the same MGV entry.</td>
<td>802.1X failed to process MGV because the same MGV entry already exists.</td>
</tr>
<tr>
<td>Failed to unauthorize port-based LogicState.</td>
<td>802.1X failed to unauthorize the port-based logic state.</td>
</tr>
<tr>
<td>Port is unauthorized, but unassigning Guest VLAN failed.</td>
<td>Port was unauthorized, but 802.1X failed to remove the guest VLAN of the port.</td>
</tr>
<tr>
<td>Secure handshake check: Failures=fail-num, Flag=flag.</td>
<td>Handshake security check:</td>
</tr>
<tr>
<td>Removed secure handshake hash of handshake packet received from host.</td>
<td>802.1X removed the hash field from the handshake packet received from the client.</td>
</tr>
<tr>
<td>Failed to delete the critical VLAN entry (mac-address) on the interface (IfIndex=ifindx).</td>
<td>802.1X failed to delete the MAC-based critical VLAN entry on the interface.</td>
</tr>
</tbody>
</table>

Table 3 describes output fields and messages for the **debugging dot1x event** command.
Table 3 Output from the debugging dot1x event command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IfIndex : IfIndex</td>
<td>Interface index.</td>
</tr>
<tr>
<td>port : PortIndex</td>
<td>Port index.</td>
</tr>
<tr>
<td>Status : Status</td>
<td>MAC address type.</td>
</tr>
<tr>
<td>cmd = CmdType</td>
<td>Command.</td>
</tr>
<tr>
<td>return code: Errcode</td>
<td>Return value or error code.</td>
</tr>
<tr>
<td>EAPoL-Logoff...</td>
<td>802.1X received an EAPOL-Logoff frame.</td>
</tr>
<tr>
<td>Executed Unauthoring op first for Reauth...</td>
<td>802.1X is processing the first unauthorized request of re-authentication.</td>
</tr>
<tr>
<td>DOT1X SuccessTrans Receive an Error Code</td>
<td>The node in success state received a VLAN assignment response.</td>
</tr>
<tr>
<td>ulParam2(ErrCode) of MSG:VlanAsgnResp...</td>
<td>Re-authentication timer expired in DOT1X Auth WorkTrans state.</td>
</tr>
<tr>
<td>DOT1X Auth WorkTrans state: Reauthentication times out.</td>
<td>A user needs to be unauthorized before the user can be re-authenticated.</td>
</tr>
<tr>
<td>User needs to be unauthorized before being reauthenticated.</td>
<td>Unauthorized state node received a VLAN unassignment response from the SC module.</td>
</tr>
<tr>
<td>Get Port Portsec info for IfIndex = IfIndex,cmd = CmdType</td>
<td>Unauthorized state node received a VLAN unassignment response from the SC module.</td>
</tr>
<tr>
<td>Get Port Portsec info for IfIndex = IfIndex,cmd = CmdType,return code = ErrCode</td>
<td>802.1X sent a request to the driver for getting port security information of a port.</td>
</tr>
<tr>
<td>DOT1X set LogicState failed</td>
<td>802.1X failed to set the control status of logical ports.</td>
</tr>
<tr>
<td>DOT1XSOP unAsgn GuestVlan from port failed</td>
<td>802.1X failed to unassign guest VLAN.</td>
</tr>
<tr>
<td>Get port MAC Max LearnStatus on port : PortIndex</td>
<td>802.1X got the maximum number of MAC addresses that can be learnt on the port.</td>
</tr>
<tr>
<td>count mac should be sync to this slot .</td>
<td>count MAC addresses should be synchronized to the specified card.</td>
</tr>
<tr>
<td>count mac(s) sent to this slot .</td>
<td>802.1X sent count MAC addresses to the specified card.</td>
</tr>
<tr>
<td>sent mac-addr to this slot finished.</td>
<td>802.1X finished to send MAC addresses to the specified card.</td>
</tr>
<tr>
<td>AuthorVlan IPC Req sent.</td>
<td>802.1X sent a VLAN assignment IPC request.</td>
</tr>
<tr>
<td>UnAuthorVlan IPC Req sent.</td>
<td>802.1X sent a VLAN unassignment IPC request.</td>
</tr>
<tr>
<td>AuthorVlan IPC Resp sent.</td>
<td>802.1X sent a VLAN assignment IPC response.</td>
</tr>
<tr>
<td>UnAuthorVlan IPC Resp sent.</td>
<td>802.1X sent a VLAN unassignment IPC response.</td>
</tr>
<tr>
<td>Recv:AuthorVlan Req</td>
<td>802.1X received a VLAN authorization request.</td>
</tr>
<tr>
<td>Recv:UnAuthorVlan Req</td>
<td>802.1X received a VLAN unauthorization request.</td>
</tr>
<tr>
<td>Recv:AuthorVlan Resp</td>
<td>802.1X received a VLAN authorization response.</td>
</tr>
<tr>
<td>Recv:UnAuthorVlan Resp</td>
<td>802.1X received a VLAN unauthorization response.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Recv:Broadcast Kickoff PreUser IPC Request</td>
<td>802.1X received a broadcast to log off logged-in users.</td>
</tr>
<tr>
<td>DOT1XSOP: Vlan HashTable Created.[port and auth index is nonsense.]</td>
<td>The SOP module created a VLAN authorization hash table.</td>
</tr>
<tr>
<td>DOT1XSOP: Vlan HashTable Deleted.[port and auth index is nonsense.]</td>
<td>The SOP module deleted the VLAN authorization hash table.</td>
</tr>
<tr>
<td>Notify DOT1X of vlan UnAsgn success</td>
<td>The device sent a message to notify 802.1X that a VLAN was removed successfully.</td>
</tr>
<tr>
<td>GuestVLAN ASGN/UNASGN SUCCEED!</td>
<td>802.1X assigned a guest VLAN to or removed it from a port successfully.</td>
</tr>
<tr>
<td>Authoring authored Vlan...</td>
<td>802.1X is authorizing an unauthorized VLAN.</td>
</tr>
<tr>
<td>Unauthoring authored Vlan...</td>
<td>802.1X is unauthorizing an authorized VLAN.</td>
</tr>
<tr>
<td>Author operation finished!</td>
<td>VLAN authorization completed.</td>
</tr>
<tr>
<td>Unauthor operation finished!</td>
<td>VLAN unauthorization completed.</td>
</tr>
<tr>
<td>Author info is INVALID, need not unauthor operation.</td>
<td>Authorization information is invalid. Do not need unauthorization.</td>
</tr>
<tr>
<td>BroadcastKickoffPreUser IPC Request</td>
<td>802.1X broadcast IPC request to log off duplicated 802.1X users.</td>
</tr>
<tr>
<td>Receive reset drv msg</td>
<td>802.1X received a message to reset the driver.</td>
</tr>
<tr>
<td>Receive clear drv msg</td>
<td>802.1X received a message to clear the driver.</td>
</tr>
<tr>
<td>Clear 802.1X drv .</td>
<td>802.1X driver was cleared.</td>
</tr>
<tr>
<td>Set 802.1X function to driver.</td>
<td>802.1X driver was reset.</td>
</tr>
<tr>
<td>Waiting 802.1X users off-line.</td>
<td>802.1X is waiting for 802.1X users to go offline.</td>
</tr>
<tr>
<td>Slot Insert, PortSec Enabled, set flag in portnum!</td>
<td>A card was inserted in the slot, and port security was enabled. 802.1X set a flag for ports to be controlled by port security on the card.</td>
</tr>
<tr>
<td>Rxed EAPOL packet. The port is unavailable.</td>
<td>The port received an EAPOL frame. The port is unavailable.</td>
</tr>
<tr>
<td>Rxed EAPOL packet. 802.1X is not enabled on this port.</td>
<td>The port received an EAPOL frame. 802.1X is not enabled on the port.</td>
</tr>
<tr>
<td>Rxed EAPOL packet. Force mode enabled on this port.</td>
<td>The port received an EAPOL frame. The port is in force-authorize state or non-authorize state. It does not process frames.</td>
</tr>
<tr>
<td>Portsec PreHandle 1X req!!</td>
<td>The port security module preprocessed the 802.1X authentication trigger packets.</td>
</tr>
<tr>
<td>Failed to alloc resource.</td>
<td>802.1X failed to allocate resources.</td>
</tr>
<tr>
<td>Alloc or search resource successfully.</td>
<td>802.1X allocated or queried resources successfully.</td>
</tr>
<tr>
<td>Send Msg EAPOL-START to 802.1X-Msg-Queue successfully.</td>
<td>A port sent an EAPOL-START message successfully to the 802.1X message queue.</td>
</tr>
<tr>
<td>Send Msg EAPOL-LOGOFF to 802.1X-Msg-Queue successfully.</td>
<td>A port sent an EAPOL-LOGOFF message successfully to the 802.1X message queue.</td>
</tr>
</tbody>
</table>
Receive shakehanding-pkt with proxy-chk tag | A port received a handshake packet with a proxy-check tag.
---|---
Port control mode is portbase and there's users on port portindex | The port control mode is port-based, and there are users on port portindex.
Auth-Fail VLAN status is set to INITIAL because unauthorization is successful. | The status of the Auth-Fail VLAN was set to INITIAL because unauthorization was successful.
Deleted the MGV(mac-address) entry on same interface (IfIndex=ifindex) after 802.1X authentication failed. | After 802.1X authentication failed on an interface (ifindex), the MGV entry (mac-address) was deleted on the interface.
Deleted the critical VLAN entry (mac-address) before adding it to the guest VLAN. | 802.1X deleted the critical VLAN entry of the MAC address before it adds the MAC address to the guest VLAN.
Processing node CRITICAL... | 802.1X is processing the nodes in CRITICAL state.

Table 4 describes output fields and messages for the **debugging dot1x packet** command.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Code</td>
<td>Code</td>
</tr>
<tr>
<td>Identifier</td>
<td>Identifier</td>
</tr>
<tr>
<td>Length</td>
<td>Length</td>
</tr>
</tbody>
</table>

**Examples**

# Enable 802.1X error debugging. Output similar to the following example is generated when a user requests authentication under the following conditions:

- 802.1X is enabled on the device, and a RADIUS scheme is configured for user authentication.
- VLAN 6 is configured as the 802.1X guest VLAN on GigabitEthernet 1/1, and the port performs MAC-based access control.
- The multicast packet retransmission has timed out.
- The network card of the client is disabled after the user passes authentication.

```
<Sysname> debugging dot1x error
*May 1 14:11:41:784 2000 Sysname 8021X/7/Error::Port:GigabitEthernet1/1, GuestVlan assign failed, for port-mode is not portbased!
```

// **802.1X failed to specify a guest VLAN for GigabitEthernet 1/1, because the port access control mode is not portbased.**

```
*May 1 14:55:05:792 2000 Sysname 8021X/7/Error::Port:GigabitEthernet1/1, Auth:211,Handshake time out ,make the user logoff...
```

// **The handshake packet timed out and the user was logged off after the network card was disabled.**

# Enable 802.1X event debugging. Output similar to the following example is generated when a user requests authentication under the following conditions:

- 802.1X is enabled on the device.
- A RADIUS scheme is configured for user authentication.

```
<Sysname> debugging dot1x event
```
// 802.1X allocated resources for the new connection because it does not conflict with existing ones.

// GigabitEthernet 1/1 received an EAP-Start packet from the supplicant. The state machine entered the Connecting state.

// GigabitEthernet 1/1 received an Identity reply packet from the supplicant. The state machine entered the Challenge state.

// GigabitEthernet 1/1 received a Challenge reply packet from the supplicant. The state machine entered the Response state.

// GigabitEthernet 1/1 received successful response from ACM. The state machine entered the Success state.

// Authorization completed. The state machine entered the Working state.

// Authentication succeeded. 802.1X sent authentication success message to the client.
May  1 12:29:15:436 2000 Sysname 8021X/7/EVENT:Auth:204,Failed to send EAPoL-Notification...

// 802.1X failed to send notification to the client.

# Enable 802.1X packet debugging. Output similar to the following example is generated when a user requests authentication under the following conditions:

- 802.1X is enabled on the device.
- A RADIUS scheme is configured for user authentication.

<Sysname> debugging dot1x packet

*May  1 12:56:27:934 2000 Sysname 8021X/7/PACKET:Port:GigabitEthernet1/1, Received an EAPOL packet.

*May  1 12:56:27:944 2000 Sysname 8021X/7/PACKET:Port:GigabitEthernet1/1, Packet type: EAPOL-START.

// GigabitEthernet 1/1 received an EAPOL-Start frame.

*May  1 12:56:27:954 2000 Sysname 8021X/7/PACKET:Port:GigabitEthernet1/1, End processing the packet received.

---Verbose information of the packet---
Destination Mac Address: ffff-ffff-ffff
Source Mac Address: 0015-e947-e45b
Mac Frame Type: 888e.
Packet Type: 1.
Packet Length: 0.

// GigabitEthernet 1/1 received an authentication request.


---Verbose information of the packet---
Destination Mac Address: 0015-e947-e45b
Source Mac Address: 00e0-fc01-5502
Mac Frame Type: 888e.
Packet Type: 0.
Packet Length: 5.

-----Packet Body-----
Code: 1.
Identifier: 1.
Length: 5.

// GigabitEthernet 1/1 sent an authentication response.
AAA debugging commands

The output description tables in this document only contain fields and messages that require an explanation.

debugging aaa

Use `debugging aaa` to enable AAA debugging.
Use `undo debugging aaa` to disable AAA debugging.

Syntax

```
debugging aaa { all | error | event } service-type { ssl-vpn | super | wapi }
undo debugging aaa { all | error | event } service-type wapi { ssl-vpn | super | wapi }
```

Default

AAA debugging is disabled.

Views

User view

Default command level

1: Monitor level

Parameters

- `all`: Specifies all types of debugging for AAA.
- `error`: Specifies debugging for errors.
- `event`: Specifies debugging for events.
- `service-type`: Specifies debugging for the specified user service type.
- `ssl-vpn`: Specifies the service type of SSL VPN.
- `super`: Specifies the service type of user privilege level switching.
- `wapi`: Specifies the service type of WAPI.

Usage guidelines

Table 5 describes output fields and messages for the `debugging aaa error` command.

Table 5 Output from the debugging aaa error command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Failed to refresh RLT timer</td>
<td>Failed to refresh real-time accounting timer.</td>
</tr>
<tr>
<td>Authen-req:</td>
<td>Authentication request.</td>
</tr>
<tr>
<td>Authen-resp:</td>
<td>Authentication response.</td>
</tr>
<tr>
<td>Continue-req:</td>
<td>Continue request.</td>
</tr>
<tr>
<td>Author-req:</td>
<td>Authorization request.</td>
</tr>
</tbody>
</table>
Table 6 describes output fields and messages for the **debugging aaa event** command.

### Table 6 Output from the debugging aaa event command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Authen-req</td>
<td>Authentication request.</td>
</tr>
<tr>
<td>Authen-resp</td>
<td>Authentication response.</td>
</tr>
<tr>
<td>Continue-req</td>
<td>Continue request.</td>
</tr>
<tr>
<td>Continue-resp</td>
<td>Continue response.</td>
</tr>
<tr>
<td>Author-req</td>
<td>Authorization request.</td>
</tr>
<tr>
<td>Author-resp</td>
<td>Authorization response.</td>
</tr>
<tr>
<td>Acct-start-req</td>
<td>Accounting-start request.</td>
</tr>
<tr>
<td>Acct-start-resp</td>
<td>Accounting-start response.</td>
</tr>
<tr>
<td>Acct-stop-req</td>
<td>Accounting-stop request.</td>
</tr>
<tr>
<td>Acct-stop-resp</td>
<td>Accounting-stop response.</td>
</tr>
<tr>
<td>Acct-RLT-req</td>
<td>Real-time accounting request.</td>
</tr>
<tr>
<td>Release-Req</td>
<td>Release request.</td>
</tr>
</tbody>
</table>

### Examples

The output in the following examples was created when a WAPI client initiated an authentication request under the following conditions:

- On the wireless AC, a RADIUS scheme is configured for WAPI access.
- IMC is used as the authentication server. It has the same CA-assigned certificate as the certificate on the WAPI client.

# On the wireless AC, enable AAA event debugging.

```
<Sysname> debugging aaa event service-type wapi
```

```
*Apr 26 12:05:36:652 2000 Sysname AAA/7/Event:UserID=0xffffffff,Service-type=0x600
0 Authen-req (AccessHandle = 0, AccessID = 0)
```

// AAA received an authentication request from the access module.

```
*Apr 26 12:05:36:662 2000 Sysname AAA/7/Event:

AAA_UserName , 1, wapi
AAA_DomainName , 63, wapi
AAA_Service , 4, 2
```
// AAA attributes carried in the access authentication request.
*Apr 26 12:05:36:669 2000 Sysname AAA/7/Event:000FE2A0789000+

// AAA successfully dispatched the authentication request to the RADIUS module.
*Apr 26 12:05:36:670 2000 Sysname AAA/7/Event:UserID=0x0,Service-type=0x600b Authen-req preprocess successfully. (AccessID = 0)
*Apr 26 12:05:36:679 2000 Sysname AAA/7/Event:UserID=0x0,Service-type=0x600b Send message to Radius, ErrorCode = 0
*Apr 26 12:05:36:690 2000 Sysname AAA/7/Event:UserID=0x0,Service-type=0x600b Authen-req: Dispatched message successfully.

// AAA displayed user information.
*Apr 26 12:05:36:699 2000 Sysname AAA/7/Event:UserID=0x0,Service-type=0x600b

// AAA received a continue-authentication response with the AAA message type 3 from the protocol module.
*Apr 26 12:05:36:781 2000 Sysname AAA/7/Event:UserID=0x0,Service-type=0x600b Continue-resp (Req-ID = 1, AAAMsgType = 3)

// AAA displayed user information.
*Apr 26 12:05:36:800 2000 Sysname AAA/7/Event:UserID=0x0,Service-type=0x6000 Continue-req

// AAA received a continue-authentication request from the access module.
*Apr 26 12:05:36:810 2000 Sysname AAA/7/Event:UserID=0x0,Service-type=0x600b

// AAA displayed user information.
*Apr 26 12:05:36:820 2000 Sysname AAA/7/Event:
AAA user information and AAA attributes carried in the request.

*Apr 26 12:05:36:840 2000 Sysname AAA/7/Event:UserID=0x0,Service-type=0x600b Send message to Radius, ErrorCode = 0

*Apr 26 12:05:36:861 2000 Sysname AAA/7/Event:UserID=0x0,Service-type=0x600b Continue-req: Dispatched message successfully.

// AAA successfully dispatched the continue-authentication request to the RADIUS module and displayed AAA user information.

*Apr 26 12:05:36:881 2000 Sysname AAA/7/Event:UserID=0x0,Service-type=0x600b Continue-resp: Send message to access.

// AAA sent a continue-authentication response to the access module.

*Apr 26 12:05:37:154 2000 Sysname AAA/7/Event:UserID=0x0,Service-type=0x600b Authen-resp: (Req-ID = 2, AAAMsgType = 4)

// AAA received an authentication response with the AAA message type 4 from the RADIUS module.

*Apr 26 12:05:36:972 2000 Sysname AAA/7/Event:

// AAA displayed attributes and user information carried in the response.

*Apr 26 12:05:37:154 2000 Sysname AAA/7/Event: UserID=0x0, Service-type=0x600b Authen-resp: Send message to access.

// AAA sent an authentication response to the access module.
// AAA received an authorization request from the access module and displayed AAA attributes and user information carried in the request.

// AAA required special flags because of the unauthorized RADIUS module.

// AAA successfully sent an authorization response to the access module.

// AAA displayed user information.
// AAA received an accounting-start request from the access module and displayed AAA attributes and user information carried in the request.

*Apr 26 12:05:37:83 2000 Sysname AAA/7/Event:UserID=0x0,Service-type=0x600b
Access-Handle= 0, Access-UserID= 0
Cur-AAA-Req = 6, Cur-AAA-State= 4
If-Replied = 0, Cur-Req-ID = 0

// AAA successfully dispatched the accounting request to the RADIUS module and displayed AAA user information.

*Apr 26 12:05:37:275 2000 Sysname AAA/7/Event:UserID=0x0,Service-type=0x600b
Acct-start-resp (Req-ID = 3, AAAMsgType = 11)

// AAA received an accounting-start response with the AAA message type 11 from the RADIUS module.

*Apr 26 12:05:37:285 2000 Sysname AAA/7/Event:
[AAA_UserMsg , 56, 341D79303430343420696E666F207930343034+
[AAA_FailCode , 68, 0]

// AAA displayed attributes and user information carried in the response.

*Apr 26 12:05:37:315 2000 Sysname AAA/7/Event:UserID=0x0,Service-type=0x600b Succeed to create online user.
// AAA created an online user.
*Apr 26 12:05:37:325 2000 Sysname AAA/7/Event:UserID=0x0,Service-type=0x600b
Acct-start-resp: Send message to access.

// AAA sent an accounting response to the access module.

# On the wireless AC, enable AAA error debugging.
<Sysname> debugging aaa error service-type wapi
*Nov 27 20:58:59:912 2007 Sysname AAA/7/Error:UserID=0x18,Service-type=0x600b Authen-req
was rejected because the last one is being processing.
**RADIUS debugging commands**

Some information in this chapter is device type specific. Devices in this chapter are categorized depending on their IRF capability and support for interface cards that use independent processors for forwarding traffic, as shown in Table 7.

**Table 7 Device types**

<table>
<thead>
<tr>
<th>Device type</th>
<th>Interface cards with on-card processors</th>
<th>IRF capability</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Distributed devices</td>
<td>Yes</td>
<td>No</td>
<td>HP 6600 routers (except for 6602)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Yes (in standalone mode)</td>
<td>HP 12500 switches HP 10500 switches</td>
</tr>
<tr>
<td>Distributed IRF devices</td>
<td>Yes</td>
<td>Yes (in IRF mode)</td>
<td>HP 12500 switches HP 10500 switches</td>
</tr>
<tr>
<td>Centralized devices</td>
<td>No</td>
<td>No</td>
<td>HP MSR routers HP 6602 router</td>
</tr>
<tr>
<td>Centralized IRF devices</td>
<td>No</td>
<td>Yes</td>
<td>HP 5800 switches HP 5500 switches</td>
</tr>
</tbody>
</table>

The output description tables in this document only contain fields and messages that require an explanation.

**debugging radius**

Use `debugging radius packet` to enable RADIUS debugging.

Use `undo debugging radius packet` to disable RADIUS debugging.

**Syntax**

Centralized devices:

```plaintext
debugging radius packet [ acl acl-number | user username ]
undo debugging radius packet
```

Distributed devices/centralized IRF devices:

```plaintext
debugging radius packet [ acl acl-number | user username ] [ slot slot-number ]
undo debugging radius packet [ slot slot-number ]
```

Distributed IRF devices:

```plaintext
designing radius packet [ acl acl-number | user username ] [ chassis chassis-number slot slot-number ]
undo debugging radius packet [ chassis chassis-number slot slot-number ]
```

**Default**

RADIUS debugging is disabled.
Views

User view

Default command level
1: Monitor level

Parameters

**packet**: Specifies debugging for packets.

**acl acl-number**: Specifies the ACL for matching RADIUS packets to be used for debugging. The acl-number argument is in the range 2000 to 3999. If this argument is configured multiple times, the most recent setting takes effect. Only the source IP address information in the ACL rules is used for matching packets.

**user username**: Specifies the username string for matching online users’ usernames. The user-name argument is a case-sensitive string of 1 to 80 characters.

**slot slot-number**: Specifies debugging for the card in a slot by the slot number. (Distributed devices.)

**slot slot-number**: Specifies debugging for a centralized IRF member device. The slot-number argument represents the member ID of the device. (Centralized IRF devices.)

**chassis chassis-number slot slot-number**: Specifies debugging for a card in an IRF member device. The chassis-number argument is the ID of a member device of the current IRF fabric. The slot-number argument specifies the slot number of the card. (Distributed IRF devices.)

Usage guidelines

Table 8 describes output fields and messages for the **debugging radius packet** command.

### Table 8 Output from the debugging radius packet command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Recv MSG</td>
<td>RADIUS received a message.</td>
</tr>
<tr>
<td>MsgType = msg type</td>
<td>Type of the received message.</td>
</tr>
<tr>
<td>Index = index</td>
<td>User index.</td>
</tr>
<tr>
<td>ulParam3 = address</td>
<td>Address carried in the message.</td>
</tr>
<tr>
<td>Send attribute list</td>
<td>Attribute list of the packet.</td>
</tr>
<tr>
<td>User-name</td>
<td>Username.</td>
</tr>
<tr>
<td>Password</td>
<td>User password.</td>
</tr>
<tr>
<td>Service-Type</td>
<td>Authentication service type for access users.</td>
</tr>
<tr>
<td>Framed-Protocol</td>
<td>Link layer protocol.</td>
</tr>
<tr>
<td>Framed-Address</td>
<td>Network address of user assigned by NAS.</td>
</tr>
<tr>
<td>NAS-Startup-Timestamp</td>
<td>Duration from NAS startup to current time.</td>
</tr>
<tr>
<td>RetryTimes = [retry-number]</td>
<td>Number of transmission attempts.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Code = [code-number]</td>
<td>Packet type identified by the code number:</td>
</tr>
<tr>
<td></td>
<td>• 1 — Access-request.</td>
</tr>
<tr>
<td></td>
<td>• 2 — Access-accept.</td>
</tr>
<tr>
<td></td>
<td>• 3 — Access-reject.</td>
</tr>
<tr>
<td></td>
<td>• 4 — Accounting-request.</td>
</tr>
<tr>
<td></td>
<td>• 5 — Accounting-response.</td>
</tr>
<tr>
<td></td>
<td>• 11 — Access-challenge.</td>
</tr>
<tr>
<td>State</td>
<td>State information when CHAP authentication was operating between the supplicant and server.</td>
</tr>
<tr>
<td>Termination-Action</td>
<td>The NAS termination action after the service ended.</td>
</tr>
<tr>
<td>Remanent_Volume</td>
<td>Remaining traffic that was allowed for the session.</td>
</tr>
<tr>
<td>Acct_Interim_Interval</td>
<td>Real-time accounting interval.</td>
</tr>
<tr>
<td>Server_String</td>
<td>Display information sent from the server to the supplicant.</td>
</tr>
<tr>
<td>Acct_Status-Type</td>
<td>Accounting request status.</td>
</tr>
<tr>
<td>Acct-Authentic</td>
<td>Authentication method.</td>
</tr>
<tr>
<td>Acct-Session-Id</td>
<td>Accounting session ID.</td>
</tr>
<tr>
<td>AAAID</td>
<td>AAA user resource ID.</td>
</tr>
<tr>
<td>Req-ID</td>
<td>Request ID (authentication, authorization, or accounting request).</td>
</tr>
</tbody>
</table>
| Error: Failed to get AAA-attr attribute | RADIUS failed to get AAA attributes.  
The AAA attribute includes:  |
|                    | • AAA_AcctType — Accounting type.                                            |
|                    | • AAA_PortVlanID — Port VLAN ID.                                             |
|                    | • AAA_AuthType — Authentication type.                                        |
|                    | • AAA_DomainName — Domain name.                                              |
|                    | • AAA_MacAddr — MAC address.                                                 |
| Event: No AAA-attr available | No AAA attributes available.  
The AAA attribute includes:  |
|                    | • AAA_IsStopAcctBufferSend Attribute — Whether or not the stop-accounting packets are buffered. |
|                    | • AAA_TerminateCause — Service termination cause.                            |

**Examples**

```
# Enable RADIUS packet debugging. Output similar to the following example is generated when a user initiates a portal authentication process through the IE browser under the following conditions:
  • Portal authentication is enabled on the device.
  • A RADIUS authentication scheme is configured for the user.

<Sysname> debugging radius packet
*Dec 6 15:36:38:956 2006 Sysname RDS/7/DEBUG:Recv MSG, [MsgType=Normal auth request Index = 5, ulParam3=109364420]
// RADIUS received an authentication request with the user index 5 and address 109364420.
*Dec 6 15:36:38:956 2006 Sysname RDS/7/DEBUG:NAS name is too long, cannot send Connect_port attribute
```

23
The connect_port attribute could not be transmitted because of a long NAS name.

*Dec 6 15:36:38:956 2006 Sysname RDS/7/DEBUG:Send attribute list:

[1] User-name                   [12] [yangliping]
[2] Password                    [18] [04E3336BFA6E7E8AADBC6B4687AB2DF1]
[32] NAS-Identifier              [17] [Sysname]
[87] NAS_Port_Id                 [18] [0100001000000002]

*Dec 6 15:36:39:03 2006 Sysname RDS/7/DEBUG:

[RADIUS displayed the attribute list in the authentication request.

*Dec 6 15:36:39:03 2006 Sysname RDS/7/DEBUG:Send: IP=[192.168.0.111], UserIndex=[5], ID=[0], RetryTimes=[0], Code=[1], Length=[192]

[RADIUS displayed the contents of the sent access-request message.

*Dec 6 15:36:39:19 2006 Sysname RDS/7/DEBUG:Send Raw Packet is:

[RADIUS displayed the contents of the sent raw packet.

*Dec 6 15:36:39:54 2006 Sysname RDS/7/DEBUG:Recv MSG,[MsgType=PKT response Index = 199, ulParam3=109358916]

[RADIUS received a response message with the user index 199 and address 109358916.

*Dec 6 15:36:39:54 2006 Sysname RDS/7/DEBUG:Receive Raw Packet is:
// RADIUS displayed the contents of the received raw packet.
*Dec 6 15:36:39:85 2006 Sysname
RDS/7/DEBUG:Receive:IP=[192.168.0.111],Code=[2],Length=[199]

// RADIUS received a 199-byte access-accept message with the source IP address of 192.168.0.111.
*Dec 6 15:36:39:85 2006 Sysname RDS/7/DEBUG:
[6 Service-Type | 6 ] [2]
[24 State | 10 ] [3267447735337676]
[29 Termination-Action | 6 ] [0]
[hw-15 Remanent_Volume | 6 ] [10485760]
[85 Acct_Interim_Interval | 6 ] [600]
[hw-26 Connect_ID | 6 ] [5]

// RADIUS displayed the attribute list in the access-accept message.
*Dec 6 15:36:39:101 2006 Sysname RDS/7/DEBUG:Recv MSG,[MsgType=PKT response Index = 199, ulParam3=109357700]

// RADIUS received a response message with the user index 199 and address 109357700.
*Dec 6 15:36:39:101 2006 Sysname RDS/7/DEBUG:Receive Raw Packet is:
*Dec 6 15:36:39:101 2006 Sysname RDS/7/DEBUG:

// RADIUS displayed the raw contents of the received response.
*Dec 6 15:36:39:336 2006 Sysname RDS/7/DEBUG:Recv MSG,[MsgType=Account request Index = 5, ulParam3=0]

// RADIUS received an accounting request message with the user index 5 and address 0.
*Dec 6 15:36:39:336 2006 Sysname RDS/7/DEBUG:Send attribute list:
*Dec  6 15:36:39:336 2006 Sysname RDS/7/DEBUG:
[1 User-name                   ] [12] [yangliping]
[32 NAS-Identifier              ] [17] [Sysname]
[5  NAS-Port                    ] [6 ] [20975618]
[87 NAS_Port_Id                 ] [18] [0100001000000002]
[61 NAS-Port-Type               ] [6 ] [15]
[31 Caller-ID                   ] [16] [303030302D303030302D30303030]

*Dec  6 15:36:39:367 2006 Sysname RDS/7/DEBUG:
[40 Acct-Status-Type            ] [6 ] [1]
[45 Acct-Authentic              ] [6 ] [1]
[44 Acct-Session-Id             ] [14] [106110615361]
[8  Framed-Address              ] [6 ] [19.19.19.2]
[4  NAS-IP-Address              ] [6 ] [192.168.0.55]
[55 Event-Timestamp             ] [6 ] [1165419399]

*Dec  6 15:36:39:383 2006 Sysname RDS/7/DEBUG:
[hw-26 Connect_ID               ] [6 ] [5]
[hw-1  Input_Peak_Rate          ] [6 ] [0]
[hw-2  Input_Average_Rate       ] [6 ] [0]
[hw-4  Output_Peak_Rate         ] [6 ] [0]
[hw-5  Output_Average_Rate      ] [6 ] [0]
[hw-22 Priority                 ] [6 ] [0]

*Dec  6 15:36:39:399 2006 Sysname RDS/7/DEBUG:
[hw-60 Ip-Host-Addr             ] [30] [19.19.19.2 00:00:00:00:00:00]

// RADIUS displayed the attribute list in the accounting request.

*Dec  6 15:36:39:414 2006 Sysname RDS/7/DEBUG:Send: IP=[192.168.0.111],
UserIndex=[5], ID=[0], RetryTimes=[0], Code=[4], Length=[211]

// RADIUS sent an accounting request message.

*Dec  6 15:36:39:414 2006 Sysname RDS/7/DEBUG:Send Raw Packet is:
*Dec  6 15:36:39:430 2006 Sysname RDS/7/DEBUG:
04 00 00 d3 03 d0 2e ac 8c c0 5a 0a 15 59 db 07
da 7c d9 62 01 0c 79 61 6e 67 6c 69 70 69 6e 67
20 11 51 75 69 64 77 61 79 2d 43 6f 6d 77 61 72
65 05 06 01 40 10 02 57 12 30 31 30 30 30 30 30
30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30
06 00 00 00 01 2d 06 00 00 00 01 2c 0e 31 30 36
31 31 30 36 31 35 33 36 31 08 06 13 13 13 02 04
06 c0 a8 00 37 37 06 45 76 e3 87 1a 48 00 00 07
db 1a 06 00 00 00 05 01 06 00 00 00 00 02 06 00
00 00 00 04 06 00 00 00 00 00 05 06 00 00 00 00 16
06 00 00 00 00 03 c1 le 31 39 2e 31 39 2e 31 39 2e
32 20 30 30 3a 30 30 3a 30 30 3a 30 30 3a 30 30
3a 30 30

// RADIUS displayed the raw contents of the sent accounting request message.

*Dec  6 15:36:39:472 2006 Sysname RDS/7/DEBUG:Recv MSG,
[MsgType=PKT response Index = 95,
ulParam3=109357572]

// RADIUS received a response message with the user index 95 and address 109357572.
*Dec 6 15:36:39:472 2006 Sysname RDS/7/DEBUG:Receive Raw Packet is:

*Dec 6 15:36:39:487 2006 Sysname RDS/7/DEBUG:
05 00 00 5f 27 5a b4 d3 4b b6 1e 19 b6 89 1b 30
40 42 2f 8e 1a 0c 00 00 07 db 1a 06 00 00 00 05
1a 3f 00 00 07 db 3d 39 33 21 c4 fa b5 c4 d5 ca
ba c5 d3 e0 b6 ee ca c7 a3 ba 39 39 36 34 30 34
32 2e 34 39 20 d4 aa a1 a3 3d 0a 32 67 44 77 35
33 76 76 3e 06 6f 00 a8 c0 3f 06 3b 23 00 00

// RADIUS displayed the raw contents of the received response.


// RADIUS received a 95-byte accounting response message with the source IP address of 192.168.0.111.

*Dec 6 15:36:39:503 2006 Sysname RDS/7/DEBUG:Receive Raw Packet is:

*Dec 6 15:36:39:503 2006 Sysname RDS/7/DEBUG:
05 00 00 5f 27 5a b4 d3 4b b6 1e 19 b6 89 1b 30
40 42 2f 8e 1a 0c 00 00 07 db 1a 06 00 00 00 05
1a 3f 00 00 07 db 3d 39 33 21 c4 fa b5 c4 d5 ca
ba c5 d3 e0 b6 ee ca c7 a3 ba 39 39 36 34 30 34
32 2e 34 39 20 d4 aa a1 a3 3d 0a 32 67 44 77 35
33 76 76 3e 06 6f 00 a8 c0 3f 06 3b 23 00 00

// RADIUS received a response message with the user index 95 and address 109359492.
HWTACACS debugging commands

Some information in this chapter is device type specific. Devices in this chapter are categorized depending on their IRF capability and support for interface cards that use independent processors for forwarding traffic, as shown in Table 9.

Table 9 Device types

<table>
<thead>
<tr>
<th>Device type</th>
<th>Interface cards with on-card processors</th>
<th>IRF capability</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Distributed devices</td>
<td>Yes</td>
<td>No</td>
<td>HP 6600 routers (except for 6602)</td>
</tr>
</tbody>
</table>
|                    |                                          | Yes (in standalone mode) | HP 12500 switches  
|                    |                                          |                 | HP 10500 switches                          |
| Distributed IRF devices | Yes                                     | Yes (in IRF mode) | HP 12500 switches 
|                    |                                          |                 | HP 10500 switches                          |
| Centralized devices | No                                       | No             | HP MSR routers                                |
|                    |                                          |                 | HP 6602 router                                |
| Centralized IRF devices | No                                      | Yes            | HP 5800 switches                              |
|                    |                                          |                 | HP 5500 switches                              |

The output description tables in this document only contain fields and messages that require an explanation.

debugging hwtacacs

Use **debugging hwtacacs** to enable HWTACACS debugging.

Use **undo debugging hwtacacs** to disable HWTACACS debugging.

Syntax

Centralized devices:

```
debugging hwtacacs { all | error | event | message | receive-packet | send-packet }
dundo debugging hwtacacs { all | error | event | message | receive-packet | send-packet }
```

Distributed devices/centralized IRF devices:

```
debugging hwtacacs { all | error | event | message | receive-packet | send-packet } [ slot slot-number ]
dundo debugging hwtacacs { all | error | event | message | receive-packet | send-packet } [ slot slot-number ]
```

Distributed IRF devices:

```
debugging hwtacacs { all | error | event | message | receive-packet | send-packet } [ chassis chassis-number slot slot-number ]
dundo debugging hwtacacs { all | error | event | message | receive-packet | send-packet } [ chassis chassis-number slot slot-number ]
```
Default

HWTACACS debugging is disabled.

Views

User view

Default command level

1: Monitor level

Parameters

all: Specifies all types of debugging for HWTACACS.
error: Specifies debugging for errors.
event: Specifies debugging for events.
message: Specifies debugging for messages.
receive-packet: Specifies debugging for received packets.
send-packet: Specifies debugging for sent packets.
slot slot-number: Specifies debugging for the card in a slot by the slot number. (Distributed devices.)
slot slot-number: Specifies debugging a centralized IRF member. The slot-number argument represents the member ID of the device. (Centralized IRF devices.)
chassis chassis-number slot slot-number: Specifies debugging for a card in an IRF member device. The chassis-number argument is the ID of a member device of the current IRF fabric. The slot-number argument specifies the slot number of the card. (Distributed IRF devices.)

Usage guidelines

Table 10 describes output fields and messages for the `debugging hwtacacs error` command.

Table 10 Output from the debugging hwtacacs error command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>(TAC_ACCT_HandleMsg): AAA send a NULL message to TAC_ACCT</td>
<td>Parameter error in TAC_ACCT_HandleMsg.</td>
</tr>
<tr>
<td>Tac malloc accounting node fail</td>
<td>HWTACACS failed to apply for the accounting node buffer.</td>
</tr>
<tr>
<td>(TAC_ACCT_EncapNode): Add node to AcctSendQ fail</td>
<td>HWTACACS failed to add the accounting node.</td>
</tr>
<tr>
<td>(TAC_ACCT_MakeStartPacket): VOS_Malloc temp attr fail</td>
<td>HWTACACS failed to allocate temporary attribute memory.</td>
</tr>
<tr>
<td>VOS_Malloc packet fail</td>
<td>HWTACACS failed to apply for the packet buffer.</td>
</tr>
<tr>
<td>(TAC_ACCT_EncapMsg): VOS_Malloc message fail</td>
<td>HWTACACS failed to apply for the buffer.</td>
</tr>
<tr>
<td>TAC_AUTHEN_HandleMsg:AAA to HWTAC AUTHEN is NULL”</td>
<td>HWTACACS received a null authentication packet from the AAA module.</td>
</tr>
<tr>
<td>TAC_AUTHEN_HandleMsg:HWTAC AUTHEN HandleMsg UserID &gt; AAA_MAX_USER</td>
<td>The user ID in the authentication packet exceeded the maximum number of users.</td>
</tr>
<tr>
<td>TAC_AUTHEN_EncapNode:no useful continue server</td>
<td>The server could not process consequent continue packets.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td>TAC_AUTHEN_RecvCallBack: tacacs packet is null or userid&gt;MAX</td>
<td>Invalid TAC_AUTHEN_RecvCallBack.</td>
</tr>
<tr>
<td>TAC_AUTHEN_RecvCallBack: replay node isn't in sendlist</td>
<td>The received node was not in the to-be-sent list.</td>
</tr>
<tr>
<td>Failed to create continue authentication request</td>
<td>HWTACACS failed to create continue-authentication requests.</td>
</tr>
<tr>
<td>Can not find a valid server when send authen packet fail</td>
<td>HWTACACS failed to send authentication packet because no available server was found.</td>
</tr>
<tr>
<td>TAC_AUTHOR_HandleMsg: aaa to hwtac author is null</td>
<td>HWTACACS received a null authorization packet from the AAA module.</td>
</tr>
<tr>
<td>TAC_AUTHOR_EncapNode: system resource isn't enough</td>
<td>Insufficient system resources.</td>
</tr>
<tr>
<td>TAC_AUTHOR_EncapNode: no useful hwtac server</td>
<td>No HWTACACS server available.</td>
</tr>
<tr>
<td>TAC_AUTHOR_RecvCallBack: hwtac author packet is null</td>
<td>HWTACACS received a null authorization packet with an incorrect parameter.</td>
</tr>
<tr>
<td>TAC_AUTHOR_NoReplyCallBack: userid is error</td>
<td>HWTACACS could not receive a response to the authorization request due to a user ID error.</td>
</tr>
<tr>
<td>TAC_AUTHOR_NoReplyCallBack: no the hwtac author node</td>
<td>HWTACACS could not receive a response to the authorization request because the user node could not be found.</td>
</tr>
<tr>
<td>TAC_AUTHOR_NoReplyCallBack: no useful hwtac server</td>
<td>No HWTACACS server available.</td>
</tr>
<tr>
<td>TAC_AUTHOR_CheckReplyPacket: hwtac author packet is null</td>
<td>HWTACACS received a null authorization packet.</td>
</tr>
<tr>
<td>Tac Receive nocallback-verify value error</td>
<td>HWTACACS received an incorrect nocallback-verify value.</td>
</tr>
<tr>
<td>Tac Receive nohangup value error</td>
<td>HWTACACS received an incorrect nohangup value.</td>
</tr>
<tr>
<td>Tac Receive unknown attribute</td>
<td>HWTACACS received an invalid attribute.</td>
</tr>
<tr>
<td>Tac Receive IPAddress value error</td>
<td>HWTACACS received an incorrect IP address value.</td>
</tr>
<tr>
<td>TAC_AUTHOR_EncapMsg: system resource isn't enough</td>
<td>HWTACACS failed to apply for the memory because of insufficient system resources.</td>
</tr>
<tr>
<td>TAC_AUTHOR_EncapMsg: hwtac author node is error</td>
<td>Authorization node error.</td>
</tr>
<tr>
<td>TAC_AUTHOR_EncapMsg: error authortype</td>
<td>Authorization type error.</td>
</tr>
<tr>
<td>Tac check authen request packet error</td>
<td>Authentication request check error.</td>
</tr>
<tr>
<td>Tac check authen continue packet error</td>
<td>Authentication continue packet check error.</td>
</tr>
<tr>
<td>Tac check author request packet error</td>
<td>Authorization request check error.</td>
</tr>
<tr>
<td>Tac check account request packet error</td>
<td>Accounting request check error.</td>
</tr>
<tr>
<td>Error: hwtacacs, find nonzero session which does not exist in TAC_GetSession</td>
<td>HWTACACS failed to search for a session by session-id due to the absence of the session identified by the session-id.</td>
</tr>
<tr>
<td>TAC_init result</td>
<td>The HWTACACS module initialization error.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td>Error: hwtacacs, switch type error in TAC_HandleNode</td>
<td>Type error.</td>
</tr>
<tr>
<td>TAC get session fail</td>
<td>HWTACACS failed to get the session.</td>
</tr>
<tr>
<td>Error: hwtacacs, the packet to be sent is too big in TAC_HandleReqMsg</td>
<td>The packet to be sent was too long.</td>
</tr>
<tr>
<td>Error: hwtacacs, session status error in TAC_GetSession</td>
<td>Session status error in the session to be obtained.</td>
</tr>
<tr>
<td>Warning: hwtacacs, used up all sessions before process a require</td>
<td>HWTACACS could not generate new session ID because the upper limit was reached.</td>
</tr>
<tr>
<td>Error: hwtacacs set source ip of socket failure</td>
<td>HWTACACS failed to set the source IP address of the socket.</td>
</tr>
<tr>
<td>Error: hwtacacs set asynchronous attribute of socket failure</td>
<td>HWTACACS failed to set the asynchronous attribute of the socket.</td>
</tr>
<tr>
<td>Tac connect to server failure</td>
<td>HWTACACS failed to establish a socket connection to the server.</td>
</tr>
<tr>
<td>Error: hwtacacs has a linker error in TAC_TXRXFreeSession</td>
<td>An error occurred when a session was released.</td>
</tr>
<tr>
<td>Tac receive ASYN CLOSE message, discard it</td>
<td>HWTACACS received a socket close message and discarded it.</td>
</tr>
<tr>
<td>Tac receive msg message, but cannot find according session</td>
<td>HWTACACS received a message msg, but could not find the corresponding session.</td>
</tr>
<tr>
<td>Tac Receive wrong aysn message</td>
<td>HWTACACS received an incorrect asynchronous socket message.</td>
</tr>
<tr>
<td>hwtacacs failure when process server’s response! session id: session-id user id: user-id server ip: ip-address</td>
<td>An error occurred when HWTACACS processed a server response.</td>
</tr>
<tr>
<td>Error: hwtacacs, receive message failure in TAC_HandleResp</td>
<td>Invalid length of the received packet.</td>
</tr>
<tr>
<td>Error: hwtacacs, receive server response but the user has already not existed, handle it as timeout</td>
<td>HWTACACS received a server response, but the user did not exist.</td>
</tr>
</tbody>
</table>

Table 11 describes output fields and messages for the **debugging hwtacacs event** command.

### Table 11 Output from the debugging hwtacacs event command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| transmit flag: flag-number | Packet transmit flag:  
  • 1—Sent.  
  • 2—Received.  
  • 3—No response. |
| server flag: flag-number | Server flag:  
  • 0—Authentication server.  
  • 1—Authorization server.  
  • 2—Accounting server. |
Table 12 describes output fields and messages for the **debugging hwtacacs message** command.

Table 12 Output from the debugging hwtacacs message command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TAC_MESSAGE for AAA-&gt;TAC</td>
<td>HWTACACS message sent from the AAA module to the HWTACACS module.</td>
</tr>
<tr>
<td>TAC_MESSAGE for TAC-&gt;AAA</td>
<td>HWTACACS message sent from the HWTACACS module to the AAA module.</td>
</tr>
<tr>
<td>PacketType=packet-type</td>
<td>Packet type:</td>
</tr>
<tr>
<td></td>
<td>• 1 — Authentication.</td>
</tr>
<tr>
<td></td>
<td>• 2 — Authorization.</td>
</tr>
<tr>
<td></td>
<td>• 3 — Accounting.</td>
</tr>
<tr>
<td>AuthenType=authen-type</td>
<td>Authentication type:</td>
</tr>
<tr>
<td></td>
<td>• 1 — ASCII.</td>
</tr>
<tr>
<td></td>
<td>• 2 — PAP.</td>
</tr>
<tr>
<td></td>
<td>• 3 — CHAP.</td>
</tr>
<tr>
<td></td>
<td>• 4 — ARAP.</td>
</tr>
<tr>
<td></td>
<td>• 5 — MSCHAP.</td>
</tr>
<tr>
<td>AuthenService=service-type</td>
<td>Service type:</td>
</tr>
<tr>
<td></td>
<td>• 0 — None.</td>
</tr>
<tr>
<td></td>
<td>• 1 — Login.</td>
</tr>
<tr>
<td></td>
<td>• 2 — Password authentication.</td>
</tr>
<tr>
<td></td>
<td>• 3 — PPP.</td>
</tr>
<tr>
<td></td>
<td>• 4 — ARAP.</td>
</tr>
<tr>
<td>PrivLevel=privlevel</td>
<td>User privilege:</td>
</tr>
<tr>
<td></td>
<td>• 0 — Visit.</td>
</tr>
<tr>
<td></td>
<td>• 1 — Monitor.</td>
</tr>
<tr>
<td></td>
<td>• 2 — Config.</td>
</tr>
<tr>
<td></td>
<td>• 3 — Manage.</td>
</tr>
<tr>
<td>RemAddress=ip-address</td>
<td>User IP address.</td>
</tr>
<tr>
<td>UserMsg=reply-msg</td>
<td>Character string entered by the user.</td>
</tr>
<tr>
<td>DataMsg=data-msg</td>
<td>CHAP authentication data.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>AuthorType=author-type</td>
<td>Authorization type:</td>
</tr>
<tr>
<td></td>
<td>• 4—EXEC user authorization request.</td>
</tr>
<tr>
<td></td>
<td>• 5—VPDN user.</td>
</tr>
<tr>
<td></td>
<td>• 6—PPP user.</td>
</tr>
<tr>
<td></td>
<td>• 7—VLAN user.</td>
</tr>
<tr>
<td>AuthenMethod = method</td>
<td>Authentication method:</td>
</tr>
<tr>
<td></td>
<td>• 0—Not set.</td>
</tr>
<tr>
<td></td>
<td>• 1—None.</td>
</tr>
<tr>
<td></td>
<td>• 2—KRB5.</td>
</tr>
<tr>
<td></td>
<td>• 3—Line.</td>
</tr>
<tr>
<td></td>
<td>• 4—Enable.</td>
</tr>
<tr>
<td></td>
<td>• 5—Local.</td>
</tr>
<tr>
<td></td>
<td>• 6—Tacacsplus.</td>
</tr>
<tr>
<td>ArgNum = number</td>
<td>Number of authorization request attributes.</td>
</tr>
<tr>
<td>Service=service</td>
<td>Service type:</td>
</tr>
<tr>
<td></td>
<td>• 0—None.</td>
</tr>
<tr>
<td></td>
<td>• 1—Login.</td>
</tr>
<tr>
<td></td>
<td>• 2—Password authentication.</td>
</tr>
<tr>
<td></td>
<td>• 3—PPP.</td>
</tr>
<tr>
<td></td>
<td>• 4—ARAP.</td>
</tr>
<tr>
<td>Protocol= protocol-type</td>
<td>Protocol type: IP, VPDN, or LCP.</td>
</tr>
<tr>
<td>Acl= acl-number</td>
<td>ACL number.</td>
</tr>
<tr>
<td>Timeout=time</td>
<td>Timeout time.</td>
</tr>
<tr>
<td>AutoExec=command</td>
<td>Automatic command after user login.</td>
</tr>
<tr>
<td>Address = ip-address</td>
<td>IP address of the user.</td>
</tr>
<tr>
<td>AddressPool</td>
<td>Address pool.</td>
</tr>
<tr>
<td>CallBackVerify=</td>
<td>No authentication after dialup.</td>
</tr>
<tr>
<td>Callbackdialstring=</td>
<td>Whether or not callback was enabled.</td>
</tr>
<tr>
<td>DNSAddress=</td>
<td>Assigned DNS IP address corresponding to the PPP link.</td>
</tr>
<tr>
<td>sVpdnGroupNumber=number</td>
<td>VPDN group of the tunnel.</td>
</tr>
<tr>
<td>TunnelType=tunnel-type</td>
<td>Tunnel type.</td>
</tr>
<tr>
<td>NO.%lu IPAddress=ip-address</td>
<td>IP address used to establish a tunnel.</td>
</tr>
<tr>
<td>TunnelID=tunnel-id</td>
<td>Tunnel ID.</td>
</tr>
<tr>
<td>SourceIP=ip-address</td>
<td>IP address of the accessing user.</td>
</tr>
<tr>
<td>HelloInterval=interval</td>
<td>L2TP hello packet transmission interval.</td>
</tr>
<tr>
<td>NoSessionTimeout=timer</td>
<td>HWTACACS cleared the corresponding tunnel when the timer timed out.</td>
</tr>
<tr>
<td>HiddenAVP=bool-value</td>
<td>Whether or not the L2TP hidden function was enabled.</td>
</tr>
<tr>
<td>TosReflect=bool-value</td>
<td>Whether or not HWTACACS mapped the TOS field of the original IP header when it created an L2TP external IP header.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>TunnelAuthen=bool-value</td>
<td>Whether or not tunnel authentication was enabled.</td>
</tr>
<tr>
<td>UDPChecksum=bool-value</td>
<td>Whether or not the L2TP UDP CheckSum function was enabled.</td>
</tr>
<tr>
<td>uclflag=flag-number</td>
<td>Authentication result flag.</td>
</tr>
<tr>
<td>ServerMsg=msg</td>
<td>Server message to login user.</td>
</tr>
<tr>
<td>Echo=number</td>
<td>Username and password echo flag.</td>
</tr>
<tr>
<td>MutilinkMax</td>
<td>Maximum number of aggregated links.</td>
</tr>
<tr>
<td>MutilinkCurrent</td>
<td>Current number of aggregated links.</td>
</tr>
<tr>
<td>PacketIn</td>
<td>Number of transmitted incoming packets upon an action.</td>
</tr>
<tr>
<td>ByteIn</td>
<td>Number of transmitted incoming bytes upon an action.</td>
</tr>
<tr>
<td>PacketOut</td>
<td>Number of transmitted outgoing packets upon an action.</td>
</tr>
<tr>
<td>ByteOut</td>
<td>Number of transmitted outgoing bytes upon an action.</td>
</tr>
<tr>
<td>ElapsedTime</td>
<td>Elapsed time since an action.</td>
</tr>
<tr>
<td>NasRxSpeed</td>
<td>Maximum receive speed.</td>
</tr>
<tr>
<td>NasTxSpeed</td>
<td>Maximum transmit speed.</td>
</tr>
<tr>
<td>AcctFlag =acct-flag</td>
<td>Accounting method:</td>
</tr>
<tr>
<td></td>
<td>• 1—Net (network).</td>
</tr>
<tr>
<td></td>
<td>• 2—Connet (connection).</td>
</tr>
<tr>
<td></td>
<td>• 3—Exec.</td>
</tr>
<tr>
<td></td>
<td>• 4—System.</td>
</tr>
<tr>
<td></td>
<td>• 5—Cmd.</td>
</tr>
<tr>
<td>AcctType =acct-type</td>
<td>Accounting type:</td>
</tr>
<tr>
<td></td>
<td>• 2—Start accounting.</td>
</tr>
<tr>
<td></td>
<td>• 4—End accounting.</td>
</tr>
<tr>
<td></td>
<td>• 8—Update accounting.</td>
</tr>
<tr>
<td>Status=status</td>
<td>Accounting result flag:</td>
</tr>
<tr>
<td></td>
<td>• 1—Success.</td>
</tr>
<tr>
<td></td>
<td>• 2—Failure.</td>
</tr>
<tr>
<td></td>
<td>• 3—No response.</td>
</tr>
<tr>
<td></td>
<td>• 4—Error.</td>
</tr>
</tbody>
</table>

Table 13 describes output fields and messages for the `debugging hwtacacs receive-packet` command.

**Table 13 Output from the debugging hwtacacs receive-packet command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Status</td>
<td>Authentication status.</td>
</tr>
<tr>
<td>Flag</td>
<td>Whether or not the entered username and password were displayed in the authentication response.</td>
</tr>
<tr>
<td>server_msg len</td>
<td>Length of the message sent from the server to the client.</td>
</tr>
<tr>
<td>data len</td>
<td>Length of the failure causes received from the server.</td>
</tr>
<tr>
<td>server_msg</td>
<td>Server message to the login user.</td>
</tr>
</tbody>
</table>
### Table 14: Output from the debugging hwtacacs send-packet command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>Failure causes returned from the server.</td>
</tr>
<tr>
<td>arg_cnt</td>
<td>Number of authorization attributes.</td>
</tr>
<tr>
<td>arg mm len</td>
<td>Length of the attribute pair.</td>
</tr>
<tr>
<td>arg</td>
<td>Attribute pair.</td>
</tr>
</tbody>
</table>

Table 14 describes output fields and messages for the **debugging hwtacacs send-packet** command.

**Examples**

The output in the following examples was created when a Telnet user logged into and out of the device configured with HWTACACS.
# Enable HWTACACS event debugging.

```bash
<Sysname> debugging hwtacacs event
```

*Dec 6 10:33:58:772 2006 Sysname TAC/7/Event: Create HWTACACS authentication request packet success

// HWTACACS created an authentication request.

*Dec 6 10:33:58:804 2006 Sysname TAC/7/Event:

```
  hwtacacs create new session:
```

// HWTACACS created a new session.

session id: 41, user id: 0, server ip: 192.168.31.178


*Dec 6 10:33:58:912 2006 Sysname TAC/7/Event:

```
  hwtacacs packet sending success!
```

// HWTACACS successfully sent the authentication request.


// HWTACACS received an authentication response.


// HWTACACS successfully sent an authentication-continue message.

*Dec 6 10:33:59:130 2006 Sysname TAC/7/Event:statistic: transmit flag:1, server flag: 0,packet flag:0x1

// HWTACACS received an authentication response.

*Dec 6 10:33:59:130 2006 Sysname TAC/7/Event:

```
  hwtacacs session is deleted due to finishing session:
```

session id: 41, user id: 0, server ip: 192.168.31.178

// Packet exchange completed. HWTACACS ended and removed the session.

*Dec 6 10:33:59:238 2006 Sysname TAC/7/Event:

```
  hwtacacs create new session:
```

// HWTACACS created a new session.


*Dec 6 10:33:59:348 2006 Sysname TAC/7/Event:

```
  hwtacacs packet sending success!
```

// HWTACACS successfully sent an authorization request.

*Dec 6 10:33:59:456 2006 Sysname TAC/7/Event:statistic: transmit flag:2, server flag: 1,packet flag:0x1

// HWTACACS received an authorization response.

*Dec 6 10:33:59:456 2006 Sysname TAC/7/Event:

```
  hwtacacs session is deleted due to finishing session:
```

36
session id: 45, user id: 0, server ip: 192.168.31.178

// Packet exchange completed. HWTACACS ended and removed the session.

*Dec 6 10:33:59:566 2006 Sysname TAC/7/Event:
  hwtaucacs create new session:
  session id: 48, user id: 0, server ip: 192.168.31.178

// HWTACACS created a new session.

*Dec 6 10:33:59:674 2006 Sysname TAC/7/Event:
  hwtaucacs packet sending success!
  version:c0 type:3 sequence:1 flag:0 session id:48 length:63

// HWTACACS successfully sent an accounting request.

*Dec 6 10:33:59:782 2006 Sysname TAC/7/Event:statistic: transmit flag:2, server flag: 2, packet flag:0x0

// HWTACACS received an accounting response.

*Dec 6 10:33:59:782 2006 Sysname TAC/7/Event:
  hwtaucacs session is deleted due to finishing session:
  session id: 48, user id: 0, server ip: 192.168.31.178

// Packet exchange completed. HWTACACS ended and removed the session.

%Dec 6 10:33:59:782 2006 Sysname SHELL/4/LOGIN: lq@yang login from 192.168.31.56
%Dec 6 10:34:46:424 2006 Sysname SHELL/4/LOGOUT: lq@yang logout from 192.168.31.56

// A user logged in and logged out.

*Dec 6 10:34:46:522 2006 Sysname TAC/7/Event:
  hwtaucacs create new session:
  session id: 51, user id: 0, server ip: 192.168.31.178

*Dec 6 10:34:46:630 2006 Sysname TAC/7/Event:
  hwtaucacs packet sending success!
  version:c0 type:3 sequence:1 flag:0 session id:51 length:183

// HWTACACS successfully sent an accounting-stop request.

*Dec 6 10:34:46:738 2006 Sysname TAC/7/Event:statistic: transmit flag:2, server flag: 2, packet flag:0x0

// HWTACACS received an accounting response.

*Dec 6 10:34:46:738 2006 Sysname TAC/7/Event:
  hwtaucacs session is deleted due to finishing session:
  session id: 51, user id: 0, server ip: 192.168.31.178

// Packet exchange completed. HWTACACS ended and removed the session.

# Enable HWTACACS message debugging.
<Sysname> debugging hwtaucacs message
*Dec 6 10:38:14:282 2006 Sysname TAC/7/Event:
  TAC_MESSAGE for AAA->TAC:
*Dec 6 10:38:14:282 2006 Sysname TAC/7/Event:
UserID=2  PacketType=3  AuthenType=1  AuthenService=1  PrivLevel=0  Version=c0  TemplateNum=0
UserName=lq  PortName=vty0  RemAddress=192.168.31.56  UserMsg=  DataMsg=

*Dec 6 10:38:14:336 2006 Sysname TAC/7/Event:
TAC_MESSAGE for AAA->TAC:
*Dec 6 10:38:14:336 2006 Sysname TAC/7/Event:
UserID=2  PacketType=3  AuthenType=1  AuthenService=1  PrivLevel=0  Version=c0  TemplateNum=0
UserName=lq  PortName=vty0  RemAddress=192.168.31.56  UserMsg=  DataMsg=

// An HWTACACS message was sent from the AAA module to the HWTACACS module.
*Dec 6 10:38:14:662 2006 Sysname TAC/7/Event:
TAC_MESSAGE for TAC->AAA:
*Dec 6 10:38:14:662 2006 Sysname TAC/7/Event:
ulUserID=2  ucTACTemplateNO=0  ucflag=1  ServerMsg=
Echo=0

// An HWTACACS message was sent from the HWTACACS module to the AAA module.
*Dec 6 10:38:14:662 2006 Sysname TAC/7/Event:
TAC_MESSAGE for AAA->TAC:
*Dec 6 10:38:14:662 2006 Sysname TAC/7/Event:
UserID=2  AuthorType=4  AuthenMethod=6  AuthenType=1  AuthenService=1  PrivLevel=0  TemplateNum=0  ArgNum=2
UserName=lq  PortName=vty0  Service=shell  Protocol=cmd*  RemAddress=192.168.31.56

*Dec 6 10:38:14:772 2006 Sysname TAC/7/Event:
TAC_MESSAGE for AAA->TAC:
*Dec 6 10:38:14:772 2006 Sysname TAC/7/Event:
UserID=2  AuthorType=4  AuthenMethod=6  AuthenType=1  AuthenService=1  PrivLevel=0  TemplateNum=0  ArgNum=2
UserName=lq  PortName=vty0  Service=shell  Protocol=cmd*  RemAddress=192.168.31.56

// An HWTACACS message was sent from the AAA module to the HWTACACS module.
*Dec 6 10:38:14:988 2006 Sysname TAC/7/Event:
TAC_MESSAGE for TAC->AAA:
*Dec 6 10:38:14:988 2006 Sysname TAC/7/Event:
AuthorType=4  ServerMsg=  DataMsg=
Acl=0  Timeout=0  PrivLevel=1  NoHangup=0  AutoExec=

// An HWTACACS message was sent from the HWTACACS module to the AAA module.
*Dec 6 10:38:15:00 2006 Sysname TAC/7/Event:
TAC_MESSAGE for AAA->TAC:
*Dec 6 10:38:15:00 2006 Sysname TAC/7/Event:
UserID=2 Flag =2 TemplateNum =0
AuthenMethod=6 AuthenType=1 AuthenService=1
UserName=lq UserPort=vty0
UserAddress=192.168.31.56
AcctService=3 NetworkFlag=0
TaskID=0 TimeZone=0
Service=shell Protocol=
MutilinkMax=0 MutilinkCurrent=0
DisconnectCause=0 DisconnectCauseExtend=0
PacketIn=0 ByteIn=0
PacketOut=0 ByteOut=0
ElapsedTime=0
NasRxSpeed=0 NasTxSpeed=0
TunnelID=0 TunnelConnectionID=0
Command= Event= Reason=
Privlevel=1

// An HWTACACS message was sent from the AAA module to the HWTACACS module.

*Dec 6 10:38:15:98 2006 Sysname TAC/7/Event:
TAC_MESSAGE for AAA->TAC:
*Dec 6 10:38:15:98 2006 Sysname TAC/7/Event:
UserID=2 AcctFlag =3 AcctType =2
Status=1 GroupNum=0
ServerMsg= DataMsg=

// An HWTACACS message was sent from the HWTACACS module to the AAA module.

%Dec 6 10:38:15:316 2006 Sysname SHELL/4/LOGIN: lq@yang login from 192.168.31.56
%Dec 6 10:38:24:238 2006 Sysname SHELL/4/LOGOUT: lq@yang logout from 19
2.168.31.56
*Dec 6 10:38:24:250 2006 Sysname TAC/7/Event:
TAC_MESSAGE for AAA->TAC:
*Dec 6 10:38:24:250 2006 Sysname TAC/7/Event:
UserID= 2        Flag = 4 TemplateNum = 0
AuthenMethod= 6  AuthenType= 1 AuthenService= 1
UserName=lq  UserPort=vty0
UserAddress=192.168.31.56
AcctService=3  NetworkFlag=0
TaskID= 0      TimeZone= 0
Service=shell  Protocol=
MutilinkMax=0  MutilinkCurrent=0
DisconnectCause=0  DisconnectCauseExtend=0
PacketIn =0  ByteIn=0
PacketOut=0  ByteOut=0
ElapsedTime=9
NasRxSpeed=0   NasTxSpeed=0
TunnelID=0  TunnelConnectionID=0
Command=  Event=  Reason=
Privlevel=1

*Dec 6 10:38:24:304 2006 Sysname TAC/7/Event:
TAC_MESSAGE for AAA->TAC:
*Dec 6 10:38:24:304 2006 Sysname TAC/7/Event:
UserID= 2        Flag = 4 TemplateNum = 0
AuthenMethod= 6  AuthenType= 1 AuthenService= 1
UserName=lq  UserPort=vty0
UserAddress=192.168.31.56
AcctService=3  NetworkFlag=0
TaskID= 0      TimeZone= 0
Service=shell  Protocol=
MutilinkMax=0  MutilinkCurrent=0
DisconnectCause=0  DisconnectCauseExtend=0
PacketIn =0  ByteIn=0
PacketOut=0  ByteOut=0
ElapsedTime=9
NasRxSpeed=0   NasTxSpeed=0
TunnelID=0  TunnelConnectionID=0
Command=  Event=  Reason=
Privlevel=1

// An HWTACACS message was sent from the AAA module to the HWTACACS module.
*Dec 6 10:38:24:522 2006 Sysname TAC/7/Event:
TAC_MESSAGE for TAC->AAA:
*Dec 6 10:38:24:522 2006 Sysname TAC/7/Event:
UserID=2        AcctFlag =3 AcctType =4
Status=1  GroupNum=0
ServerMsg=  DataMsg=

// An HWTACACS message was sent from the HWTACACS module to the AAA module.
# Enable debugging for received HWTACACS packets.

```<Sysname> debugging hwtacacs receive-packet
```

```
*Dec 6 10:43:07:662 2006 Sysname TAC/7/Event:
version:c0  type:AUTHEN_REPLY
seq_no:2  flag:ENCRYPTED_FLAG
session_id:51  length:16
status:AUTHEN_STATUS_GETPASS  flag:REPLY_FLAG_NOECHO
server_msg len:10  data len:0
server_msg:Password:   data:

// HWTACACS received an authentication response message. The server required the password be entered.
```

```
*Dec 6 10:43:07:772 2006 Sysname TAC/7/Event:
version:c0  type:AUTHEN_REPLY
seq_no:4  flag:ENCRYPTED_FLAG
session_id:51  length:6
status:AUTHEN_STATUS_PASS  flag:REPLY_FLAG_ECHO
server_msg len:0  data len:0
server_msg:  data:

// HWTACACS received an authentication success message.
```

```
*Dec 6 10:43:08:98 2006 Sysname TAC/7/Event:
version:c0  type:AUTHOR_REPLY
seq_no:2  flag:ENCRYPTED_FLAG
session_id:54  length:67
status:AUTHOR_STATUS_PASS_ADD
server_msg len:0  data len:0
arg_cnt:3
arg1 len:28  arg2 len:10  arg3 len:20
server_msg:
data:
arg1 :callback-dialstring=82770000  arg2 :priv-lvl=1  arg3 :ftp-directory=flash:

// HWTACACS received an authorization success message.
```

```
*Dec 6 10:43:08:424 2006 Sysname TAC/7/Event:
version:c0  type:ACCOUNTING_REPLY
seq_no:2  flag:ENCRYPTED_FLAG
session_id:57  length:5
server_msg len:0  data len:0
status:ACCT_STATUS_SUCCESS
server_msg:
data:

// HWTACACS received an accounting-start message.
```

```
%Dec 6 10:43:08:446 2006 Sysname SHELL/4/LOGIN: lq@yang login from 192.168.31.56
%Dec 6 10:43:14:146 2006 Sysname SHELL/4/LOGOUT: lq@yang logout from 192.168.31.56

// A user logged in and logged out.
```

```
*Dec 6 10:43:14:446 2006 Sysname TAC/7/Event:
version:c0  type:ACCOUNTING_REPLY
seq_no:2  flag:ENCRYPTED_FLAG
session_id:5a  length:5
```

41
// HWTACACS received an accounting-stop message.

# Enable debugging for sent HWTACACS packets.
<Sysname> debugging hwtacacs send-packet
*Dec  6 10:46:35:206 2006 Sysname TAC/7/Event:
version:c0 type:AUTHEN_REQUEST
seq_no:1 flag:ENCRYPTED_FLAG
session_id:5e length:27
action:AUTHEN_LOGIN priv_lvl:VISIT authen_type:AUTHEN_TYPE.ASCII
service:AUTHEN_SVC_LOGIN
user len:2 port len:4 rem_addr len:13 data len:0
user name:lq port:vty0 rem_addr:192.168.31.56 data:
// HWTACACS sent an authentication request message.

*Dec  6 10:46:35:424 2006 Sysname TAC/7/Event:
version:c0 type:AUTHEN_CONTINUE
seq_no:3 flag:ENCRYPTED_FLAG
session_id:5e length:13
user_msg len:****** data len:0 flag:0
user_msg:******
data:
// HWTACACS sent an authentication-continue message and the password to the HWTACACS server.

*Dec  6 10:46:35:642 2006 Sysname TAC/7/Event:
version:c0 type:AUTHOR_REQUEST
seq_no:1 flag:ENCRYPTED_FLAG
session_id:61 length:46
authen_method:AUTHEN_METH_PLUS priv_lvl:VISIT
authen_type:AUTHEN_TYPE.ASCII authen_service:AUTHEN_SVC_LOGIN
user len:2 port len:4 rem_addr len:13
user :lq port:vty0 rem_addr:192.168.31.56
arg1:service=shell arg2:cmd
// HWTACACS sent an authorization request message.

*Dec  6 10:46:35:978 2006 Sysname TAC/7/Event:
version:c0 type:ACCOUNTING_REQUEST
seq_no:1 flag:ENCRYPTED_FLAG
session_id:64 length:63
flag:ACCT_FLAG_START authen_method:AUTHEN_METH_PLUS
priv_lvl:MONITOR
authen_type:AUTHEN_TYPE.ASCII authen_service:AUTHEN_SVC_LOGIN
user len:2 port len:4 rem_addr len:13
arg_cnt:3
arg1 len:9 arg2 len:10 arg3 len:13
user :lq port:vty0 rem_addr:192.168.31.56
// HWTACACS sent an accounting-start request message.
%Dec 6 10:46:36 2006 Sysname SHELL/4/LOGIN: lq@yang login from 192.168.31.56
%Dec 6 10:46:37 2006 Sysname SHELL/4/LOGOUT: lq@yang logout from 192.168.31.56

// A user logged in and logged out.
*Dec 6 10:46:37 2006 Sysname TAC/7/Event:
version:c0 type:ACCOUNTING_REQUEST
seq_no:1 flag:ENCRYPTED_FLAG
session_id:67 length:182
flag:ACCT_FLAG_STOP authen_method:AUTHEN_METH_PLUS
priv_lvl:MONITOR
authen_type:AUTHEN_TYPE_ASCII authen_service:AUTHEN_SVC_LOGIN
user len:2 port len:4 rem_addr len:13
arg_cnt:12
arg1 len:9 arg2 len:10 arg3 len:13 arg4 len:12
arg5 len:16 arg6 len:10 arg7 len:11 arg8 len:9
arg9 len:10 arg10 len:14 arg11 len:14 arg12 len:14
user:lq port:vty0 rem_addr:192.168.31.56
arg1 :task_id=0 arg2 :timezone=0 arg3 :service=shell arg4 :disc_cause=0
arg5 :disc_cause_ext=0 arg6 :bytes_in=0 arg7 :bytes_out=0 arg8 :paks_in=0
arg9 :paks_out=0 arg10:elapsed_time=1 arg11:nas_rx_speed=0 arg12:nas_tx_speed=0

// HWTACACS sent an accounting-stop request message.
The output description tables in this document only contain fields and messages that require an explanation.

### debugging aaa-ldap

Use `debugging aaa-ldap` to enable LDAP debugging.

Use `undo debugging aaa-ldap` to disable LDAP debugging.

**Syntax**

```
debugging aaa-ldap { all | error | event | fsm }
undo debugging aaa-ldap { all | error | event | fsm }
```

**Default**

LDAP debugging is disabled.

**Views**

User view

**Default command level**

1: Monitor level

**Parameters**

- **all**: Specifies all types of debugging for LDAP.
- **error**: Specifies debugging for errors.
- **event**: Specifies debugging for events.
- **fsm**: Specifies debugging for state machine.

**Usage guidelines**

Table 15 describes output fields and messages for the `debugging aaa-ldap error` command.

### Table 15 Output for the debugging aaa-ldap error command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Failed to connect with server (session id=session-id, scheme id=scheme-id)</td>
<td>The device could not connect to the LDAP server specified in the scheme because of an incorrect IP address or port number of the LDAP server or an LDAP server failure.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| Received invalid message (type=message-type) in state fsm-state-name | LDAP received an invalid message during authentication.  
Message type:  
• 0—AAA request.  
• 1—LDAP response.  
• 2—Timer timeout.  
The `fsm-state-name` argument indicates the current status of the state machine:  
• AUTHEN_INIT.  
• AUTHEN_ADMINBIND.  
• AUTHEN_USRSEARCH.  
• AUTHEN_USRBIND.  
• AUTHOR_INIT.  
• AUTHOR_ADMINBIND1.  
• AUTHOR_USRSEARCH.  
• AUTHOR_USRBIND.  
• AUTHOR_ADMINBIND2.  
• AUTHOR_GRPSEARCH. |
| No response from LDAP server | The LDAP server did not respond after the device sent an LDAP request or established a connection with the LDAP server. The reason might be an incorrect IP address or port number of the LDAP server is configured in the LDAP scheme, or the LDAP server fails. |
| Failed to bind administrator | LDAP failed to bind the administrator due to an incorrect administrator DN in the LDAP scheme. |
| Failed to search user | LDAP failed to find the user because an incorrect username was entered or incorrect user parameters were configured in the LDAP scheme. |
| Failed to bind user | LDAP failed to bind the user because an incorrect user password was entered. |
| Length of user DN exceeded the maximum | The length of user DN exceeded 255 characters. |
| Failed to get user’s group attributes | LDAP failed to get user’s group attributes because the user did not belong to any group. |
| The second binding administrator failed | The second binding of administrator failed because the administrator DN configured in the LDAP scheme is wrong. |
| Failed to send searchUserGroupRequest to server | LDAP failed to send searchUserGroupRequest to the server because of an LDAP server failure or network congestion. |
| Failed to search user’s groups | LDAP failed to search user’s groups because the user did not belong to any group. |
| The message type (type=msg-type) unsupported | LDAP received an unsupported request. The `msg-type` argument can be authentication, authorization or accounting.  
LDAP supports only authentication and authorization in this software version. |
| Failed to connect with LDAP server | LDAP failed to connect to the LDAP server because of an incorrect IP address of the LDAP server configured in the LDAP scheme or insufficient system resources. |
Table 16 describes output fields and messages for the **debugging aaa-ldap event** command.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>user</td>
<td>Authentication/authorization user name.</td>
</tr>
<tr>
<td>user id</td>
<td>User ID assigned by AAA.</td>
</tr>
<tr>
<td>scheme name</td>
<td>LDAP scheme name.</td>
</tr>
<tr>
<td>session id</td>
<td>LDAP session ID.</td>
</tr>
</tbody>
</table>

**Failure reason:**
- 68—Connection with the LDAP server failed.
- 69—No response is from the LDAP server.
- 70—Failed to send request to the LDAP server.
- 71—The number of LDAP sessions reaches the maximum.
- 72—The authentication information is invalid.
- 73—The access method is not supported.
- 74—Failed to bind the administrator DN.
- 75—The user does not belong to any user group.

**Message response timer expired**
When the device established a connection with or sent a request to the LDAP server, the message response timer expired.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>server ip</td>
<td>IP address of the LDAP server.</td>
</tr>
<tr>
<td>server port</td>
<td>TCP port number of the LDAP server.</td>
</tr>
<tr>
<td>login name</td>
<td>Administrator DN configured in the LDAP scheme.</td>
</tr>
</tbody>
</table>

**Search scope:**
- 1—Next lower level of subdirectory of base-dn.
- 2—All subdirectories of base-dn.

**Examples**
The output in the following examples was created when a Telnet user logged into and out of the device under the following conditions:
- The host of the Telnet user is connected to the device through the console port.
- The device uses an LDAP authentication/authorization scheme for Telnet users.

# Enable LDAP event debugging.

```bash
<Sysname> debugging aaa-ldap event
```

*Dec 26 17:30:28:31 2007 Sysname ALDP/7/EVENT: Received authentication request from AAA (user=xyz@aaa, user id=2, scheme name=scheme1).

*Dec 26 17:30:28:47 2007 Sysname ALDP/7/EVENT: Created a session successfully (session id=5).

// LDAP received an authentication request and created a session.


// LDAP was establishing a connection with the LDAP server.
// LDAP received a response and established the connection.
*Dec 26 17:30:28:94 2007 Sysname ALDP/7/EVENT: Sent bindAdminRequest to server successfully (message id=1, session id=5, login name="cn=root").

// LDAP sent a request to bind the administrator DN.
*Dec 26 17:30:28:140 2007 Sysname ALDP/7/EVENT: Received LDAP response message (session id=5).

// LDAP sent a user DN search request.
*Dec 26 17:30:28:203 2007 Sysname ALDP/7/EVENT: Received LDAP response message (session id=5).

// LDAP sent a user DN bind request.
*Dec 26 17:30:28:312 2007 Sysname ALDP/7/EVENT: Received LDAP response message (session id=5).

// The user was bound successfully.
*Dec 26 17:30:28:328 2007 Sysname ALDP/7/EVENT: Sent response message to AAA (user id=2, Authentication success.).

// LDAP sent an authentication success response to the AAA module.
*Dec 26 17:30:28:328 2007 Sysname ALDP/7/EVENT: Unbound the connection from LDAP server (session id=5).
*Dec 26 17:30:28:328 2007 Sysname ALDP/7/EVENT: Deleted session successfully (session id=5).
LDAP was unbound from the LDAP server and tore down the connection.
*Dec 26 17:30:28:375 2007 Sysname ALDP/7/EVENT: Received authorization request from AAA (user=xyz@aaa, user id=2, scheme=scheme1).
*Dec 26 17:30:28:375 2007 Sysname ALDP/7/EVENT: Created a session successfully (session id=6).

LDAP received an authorization request and created a session.

LDAP was establishing a connection with the server.
*Dec 26 17:30:28:390 2007 Sysname ALDP/7/EVENT: Created timer (session id=6, timer id=275).
*Dec 26 17:30:28:422 2007 Sysname ALDP/7/EVENT: Received LDAP response message (session id=6).

LDAP received a reply from the server and created a connection with the server.
*Dec 26 17:30:28:422 2007 Sysname ALDP/7/EVENT: Created timer (session id=6, timer id=276).
*Dec 26 17:30:28:484 2007 Sysname ALDP/7/EVENT: Sent bindAdminRequest to server successfully (message id=1, session id=6, login name="cn=root").

LDAP sent a request to bind the administrator DN.
*Dec 26 17:30:28:500 2007 Sysname ALDP/7/EVENT: LDAP bind result code = 0.
*Dec 26 17:30:28:562 2007 Sysname ALDP/7/EVENT: Unbound the connection to LDAP server (session id=6).

LDAP sent a user DN search request.
*Dec 26 17:30:28:562 2007 Sysname ALDP/7/EVENT: Created timer (session id=6, timer id=277).
*Dec 26 17:30:28:609 2007 Sysname ALDP/7/EVENT: Sent bindUserRequest to server successfully (message id=1, session id=6, user dn=cn=xyz,ou=Austin,o=scheme1,c=us).

LDAP sent a user DN bind request.
*Dec 26 17:30:28:609 2007 Sysname ALDP/7/EVENT: Created timer (session id=6, timer id=277).
*Dec 26 17:30:28:672 2007 Sysname ALDP/7/EVENT: LDAP bind result code = 0.
* Dec 26 17:30:28:672 2007 Sysname ALDP/7/EVENT: Unbound the connection from LDAP server (session id=6).
* Dec 26 17:30:28:687 2007 Sysname ALDP/7/EVENT: Created timer (session id=6, timer id=278).
* Dec 26 17:30:28:734 2007 Sysname ALDP/7/EVENT: Received LDAP response message (session id=6).
* Dec 26 17:30:28:734 2007 Sysname ALDP/7/EVENT: Sent bindAdminRequest to server successfully (message id=1, session id=6, login name="cn=root").

// LDAP sent a secondary administrator DN bind request.

* Dec 26 17:30:28:781 2007 Sysname ALDP/7/EVENT: Received LDAP response message (session id=6).
* Dec 26 17:30:28:781 2007 Sysname ALDP/7/EVENT: Received LDAP response to the secondbindAdminRequest successfully (session id=6).
* Dec 26 17:30:28:781 2007 Sysname ALDP/7/EVENT: LDAP bind result code = 0.
* Dec 26 17:30:28:781 2007 Sysname ALDP/7/EVENT: Sent searchUserGroupRequest to server successfully (message id=2, session id=6, search base dn="o=scheme1,c=us", search scope=2, filter="(&(objectClass=groupOfNames)(member=cn=xyz,ou=Austin,o=scheme1,c=us))").

// LDAP sent a user group search request.

* Dec 26 17:30:28:797 2007 Sysname ALDP/7/EVENT: Created timer (session id=6, timer id=279).
* Dec 26 17:30:28:844 2007 Sysname ALDP/7/EVENT: Received LDAP response to searchUserGroupRequest (session id=6).
* Dec 26 17:30:28:844 2007 Sysname ALDP/7/EVENT: LDAP search result code = 0.
* Dec 26 17:30:28:844 2007 Sysname ALDP/7/EVENT: Sent response message to AAA (user id=2, Authorization success.).

// LDAP sent an authorization success response to the AAA module.

* Dec 26 17:30:28:859 2007 Sysname ALDP/7/EVENT: Unbound the connection to LDAP server (session id=6).
* Dec 26 17:30:28:859 2007 Sysname ALDP/7/EVENT: Deleted session successfully (session id=6).

// LDAP was unbound from the LDAP server and tore down the connection.

# Enable debugging for the LDAP state machine.
<Sysname> debugging aaa-ldap fsm
* Dec 10 11:25:30:984 2007 Sysname ALDP/7/FSM: Session (session id=1) state changed from AUTHEN_INIT to AUTHEN_ADMINBIND.
* Dec 10 11:25:32:476 2007 Sysname ALDP/7/FSM: Session (session id=1) state changed from AUTHEN_ADMINBIND to AUTHEN_USRSEARCH.
* Dec 10 11:25:33:457 2007 Sysname ALDP/7/FSM: Session (session id=1) state changed from AUTHEN_USRSEARCH to AUTHEN_USRBIND.

// Authentication state machine's state was changed.
* Dec 10 11:25:33:888 2007 Sysname ALDP/7/FSM: Session (session id=2) state changed from AUTHOR_INIT to AUTHOR_ADMINBIND1.
* Dec 10 11:25:33:808 2007 Sysname ALDP/7/FSM: Session (session id=2) state changed from AUTHOR_ADMINBIND1 to AUTHOR_USRSEARCH.
*Dec 10 11:25:33:858 2007 Sysname ALDP/7/FSM: Session (session id=2) state changed from AUTHOR_USRSEARCH to AUTHOR_USRBIND.
*Dec 10 11:25:34:58 2007 Sysname ALDP/7/FSM: Session(session id=2) state changed from AUTHOR_USRBIND to AUTHOR_ADMINBIND2.
*Dec 10 11:25:34:158 2007 Sysname ALDP/7/FSM: Session(session id=2) state changed from AUTHOR_ADMINBIND2 to AUTHOR_GRPSEARCH.

// Authorization state machine's state was changed.

# Enable LDAP error debugging. When an incorrect administrator DN is configured on the device, output similar to the following example is generated:
<Sysname> debugging aaa-ldap error
*Jan 5 08:48:57:344 2008 Sysname ALDP/7/ERROR: Failed to bind administrator (session id=1).

// LDAP failed to bind the administrator DN.
The output description tables in this document only contain fields and messages that require an explanation.

**debugging sc**

Use **debugging sc** to enable debugging for the SC module.

Use **undo debugging sc** to disable debugging for the SC module.

**Syntax**

```
debugging sc { all | error | event }
undo debugging sc { all | error | event }
```

**Default**

Debugging for the SC module is disabled.

**Views**

User view

**Default command level**

1: Monitor level

**Parameters**

- **all**: Specifies all types of debugging for the SC module.
- **error**: Specifies debugging for SC errors.
- **event**: Specifies debugging for SC events.

**Usage guidelines**

Table 17 describes output fields and messages for the **debugging sc event** command.

**Table 17 Output from the debugging sc event command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ulRequestType</td>
<td>AAA request type</td>
</tr>
<tr>
<td>ulLoginUserID</td>
<td>Login user ID</td>
</tr>
<tr>
<td>ulUserID</td>
<td>AAA connection index</td>
</tr>
<tr>
<td>ulPhyType</td>
<td>Physical port type</td>
</tr>
<tr>
<td>ulIpAddr</td>
<td>IP address</td>
</tr>
<tr>
<td>szCalledNumber</td>
<td>Called number</td>
</tr>
<tr>
<td>szCallingNumber</td>
<td>Calling number</td>
</tr>
<tr>
<td>szUserName</td>
<td>Username</td>
</tr>
<tr>
<td>szDomainName</td>
<td>Domain name</td>
</tr>
<tr>
<td>ulIfNetIndex</td>
<td>Interface index</td>
</tr>
</tbody>
</table>
### Field Description

- **szPassword**: Password.
- **ulIsStopOnlyAcct**: Whether stop-only accounting is used.
- **ulVOIPAcctPacketType**: Accounting packet type.
- **ulVOIPServiceType**: Service type: card user or calling user.
- **ulVOIPRemainMoney**: Remaining money.
- **ulVOIPRemainTime**: Remaining time.
- **ulVOIPAcctConnectionTime**: Accounting connection time.
- **ulVOIPErrorReason**: AAA service error reason.
- **ulVOIPCodecType**: Code of a call, reserved.
- **ulVOIPCallReference**: Call reference that identifies a call.
- **ulVOIPOrg_Gk_Addr**: IP address of source gatekeeper.
- **ulVOIPOrg_Gw_Addr**: IP address of source gateway.
- **ulVOIPDst_Gk_Addr**: IP address of the destination gatekeeper.
- **ulVOIPDst_Gw_Addr**: IP address of the destination gateway.
- **ulVOIPAcctSessionTime**: Accounting session time.
- **ucVOIPAccessNum**: Access number.
- **ucVOIPTransferNum**: Call transfer number, reserved.
- **Use new AAA connection(Log-index=mm)**: Call mm requests to use a new AAA connection.

#### Examples

# Enable SC error debugging. Output similar to the following example is generated when a user attempts to use an expired local guest account for login:

```
<Sysname> debugging sc error
// Local user named test failed the authentication because the account had expired.
```
Local server debugging commands

debugging local-server

Use `debugging local-server` to enable debugging for the local server function.

Use `undo debugging local-server` to disable debugging for the local server function.

Syntax

```
debugging local-server { all | error | event | packet }
undo debugging local-server { all | error | event | packet }
```

Default

Debugging for the local server function is disabled.

Views

User view

Default command level

```
1: Monitor level
```

Parameters

- **all**: Specifies all types of debugging for the local server function.
- **error**: Specifies error debugging.
- **event**: Specifies event debugging.
- **packet**: Specifies packet debugging.

Examples

# Enable local server error debugging. The output in this example was created when the following
# conditions exist:
#  - The RADIUS offload function is enabled on the device.
#  - PEAP-MSCHAPv2 is not specified as the EAP authentication method on the local server.
```
<Sysname> debugging local-server error
*Mar 28 09:46:57:563 2008 Sysname LS/7/ERROR:
EAP type is invalid or not supported.
*Mar 28 09:46:57:563 2008 Sysname LS/7/ERROR:
Failed to initiate user's EAP type handler.
*Mar 28 09:46:57:563 2008 Sysname LS/7/ERROR:
Failed to select and call EAP type interface according to the EAP packet.
```

// Authentication failed because the EAP module could not find the interface of the PEAP module.

# Enable local server event debugging. Output similar to the following example is generated when an
# 802.1X client initiates an authentication process under the following conditions:
#  - Local EAP authentication is enabled on the device.
#  - EAP-MD5 is specified as the EAP authentication method on the local server.
```
<Sysname> debugging local-server event
```

53
*Mar 28 09:52:58:860 2008 Sysname LS/7/EVENT:
  EAP_Type: 1
  // The EAP server received a response of the ID type.
*Mar 28 09:52:59:313 2008 Sysname LS/7/EVENT:
  EAP_Type: 4
  // The EAP server received a response of the type MD5 challenge.
*Mar 28 09:52:59:313 2008 Sysname LS/7/EVENT:
  EAP_MD5 passwords matched
  // The password for EAP_MD5 authentication was matched.

# Enable local server packet debugging. Output similar to the following example is generated when an 802.1X client initiates an authentication process under the following conditions:
- Local EAP authentication is enabled on the device.
- EAP-MD5 is specified as the EAP authentication method on the local server.

< Sysname > debugging local-server packet
*Mar 28 08:45:15:253 2008 Sysname LS/7/PACKET:
  EAP Server received an eap packet:
*Mar 28 08:45:15:260 2008 Sysname LS/7/PACKET:
  02 01 00 0d 01 63 61 69 7a 69 62 69 6e
  // The EAP server received a response of the ID type and including the username.
*Mar 28 08:45:15:275 2008 Sysname LS/7/PACKET:
  MD5 challenge:
*Mar 28 08:45:15:280 2008 Sysname LS/7/PACKET:
  c2 a5 3a eb 40 bb 46 a6 d9 21 7d 61 b8 65 57 a8
  // The MD5 module generated an MD5 challenge.
*Mar 28 08:45:15:290 2008 Sysname LS/7/PACKET:
  EAP Server replied an eap packet:
*Mar 28 08:45:15:300 2008 Sysname LS/7/PACKET:
  01 02 00 16 04 10 c2 a5 3a eb 40 bb 46 a6 d9 21 7d 61 b8 65 57 a8
  // The EAP server sent an MD5 challenge request.
*Mar 28 08:45:15:310 2008 Sysname LS/7/PACKET:
  EAP Server received an eap packet:
*Mar 28 08:45:15:320 2008 Sysname LS/7/PACKET:
  02 02 00 04 10 3d 1e 81 b5 56 3d 18 ee 4d a9 4c ec 0b 24 85 b2 63 61 69 7a 69 62 69 6e
  // The EAP server received an MD5 challenge response.
*Mar 28 08:45:15:330 2008 Sysname LS/7/PACKET:
  EAP Server replied an eap packet:
*Mar 28 08:45:15:340 2008 Sysname LS/7/PACKET:
  03 02 00 04
  // The EAP server sent an authentication acknowledgement packet.
RADIUS server debugging commands

debugging radius-server

Use debugging radius-server to enable debugging for the RADIUS server function.
Use undo debugging radius-server to disable debugging for the RADIUS server function.

Syntax

debugging radius-server { all | error | event | packet }
undo debugging radius-server { all | error | event | packet }

Default

Debugging for the RADIUS server function is disabled.

Views

User view

Default command level

1: Monitor level

Parameters

all: Specifies all types of debugging.
error: Specifies error debugging.
event: Specifies event debugging.
packet: Specifies packet debugging.

Examples

The output in the following examples was created when the RADIUS server function is enabled on the device.

# Enable RADIUS server event debugging. Output similar to the following example is generated when a RADIUS client is specified on the device:
<Sysname> debugging radius-server event
<Sysname> system-view
[Sysname] radius-server client-ip 8.8.8.2
[Sysname]
*Jan 25 18:57:10:583 2010 Sysname RDSERVER/7/DEBUG:
  Open socket.
  // The RADIUS server's socket was open.

# Enable RADIUS server packet debugging. Output similar to the following example is generated when a user named aaa initiates an authentication process to the device:
<Sysname> debugging radius-server packet
*Apr 30 16:23:05:906 2010 Sysname RDSERVER/7/DEBUG
  Received Packet: Authentication request
  // The device received an authentication request from the RADIUS client.
The device displayed packet data in hexadecimal format.

*Apr 30 16:23:06:487 2010 Sysname RDSERVER/7/DEBUG:
[1 User-Name ] [3 ] [aaa]
[2 User-Password ] [18 ] [c13a0f403cbf3a24d1fc8e0fae32fb86]
[4 NAS-IP-Address ] [6 ] [192.168.1.12]
[32 NAS-Identifier ] [4 ] [Sysname]

The device sent an authentication success response to the RADIUS client.
*Apr 30 16:23:06:838 2010 Sysname RDSERVER/7/DEBUG:
Send Packet : Authentication success

The device displayed packet data in hexadecimal format.
ACFP debugging commands

The output description tables in this document only contain fields and messages that require an explanation.

debugging acfp client

Use `debugging acfp client` to enable ACFP client debugging.

Use `undo debugging acfp client` to disable ACFP client debugging.

Syntax

```
depugging acfp client error [ verbose ]
do debugging acfp client error
```

Default

ACFP client debugging is disabled.

Views

User view

Default command level

1: Monitor level

Parameters

- **error**: Specifies debugging for ACFP client errors.
- **verbose**: Specifies detailed debugging for ACFP client errors.

Usage guidelines

Table 18 describes output fields and messages for the `debugging acfp client error` command.

**Table 18 Output from the debugging acfp client error command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Failed to get OID by CMO</td>
<td>The ACFP client failed to obtain the OID through configuration management object.</td>
</tr>
</tbody>
</table>

Table 19 describes output fields and messages for the `debugging acfp client error verbose` command.

**Table 19 Output from the debugging acfp client verbose command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>creating/deleting/modifying the client information</td>
<td>The ACFP client was creating, deleting, or modifying the client information on the server.</td>
</tr>
<tr>
<td>getting/addiing/deleting/modifying the policy</td>
<td>The ACFP client was getting, adding, deleting, or modifying the policy from the server.</td>
</tr>
<tr>
<td>deleting the policy by interface</td>
<td>The ACFP client was deleting all policy information related to an interface.</td>
</tr>
<tr>
<td>Filed</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>getting the context ID</td>
<td>The ACFP client was getting the context ID of a policy.</td>
</tr>
<tr>
<td>getting/adding/deleting the rule</td>
<td>The ACFP client was getting, adding, or deleting a rule from the server.</td>
</tr>
<tr>
<td>getting operation status of a rule</td>
<td>The ACFP client was getting the operation status of a rule from the server.</td>
</tr>
</tbody>
</table>

**Examples**

# Enable ACFP client error debugging. When an interface without an IP address is selected on the ACFP client configuration page, output similar to the following example is generated:

```
<Sysname> debugging acfp client error
```

# Enable ACFP client detailed debugging. The output in this example was created when the following conditions exist:

- ACSEI client, ACFP client, SNMP client, and remote interface management are enabled on the device.
- ACSEI server, ACFP server, and SNMP agent features are disabled on the device specified as the ACFP server.

```
<DeviceA> debugging acfp client error verbose
```

debugging acfp server

Use **debugging acfp server** to enable ACFP server debugging.

Use **undo debugging acfp server** to disable ACFP server debugging.

**Syntax**

```
debugging acfp server { all | error | event | packet }
undo debugging acfp server { all | error | event | packet }
```

**Default**

ACFP server debugging is disabled.

**Views**

User view

**Default command level**

2: System level

**Parameters**

- **all**: Specifies all types of debugging for ACFP server.
- **error**: Specifies ACFP server error debugging.
- **event**: Specifies ACFP server event debugging.
- **packet**: Specifies ACFP server packet debugging.
Usage guidelines

Table 20 describes output fields and messages for the `debugging acfp client error` command.

**Table 20 Output from the debugging acfp server error command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Must bind EndTime with StartTime</td>
<td>The policy end time must be bound with the start time. This information is output when the end time is not configured.</td>
</tr>
<tr>
<td>Must bind StartTime with EndTime</td>
<td>The policy start time must be bound with the end time. This information is output when the end time is not configured.</td>
</tr>
<tr>
<td>Cannot specify both the inbound interface and the outbound interface</td>
<td>The inbound interface and the outbound interface cannot be specified at the same time.</td>
</tr>
</tbody>
</table>

Table 21 describes output fields and messages for the `debugging acfp client error verbose` command.

**Table 21 Output from the debugging acfp server packet command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP mirrored packet was forwarded</td>
<td>IP packet was mirrored to the destination port.</td>
</tr>
<tr>
<td>IP mirrored packet was fast forwarded</td>
<td>Mirrored IP packet was fast forwarded to the destination port.</td>
</tr>
<tr>
<td>IP redirected packet was forwarded</td>
<td>IP packet was redirected to the destination port.</td>
</tr>
<tr>
<td>IP redirected packet was fast forwarded</td>
<td>Redirected IP packet was fast forwarded to the destination port.</td>
</tr>
<tr>
<td>IP mirrored or redirected packet was forwarded</td>
<td>Mirrored or redirected IP packet was forwarded to the destination port.</td>
</tr>
<tr>
<td>Mirrored Packet was sent to slot slot-number</td>
<td>Mirrored packet was transparently transmitted to the board in slot slot-number.</td>
</tr>
<tr>
<td>Redirected packet was sent to slot slot-number</td>
<td>Redirected packet was transparently transmitted to the board in slot slot-number.</td>
</tr>
</tbody>
</table>

Examples

# Enable ACFP server error debugging. If you configure an ACFP client by using the MIB browser when ACFP is disabled, output similar to the following example is generated:

```
<Sysname> debugging acfp server error
*Feb 24 10:33:58:500 2006 Sysname ACFP/7/debug:
ACFP_Error:   ACFP is disabled.
// Configuration failed because ACFP was disabled.
```

# Enable ACFP server event debugging. If you configure an ACFP client by using the MIB browser when ACFP is enabled, output similar to the following example is generated:

```
<Sysname> debugging acfp server event
*Feb 24 10:36:29:00 2006 Sysname ACFP/7/debug:
ACFP_Event:   A client was added successfully.
// The ACFP client was created successfully.
```
# Enable ACFP server packet debugging. If you configure the ACFP rule as dropping all packets received on Ethernet 1/1, output similar to the following example is generated:

<Sysname> debugging acfp server packet
*Feb 24 10:56:16:256 2006 Sysname ACFP/7/debug:
ACFP_Packet: IP packet was dropped.

// IP packets received on Ethernet 1/1 were dropped.
ACSEI debugging commands

The output description tables in this document only contain fields and messages that require an explanation.

ACSEI server debugging commands

debugging acsei server

Use `debugging acsei server` to enable ACSEI debugging.

Use `undo debugging acsei server` to disable ACSEI debugging.

Syntax

```
debugging acsei server { all | error | event | packet }
undo debugging acsei server { all | error | event | packet }
```

Default

ACSEI debugging is disabled.

Views

User view

Default command level

2: System level

Parameters

- `all`: Specifies all types of ACSEI debugging.
- `error`: Specifies ACSEI error debugging.
- `event`: Specifies ACSEI event debugging.
- `packet`: Specifies ACSEI packet debugging.

Usage guidelines

Table 22 describes output fields and messages for the `debugging acsei server packet` command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interface: <code>Interface</code></td>
<td>Interface enabled with ACSEI client.</td>
</tr>
<tr>
<td>DstMac: <code>DstMac</code></td>
<td>Destination MAC address of the packet.</td>
</tr>
<tr>
<td>SrcMac: <code>SrcMac</code></td>
<td>Source MAC address of the packet.</td>
</tr>
<tr>
<td>FrameType: <code>FrameType</code></td>
<td>Frame type of the packet.</td>
</tr>
<tr>
<td>ProtocolType: <code>ProtocolType</code></td>
<td>Protocol type of the packet.</td>
</tr>
<tr>
<td>Reserved: <code>Reserved</code></td>
<td>Reserved bits of the packet protocol.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Ver: Version</td>
<td>ACSEI version.</td>
</tr>
<tr>
<td>clientOrserver: 0 or 1</td>
<td>Packet sender:</td>
</tr>
<tr>
<td></td>
<td>• 0–ACSEI client.</td>
</tr>
<tr>
<td></td>
<td>• 1–ACSEI server.</td>
</tr>
<tr>
<td>Type: Message-Type</td>
<td>Packet type:</td>
</tr>
<tr>
<td></td>
<td>• 0x01–Registration packets.</td>
</tr>
<tr>
<td></td>
<td>• 0x02–Information notifications.</td>
</tr>
<tr>
<td></td>
<td>• 0x03–Operation notifications.</td>
</tr>
<tr>
<td></td>
<td>• 0x04–Monitoring packets.</td>
</tr>
<tr>
<td></td>
<td>• 0x05–Deregistration packets.</td>
</tr>
<tr>
<td>clientID: clientID</td>
<td>Client with which the server is communicating. A client ID uniquely identifies a client.</td>
</tr>
<tr>
<td>Length: Length</td>
<td>Length of the ACSEI packet excluding the Ethernet header.</td>
</tr>
<tr>
<td>Inner Code: InnerCode</td>
<td>Inner code of the ACSEI packet:</td>
</tr>
<tr>
<td></td>
<td>• Request, acknowledgement, or denial for a registration packet.</td>
</tr>
<tr>
<td></td>
<td>• Monitoring request or acknowledgement for a monitoring packet.</td>
</tr>
<tr>
<td>Identifier: Identifier</td>
<td>Sequence number of the ACSEI packet.</td>
</tr>
<tr>
<td>Length: Length</td>
<td>Length of the ACSEI packet excluding the Ethernet header and ACSEI header.</td>
</tr>
</tbody>
</table>

**Examples**

```
# Enable ACSEI server, terminal monitoring, and ACSEI server debugging. When an ACSEI client registers with the ACSEI server, output similar to the following example is generated:
<Sysname> system-view
<Sysname> acsei server enable
<Sysname> quit
<Sysname> terminal monitor
<Sysname> terminal debugging
<Sysname> debugging acsei server all
# Jul 17 07:54:21:921 2007 Sysname ACSEI/8/Debug_ACSEI_Recv_Pkt:
Receiving ACSEI packet:

Interface: Ethernet1/0
DstMac   : 010f-e200-0021 SrcMac   : 000d-88f7-6701
FrameType: 88a7 ProtocolType: 0007 Reserved: 0000
Ver: 1 clientOrserver: 0 Type: 1
clientID: 0 Length: 10
Inner Code: 1 Identifier: 72 Length: 4

// Packet debugging information: The ACSEI server received a registration request on Ethernet 1/0 from the ACSEI client. The MAC address of the ACSEI client is 000d-88f7-6701.
# Jul 17 07:54:21:921 2007 Sysname ACSEI/8/Debug_ACSEI_Send_Pkt:
Sending ACSEI packet:

Interface : Ethernet1/0
```

62
// Packet debugging information: The ACSEI server sent a registration acknowledgement to the ACSEI client and assigned the client the ID 1.

// Event debugging information: Client 1 registered successfully and its status changed to enabled.

// Packet debugging information: The ACSEI server received a message notification from the ACSEI client.

// Event debugging information: The ACSEI server notified other applications on the device that they could obtain client information.

// Packet debugging information: The ACSEI server sent a monitoring request to the ACSEI client.

// Packet debugging information: The ACSEI server received a monitoring request from the ACSEI client.
Sending ACSEI packet:

Interface : Ethernet1/0
DstMac   :000d-88f7-6701  SrcMac   :00e0-fc34-ec14
FrameType : 88a7  ProtocolType : 0007 Reserved : 0000
Ver : 1 clientOrserver : 1 Type: 4
clientID : 1  Length : 16
Inner Code : 2  Identifier : 1 Length : 10

// Packet debugging information: The ACSEI server sent a monitoring acknowledgement to the ACSEI client.

Receiving ACSEI packet:

Interface : Ethernet1/0
DstMac   :00e0-fc34-ec14  SrcMac   :000d-88f7-6701
FrameType : 88a7  ProtocolType : 0007 Reserved : 0000
Ver : 1 clientOrserver : 0 Type: 4
clientID : 1  Length : 16
Inner Code : 2  Identifier : 1 Length : 10

// Packet debugging information: The ACSEI server received a monitoring acknowledgement from the ACSEI client.

ACSEI client debugging commands

debugging acsei-client

Use **debugging acsei-client** to enable ACSEI client debugging.
Use **undo debugging acsei-client** to disable ACSEI client debugging.

**Syntax**

debugging acsei-client { all | error | event | packet }
undo debugging acsei-client { all | error | event | packet }

**Default**

ACSEI client debugging is disabled.

**Views**

User view

**Default command level**

2: System level

**Parameters**

- **all**: Specifies all types of ACSEI client debugging.
- **error**: Specifies ACSEI client error debugging.
- **event**: Specifies ACSEI client event debugging.
packet: Specifies ACSEI client packet debugging.

Usage guidelines

Table 23 describes output fields and messages for the debugging acsei-client packet command.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interface: interface</td>
<td>Interface enabled with ACSEI client.</td>
</tr>
<tr>
<td>DstMac: dst-mac</td>
<td>Destination MAC address of the packet.</td>
</tr>
<tr>
<td>SrcMac: src-mac</td>
<td>Source MAC address of the packet.</td>
</tr>
<tr>
<td>FrameType: frame-type</td>
<td>Frame type of the packet.</td>
</tr>
<tr>
<td>ProtocolType: protocol-type</td>
<td>Protocol type of the packet.</td>
</tr>
<tr>
<td>Reserved: reserved-value</td>
<td>Reserved bits of the packet protocol.</td>
</tr>
<tr>
<td>Ver: acsei-version</td>
<td>ACSEI version.</td>
</tr>
<tr>
<td>clientOrserver: value</td>
<td>Packet sender:</td>
</tr>
<tr>
<td></td>
<td>• 0–ACSEI client.</td>
</tr>
<tr>
<td></td>
<td>• 1–ACSEI server.</td>
</tr>
<tr>
<td>Type: message-type</td>
<td>Packet type:</td>
</tr>
<tr>
<td></td>
<td>• 0X01–Registration packet.</td>
</tr>
<tr>
<td></td>
<td>• 0X02–Information notification.</td>
</tr>
<tr>
<td></td>
<td>• 0X03–Operation notification.</td>
</tr>
<tr>
<td></td>
<td>• 0X04–Monitoring packet.</td>
</tr>
<tr>
<td></td>
<td>• 0X05–Deregistration packet.</td>
</tr>
<tr>
<td>ClientID: client-id</td>
<td>ID of the client. A client ID uniquely identifies a client. A client ID must</td>
</tr>
<tr>
<td></td>
<td>be greater than 0 unless it is in a registration request, registration</td>
</tr>
<tr>
<td></td>
<td>denial, or deregistration packet.</td>
</tr>
<tr>
<td>Length: length</td>
<td>Length of the ACSEI packet excluding the Ethernet frame header.</td>
</tr>
<tr>
<td>Inner Code: inner-code</td>
<td>Inner code of the ACSEI packet:</td>
</tr>
<tr>
<td></td>
<td>• Request, acknowledgement, or denial for a registration packet.</td>
</tr>
<tr>
<td></td>
<td>• Monitoring request or acknowledgement for a monitoring packet.</td>
</tr>
<tr>
<td>Identifier: identifier</td>
<td>Sequence number of the ACSEI packet.</td>
</tr>
<tr>
<td>Length: length</td>
<td>Length of the ACSEI packet excluding the Ethernet frame header and ACSEI</td>
</tr>
<tr>
<td></td>
<td>header.</td>
</tr>
</tbody>
</table>

Examples

# Enable the ACSEI client function on interface Ethernet 1/0, and enable terminal monitoring and ACSEI client debugging. When the ACSEI client registers with an ACSEI server, output similar to the following example is generated:

<Sysname> system-view
[Sysname] interface ethernet 1/0
[Sysname-Ethernet1/0] acsei-client enable
[Sysname-Ethernet1/0] quit
<Sysname> terminal monitor
<Sysname> terminal debugging
Sending ACSEI packet:

Interface: Ethernet 1/0
DstMac : 010f-e200-0021 SrcMac : 000d-88f7-6701
FrameType: 88a7 ProtocolType: 0007 Reserved: 0000
Ver: 1 clientOrserver : 0 Type: 1
clientID : 0 Length: 10
Inner Code : 1 Identifier : 72 Length : 4

Packet debugging information: The ACSEI client sent a registration request to an ACSEI server. The MAC address of the ACSEI server is 010f-e200-0021.

Receiving ACSEI packet:

Interface : Ethernet 1/0
DstMac : 000d-88f7-6701 SrcMac : 010f-e200-0021
FrameType : 88a7 ProtocolType : 0007 Reserved : 0000
Ver : 1 clientOrserver : 1 Type: 1
clientID : 1 Length : 10
Inner Code : 2 Identifier : 72 Length : 4

Packet debugging information: The ACSEI client received a registration acknowledgement from the ACSEI server and was assigned the client ID 1.

Sending ACSEI packet:

Interface: Ethernet 1/0
DstMac : 010f-e200-0021 SrcMac : 000d-88f7-6701
FrameType: 88a7 ProtocolType: 0007 Reserved: 0000
Ver: 1 clientOrserver : 0 Type: 2
clientID : 1 Length: 10

Packet debugging information: The ACSEI client sent an information notification to the ACSEI server.

Sending ACSEI packet:

Interface: Ethernet 1/0
DstMac : 010f-e200-0021 SrcMac : 000d-88f7-6701
FrameType: 88a7 ProtocolType: 0007 Reserved: 0000
Ver: 1 clientOrserver : 0 Type: 4
clientID : 1 Length: 10
Inner Code : 1 Identifier : 5 Length : 10

Packet debugging information: The ACSEI client sent a monitoring request to the ACSEI server.
Receiving ACSEI packet:

Interface : Ethernet 1/0  
DstMac   :000d-88f7-6701  SrcMac   :010f-e200-0021  
FrameType : 88a7  ProtocolType : 0007  Reserved : 0000  
Ver : 1  clientOrserver : 1  Type: 4  
clientID : 1  Length : 10  
Inner Code : 2  Identifier : 5  Length : 10  

// Packet debugging information: The ACSEI client received a monitoring acknowledgement from the ACSEI server.

# Jul 17 08:01:12:00 2007 Sysname ACSEI/8/ Debug_ACSEI_Recv_Pkt:
Receiving ACSEI packet:

Interface : Ethernet 1/0  
DstMac   :000d-88f7-6701  SrcMac   :010f-e200-0021  
FrameType : 88a7  ProtocolType : 0007  Reserved : 0000  
Ver : 1  clientOrserver : 1  Type: 4  
clientID : 1  Length : 10  
Inner Code : 1  Identifier : 1  Length : 4  

// Packet debugging information: The ACSEI client received a monitoring request from the ACSEI server.

# Jul 17 08:01:12:00 2007 Sysname ACSEI/8/ Debug_ACSEI_Send_Pkt:
Sending ACSEI packet:

Interface: Ethernet 1/0  
DstMac   : 010f-e200-0021  SrcMac   : 000d-88f7-6701  
FrameType: 88a7  Protocol Type: 0007  Reserved: 0000  
Ver: 1  clientOrserver : 0  Type: 4  
clientID : 1  Length: 10  
Inner Code : 2  Identifier : 1  Length : 4  

// Packet debugging information: The ACSEI client sent monitoring acknowledgements to ACSEI server.

# Jul 17 08:02:23:386 2007 Sysname ACSEI/8/ Debug_ACSEI_Send_Pkt:
Sending ACSEI packet:

Interface: Ethernet 1/0  
DstMac   : 010f-e200-0021  SrcMac   : 000d-88f7-6701  
FrameType: 88a7  Protocol Type: 0007  Reserved: 0000  
Ver: 1  clientOrserver : 0  Type: 1
The ACSEI client sent a registration request to an ACSEI server. The MAC address of the ACSEI server is 010f-e200-0021.

Receiving ACSEI packet:

Interface: Ethernet 1/0
DstMac: 000d-88f7-6701  SrcMac: 010f-e200-0021
FrameType: 88a7  ProtocolType: 0007  Reserved: 0000
Ver: 1  clientOrserver: 1  Type: 1
clientID: 0  Length: 10
Inner Code: 3  Identifier: 72  Length: 4

The ACSEI client received a registration-denied packet from the ACSEI server.
Active and standby switchover debugging commands

The output description tables in this document only contain fields and messages that require an explanation.

The active and standby switchover module name is identified as "HA" in debugging messages.

deeding ha

Use debugging ha to enable active and standby switchover debugging.
Use undo debugging ha to disable active and standby switchover debugging.

Syntax

debugging ha { all | error | event | message | state }
undo debugging ha { all | error | event | message | state }

Default

Active and standby switchover debugging is disabled.

Views

User view

Default command level

1: Monitor level

Parameters

all: Specifies all types of debugging for active and standby switchover.
error: Specifies debugging for active and standby switchover errors.
event: Specifies debugging for active and standby switchover events.
message: Specifies debugging for active and standby switchover messages.
state: Specifies debugging for active and standby switchover states.

Usage guidelines

Table 24 describes output fields and messages for the debugging ha error command.
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>HA IPC Information:</strong></td>
<td>Output for Inter-Process Communication (IPC) failure:</td>
</tr>
<tr>
<td>NextSend</td>
<td>Next sending position.</td>
</tr>
<tr>
<td>NextFreeSlot</td>
<td>Next free position.</td>
</tr>
<tr>
<td>Acked</td>
<td>Last acknowledge position.</td>
</tr>
<tr>
<td>NumOfMsg</td>
<td>Number of messages.</td>
</tr>
<tr>
<td>WindowAllowed</td>
<td>Free space of the window.</td>
</tr>
<tr>
<td><strong>SMB is down, Can’t send the command to SMB</strong></td>
<td>The standby MPU was not in position when information was synchronized to the standby MPU.</td>
</tr>
<tr>
<td>Len</td>
<td>The following information was printed when the packet exceeded the maximum length:</td>
</tr>
<tr>
<td>length, Msg Type = msg-type, SrcModID = src-mid, SrcSID = src-sid, DstModID = dst-mid, DstSID = dst-sid</td>
<td>• length—Message length.</td>
</tr>
<tr>
<td></td>
<td>• msg-type—Message type.</td>
</tr>
<tr>
<td></td>
<td>• src-mid—Source module ID.</td>
</tr>
<tr>
<td></td>
<td>• src-sid—Source module sub ID.</td>
</tr>
<tr>
<td></td>
<td>• dst-mid—Destination module ID.</td>
</tr>
<tr>
<td></td>
<td>• dst-sid—Destination module sub ID.</td>
</tr>
<tr>
<td>ulSeqNum</td>
<td>IPC output the following message information when it detected that the sequence number of the message exceeded the maximum number:</td>
</tr>
<tr>
<td>ul-seqnumber, Msg Type = msg-type, SrcModID = src-mid, SrcSID = src-sid, DstModID = dst-mid, DstSID = dst-sid</td>
<td>• ul-seqnumber—Message sequence number.</td>
</tr>
<tr>
<td></td>
<td>• msg-type—Message type.</td>
</tr>
<tr>
<td></td>
<td>• src-mid—Source module ID.</td>
</tr>
<tr>
<td></td>
<td>• src-sid—Source module sub ID.</td>
</tr>
<tr>
<td></td>
<td>• dst-mid—Destination module ID.</td>
</tr>
<tr>
<td></td>
<td>• dst-sid—Destination module sub ID.</td>
</tr>
<tr>
<td>Ack Received %d is not matching the expected ack %d, pHaRecvMsg = %X</td>
<td>IPC detected that the message sequence number was incorrect, so it output message information. pHaRecvMsg in the message indicates the memory address of the received message.</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>SrcMID = src-mid, SrcSID = src-sid, DstMID = dst-mid, DstSid = dst-sid, Msg Type = msg-type, Msg Len = length, Msg Seq = msg-seq</td>
<td>IPC failed to send the message, and sent the message again.</td>
</tr>
<tr>
<td></td>
<td>• src-mid—Source module ID.</td>
</tr>
<tr>
<td></td>
<td>• src-sid—Source module sub ID.</td>
</tr>
<tr>
<td></td>
<td>• dst-mid—Destination module ID.</td>
</tr>
<tr>
<td></td>
<td>• dst-sid—Destination module sub ID.</td>
</tr>
<tr>
<td></td>
<td>• msg-type—Message type.</td>
</tr>
<tr>
<td></td>
<td>• length—Message length.</td>
</tr>
<tr>
<td></td>
<td>• msg-seq—Message sequence number.</td>
</tr>
<tr>
<td>Resetting the send position</td>
<td>The sending position was reset because the replied serial number was wrong.</td>
</tr>
</tbody>
</table>
Table 25 describes output fields and messages for the **debugging ha event** command.

### Table 25 Output from the debugging ha event command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOT Resetting the send position</td>
<td>The sending position was not reset although the replied serial number was wrong.</td>
</tr>
<tr>
<td>Received seqnum seq-number, expected exp-number dropping...</td>
<td>The standby MPU discarded the packet because the serial number was wrong.</td>
</tr>
<tr>
<td>WARNING: IPC Queue is Full, Will be Resent Later size:%d MID=%x SID=%x</td>
<td>The IPC queue was full when HA was sending a message.</td>
</tr>
<tr>
<td>Event Write Failed .....</td>
<td>Failed to write events to the HA module.</td>
</tr>
<tr>
<td>Error in Creating HA Timer</td>
<td>HA failed to create a timer.</td>
</tr>
<tr>
<td>Error in Deleting HA Timer</td>
<td>HA failed to delete the timer.</td>
</tr>
<tr>
<td>HA_Send Message: IPC Allocation Failed....</td>
<td>HA failed to allocate memory when sending a message.</td>
</tr>
<tr>
<td>IPC Send Failed because IPC Queue is FULL</td>
<td>HA failed to send a message because the IPC queue was full.</td>
</tr>
<tr>
<td>HA Send ACK: IPC Alloc Failed...</td>
<td>HA failed to allocate memory when sending an acknowledgement.</td>
</tr>
<tr>
<td>HA Can’t allocate memory</td>
<td>HA failed to allocate memory.</td>
</tr>
<tr>
<td>IPC Create Failed for HA....</td>
<td>HA failed to create an IPC tunnel.</td>
</tr>
<tr>
<td>Register to IPC for Status Notification Failed....</td>
<td>HA failed to register with IPC.</td>
</tr>
<tr>
<td>HA Receives Unknown Message type from IPC...</td>
<td>HA received a message with unknown type from IPC.</td>
</tr>
<tr>
<td>Unknown Status message from IPC ..</td>
<td>HA received a message with unknown state from IPC.</td>
</tr>
<tr>
<td>HA Send Message: IPC Alloc Failed...</td>
<td>HA failed to allocate memory when sending a message.</td>
</tr>
<tr>
<td>ERROR:Send Message is NULL.</td>
<td>HA failed to send a message because the queue was empty.</td>
</tr>
<tr>
<td>IPC Send Failed because Bad parameter</td>
<td>IPC message sending failed because of parameter error.</td>
</tr>
</tbody>
</table>

Table 26 describes output fields and messages for the **debugging ha message** command.

### Table 26

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>%d:%d:%d %d/%d/%d HA: Notify module %lx.%lx to batch backup at %s.</td>
<td>HA notified each module when it bulk backed up data.</td>
</tr>
<tr>
<td>%d:%d:%d %d/%d/%d Notify module %lx.%lx to smooth at %s.</td>
<td>HA notified each module when it performed data smoothing.</td>
</tr>
</tbody>
</table>
Table 26 Output from the debugging ha message command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Sending Src MID = src-mid, Src SID = src-sid, Dst MID = dst-mid, Dst SID = dst-sid, type = msg-type, Len = length, Seq Num = msg-seq | HA sent a message to the destination module:  
  • src-mid—Source module ID.  
  • src-sid—Source module sub ID.  
  • dst-mid—Destination module ID.  
  • dst-sid—Destination module sub ID.  
  • msg-type—Message type.  
  • length—Message length.  
  • msg-seq—Message sequence number. |
| Deliver up Src MID = src-mid, Src SID = src-sid, Dst MID = dst-mid, Dst SID = dst-sid, type = msg-type, Len = length, Seq Num = msg-seq | HA sent a message to the application module. |
| From Module Src MID = src-mid, Src SID = src-sid, Dst MID = dst-mid, Dst SID = dst-sid, type = msg-type, Len = length, Seq Num = msg-seq | HA received a message from the application module. |

Table 27 describes output fields and messages for the debugging ha state command.

Table 27 Output from the debugging ha state command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPC Channel is down</td>
<td>IPC channel was down.</td>
</tr>
<tr>
<td>IPC Channel is UP</td>
<td>IPC channel was up.</td>
</tr>
<tr>
<td>FSM State: Waiting For Slave To Insert.</td>
<td>The active MPU is waiting for the standby MPU to be inserted into slot.</td>
</tr>
<tr>
<td>FSM State: Slave Have Been Inserted and Waiting for batch backup request from slave.</td>
<td>The standby MPU is inserted into the slot, and the active MPU is waiting for the standby MPU to send batch backup requests.</td>
</tr>
<tr>
<td>FSM State: Batch backup.</td>
<td>The active and standby MPUs are performing batch data backup.</td>
</tr>
<tr>
<td>FSM State: Realtime and routine backup.</td>
<td>The active MPU is performing real-time data backup.</td>
</tr>
<tr>
<td>FSM State: Data smooth.</td>
<td>The active MPU is performing data smoothing.</td>
</tr>
<tr>
<td>FSM State(slave): Ready.</td>
<td>The standby MPU is ready.</td>
</tr>
<tr>
<td>FSM State(slave): Send batch backup request to AMB</td>
<td>The standby MPU is sending a batch backup request to the active MPU.</td>
</tr>
<tr>
<td>FSM State(slave): Receiving batch data.</td>
<td>The standby MPU is receiving data in batches from the active MPU.</td>
</tr>
<tr>
<td>FSM State(slave): Receiving realtime and routine data.</td>
<td>The standby MPU is receiving real-time data from the active MPU.</td>
</tr>
</tbody>
</table>

Examples

# Enable active and standby switchover event debugging. When VLAN 3 is created on the device, output similar to the following example is generated:
<Sysname> debugging ha event
*Jan 29 14:56:54:32 2007 Sysname HA/7/DEBUG:From Module Src MID=2150000, Src SID=1, Dst MID=2150000, Dst SID=1, type=c, Len=124, Seq Num = 0
    // HA received a message from the VLAN module.
    # Enable active and standby switchover message debugging.
<Sysname> debugging ha message
*Jan 29 14:44:52:299 2007 Sysname HA/7/DEBUG:Deliver up Src MID=2800000, Src SID=0, Dst MID=3440000, Dst SID=0, type=b, Len=0, Seq Num = 0
    // HA sent a backup message to the application module.
    # Enable active and standby switchover state debugging.
<Sysname> debugging ha state
    // The HA state changed from batch backup to real-time backup.
Adjacency table debugging commands

The output description tables in this document only contain fields and messages that require an explanation.

IPv4 adjacency table debugging commands

debbuging adjacent-table all

Use `debugging adjacent-table all` to enable both IPv4 adjacency entry debugging and IPv4 adjacency table event debugging.

Use `undo debugging adjacent-table all` to disable IPv4 adjacency entry debugging and IPv4 adjacency table event debugging.

Syntax

`debugging adjacent-table all`
`undo debugging adjacent-table all`

Default

IPv4 adjacency entry debugging and IPv4 adjacency table event debugging are disabled.

Views

User view

Default command level

1: Monitor level

Usage guidelines

See Table 28 and Table 29 for a description of the command output fields.

debbuging adjacent-table entry

Use `debugging adjacent-table entry` to enable IPv4 adjacency entry debugging.

Use `undo debugging adjacent-table entry` to disable IPv4 adjacency entry debugging.

Syntax

`debugging adjacent-table entry`
`undo debugging adjacent-table entry`

Default

IPv4 adjacency entry debugging is disabled.

Views

User view
Default command level

1: Monitor level

Usage guidelines

Table 28 describes output fields and messages for the **debugging adjacent-table entry** command.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP Address</td>
<td>Next hop address of the outgoing interface of the adjacency table.</td>
</tr>
<tr>
<td>ulRouteIfIndex</td>
<td>Routing interface.</td>
</tr>
<tr>
<td>ulPhyIfIndex</td>
<td>Physical interface.</td>
</tr>
<tr>
<td>ulLogicIfIndex</td>
<td>Logical interface.</td>
</tr>
<tr>
<td>ucServiceType</td>
<td>Service type of the adjacency entry, such as PPP and IP over ATM.</td>
</tr>
<tr>
<td>ucLinkMediaType</td>
<td>Link type, such as P2P, broadcast, and NBMA.</td>
</tr>
<tr>
<td>ucActionType</td>
<td>Packet processing type, such as forwarding, discarding, and delivering to the CPU.</td>
</tr>
<tr>
<td>ulTimeStamp</td>
<td>Time stamp value of an entry.</td>
</tr>
<tr>
<td>ucslot</td>
<td>Slot number.</td>
</tr>
<tr>
<td>ucSynFlag</td>
<td>Flag indicating whether the entry is synchronized.</td>
</tr>
<tr>
<td>ulVirtualCircuitInfo</td>
<td>Virtual link information of the adjacency entry, such as PVC and DLCI.</td>
</tr>
<tr>
<td>ulMtu</td>
<td>MTU value.</td>
</tr>
<tr>
<td>LinkHeadLen(IP)</td>
<td>Length of the link layer header (IP).</td>
</tr>
<tr>
<td>LinkHeadLen(MPLS)</td>
<td>Length of the link layer header (MPLS).</td>
</tr>
<tr>
<td>LinkHead(IP)</td>
<td>Link layer header (IP).</td>
</tr>
<tr>
<td>LinkHead(MPLS)</td>
<td>Link layer header (MPLS).</td>
</tr>
</tbody>
</table>

Examples

# Enable IPv4 adjacency entry debugging.

```bash
< Sysname > debugging adjacent-table entry

*Aug 19 14:31:35:656 2007 Sysname ADJ4/7/Adj4_Entry_Add:
IP Address     : 0.0.0.0           ulRouteIfIndex       : S2/0
ulPhyIfIndex   : S2/0              ulLogicIfIndex       : N/A
ucServiceType  : PPP               ucLinkMediaType      : P2P
ucActionType   : FORWARDING       ulTimeStamp          : 402650
usSlot         : 0                 ulVirtualCircuitInfo : N/A
ulMtu          : 1500
LinkHeadLen(IP): 4                 LinkHeadLen(MPLS)    : 4
LinkHead(IP)   : ff030021
LinkHead(MPLS) : ff030281
```

75
// Debugging information for adding an adjacency entry.

Aug 19 14:28:39:875 2007 Sysname ADJ4/7/Adj4_Entry_Delete:

IP Address : 0.0.0.0   ulRouteIfIndex : S2/0
ulPhyIfIndex : S2/0   ulLogicIfIndex : N/A
ucServiceType : PPP   ucLinkMediaType : P2P
ucActionType : FORWARDING   ulTimeStamp : 99013
usSlot : 0   ulVirtualCircuitInfo : N/A
ulMtu : 1500
LinkHeadLen(IP) : 4   LinkHeadLen(MPLS) : 4
LinkHead(IP) : ff030021
LinkHead(MPLS) : ff030281

// Debugging information for deleting an adjacency entry.

debugging adjacent-table event

Use `debugging adjacent-table event` to enable IPv4 adjacency table event debugging.

Use `undo debugging adjacent-table event` to disable IPv4 adjacency table event debugging.

Syntax

`debugging adjacent-table event`

`undo debugging adjacent-table event`

Default

IPv4 adjacency table event debugging is disabled.

Views

User view

Default command level

1: Monitor level

Usage guidelines

Table 29 describes output fields and messages for the `debugging adjacent-table event` command.

Table 29 Output from the debugging adjacent-table event command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADJ4 ADD Entry Information</td>
<td>Information about the added entry.</td>
</tr>
<tr>
<td>IP Address</td>
<td>Next hop address of the outgoing interface of the adjacency table.</td>
</tr>
<tr>
<td>ulRouteIfIndex</td>
<td>Routing interface.</td>
</tr>
<tr>
<td>ulPhyIfIndex</td>
<td>Physical interface.</td>
</tr>
<tr>
<td>ulLogicIfIndex</td>
<td>Logical interface.</td>
</tr>
<tr>
<td>ucServiceType</td>
<td>Service type of the adjacency entry, such as PPP and IP over ATM. Only PPP is supported.</td>
</tr>
<tr>
<td>usSlot</td>
<td>Slot number.</td>
</tr>
</tbody>
</table>
### Field Description

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ucSynFlag</td>
<td>Flag indicating whether the entry is synchronized.</td>
</tr>
<tr>
<td>ulVirtualCircuitInfo</td>
<td>Virtual link information of the adjacency entry, such as PVC and DLCI. <strong>N/A</strong> is displayed for the virtual link of ATM, because the adjacency table does not support ATM.</td>
</tr>
<tr>
<td>ulMtu</td>
<td>MTU value.</td>
</tr>
<tr>
<td>ADJ4 slot insert(remove) information</td>
<td>Information of the card that is inserted or removed.</td>
</tr>
<tr>
<td>The Inserted(Removed) Slot ID is 2</td>
<td>Number of the slot where the card is inserted or removed.</td>
</tr>
</tbody>
</table>

### Examples

**# Enable IPv4 adjacency table event debugging.**

```plaintext
<Sysname> debugging adjacent-table event
*Jun 5 13:33:02:250 2007 Sysname ADJ4/7/Adj4_Event:
ADJ4 ADD Entry Information :
  IP Address    : 0.0.0.0       ulRouteIfIndex  : Pos1/0
  ulPhyIfIndex  : Pos1/0        ulLogicIfIndex : N/A
  ucServicType  : PPP            usSlot        : 1
  usMtu         : 4470           ulVirtualCircuitInfo : N/A
```

// IPv4 adjacency table added an entry.

```plaintext
*0.2407656 Sysname ADJ4/7/Adj4_Event:
ADJ4 slot insert information : The Inserted Slot ID is 2
```

// Debugging information for inserting a card.

```plaintext
*0.2272703 Sysname ADJ4/7/Adj4_Event:
ADJ4 slot remove information : The Removed Slot ID is 2
```

// Debugging information for removing a card.

### IPv6 adjacency table debugging commands

**debugging ipv6 adjacent-table all**

Use **debugging ipv6 adjacent-table all** to enable both IPv6 adjacency entry debugging and IPv6 adjacency table event debugging.

Use **undo debugging ipv6 adjacent-table all** to disable IPv6 adjacency entry debugging and IPv6 adjacency table event debugging.

**Syntax**

debugging ipv6 adjacent-table all

undo debugging ipv6 adjacent-table all

**Default**

IPv6 adjacency entry debugging and IPv6 adjacency table event debugging are disabled.
Views
User view

Default command level
1: Monitor level

Usage guidelines
See Table 30 and Table 31 for output fields and messages for the debugging command.

debbuging ipv6 adjacent-table entry

Use debuging ipv6 adjacent-table entry to enable IPv6 adjacency entry debugging.

Use undo debuging ipv6 adjacent-table entry to disable IPv6 adjacency entry debugging.

Syntax

debugging ipv6 adjacent-table entry
undo debugging ipv6 adjacent-table entry

Default
IPv6 adjacency entry debugging is disabled.

Views
User view

Default command level
1: Monitor level

Usage guidelines
Table 30 describes output fields and messages for the debuging ipv6 adjacent-table entry command.

Table 30 Output from the debuging ipv6 adjacent-table entry command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv6 Address</td>
<td>Next hop address of the outgoing interface of the IPv6 adjacency table.</td>
</tr>
<tr>
<td>ulRoutingIndex</td>
<td>IPv6 routing interface.</td>
</tr>
<tr>
<td>ulPhyIfIndex</td>
<td>IPv6 physical interface.</td>
</tr>
<tr>
<td>ulLogicIfIndex</td>
<td>IPv6 logical interface.</td>
</tr>
<tr>
<td>ucServiceType</td>
<td>Service type of the adjacency entry, such as PPP and IP over ATM. Only PPP is supported currently.</td>
</tr>
<tr>
<td>ucslot</td>
<td>Slot number.</td>
</tr>
<tr>
<td>ucSynFlag</td>
<td>Flag indicating whether the entry is synchronized.</td>
</tr>
<tr>
<td>ulVirtualCircuitInfo</td>
<td>Virtual link information of the IPv6 adjacency entry, such as PVC and DLCI. N/A is displayed for the virtual link of ATM, because the adjacency table does not support ATM.</td>
</tr>
<tr>
<td>ulMtu</td>
<td>MTU value.</td>
</tr>
</tbody>
</table>

78
Examples

# Enable IPv6 adjacency entry debugging.
<Sysname> debugging ipv6 adjacent-table entry
*Jun 5 11:19:37:250 2007 Sysname ADJ6/7/Adj6_Entry_Add:
IPv6 Address : N/A
ulRouteffIndex : Pos1/0 ulPhyfIndex : Pos1/0
ulLogicffIndex : N/A ucServiceType : PPP
ucSlot : 1 ucSynFlag : Y
ulVirtualCircuitInfo : N/A ulMtu : 4478

// IPv6 adjacency table added an entry.
*Jun 5 11:19:21:890 2007 Sysname ADJ6/7/Adj6_Entry_Delete:
IPv6 Address : N/A
ulRouteffIndex : Pos1/0 ulPhyfIndex : Pos1/0
ulLogicffIndex : N/A ucServiceType : PPP
ucSlot : 1 ucSynFlag : Y
ulVirtualCircuitInfo : N/A ulMtu : 4478

// IPv6 adjacency table deleted an entry.

debugging ipv6 adjacent-table event

Use debugging ipv6 adjacent-table event to enable IPv6 adjacency table event debugging.
Use undo debugging ipv6 adjacent-table event to disable IPv6 adjacency table event debugging.

Syntax

default ipv6 adjacent-table event

undo debugging ipv6 adjacent-table event

Default

IPv6 adjacency table event debugging is disabled.

Views

User view

Default command level

1: Monitor level

Usage guidelines

Table 31 describes output fields and messages for the debugging ipv6 adjacent-table event command.

Table 31 Output from the debugging ipv6 adjacent-table event command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADJ6 LinkUp Information</td>
<td>Link-up event information.</td>
</tr>
<tr>
<td>ADJ6 LinkDown Information</td>
<td>Link-down event information.</td>
</tr>
<tr>
<td>IPv6 Address</td>
<td>Next hop address of the outgoing interface of the IPv6 adjacency table.</td>
</tr>
</tbody>
</table>
### Field Description

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ulRouteIfIndex</td>
<td>IPv6 routing interface.</td>
</tr>
<tr>
<td>ulPhyIfIndex</td>
<td>IPv6 physical interface.</td>
</tr>
<tr>
<td>ulLogicIfIndex</td>
<td>IPv6 logical interface.</td>
</tr>
<tr>
<td>ucServiceType</td>
<td>Service type of the adjacency entry, such as PPP and IP over ATM. Only PPP is supported.</td>
</tr>
<tr>
<td>ulVirtualCircuitInfo</td>
<td>Virtual link information of the IPv6 adjacency entry, such as PVC and DLCI. N/A is displayed for the virtual link of ATM, because the adjacency table does not support ATM.</td>
</tr>
<tr>
<td>ulMtu</td>
<td>MTU value.</td>
</tr>
<tr>
<td>The Inserted(Removed) Slot ID is 2</td>
<td>Number of the slot where the card is inserted or removed.</td>
</tr>
</tbody>
</table>

### Examples

**# Enable IPv6 adjacency table event debugging.**

```bash
<Sysname> debugging ipv6 adjacent-table event
*Jun  5 11:19:37:226 2007 Sysname ADJ6/7/Adj6_Event:
  ADJ6 LinkUp Information :
    IPv6 Address         : N/A
    ulRouteIfIndex       : Pos1 /0         ulPhyIfIndex : Pos1/0
    ulLogicIfIndex       : N/A              ucServicType : PPP
    ucSlot               : 1                 ucSynFlag    : Y
    ulVirtualCircuitInfo : N/A               ulMtu        : 4478
```

**// Debugging information for a link-up event.**

```bash
*Jun  5 11:19:21:890 2007 Sysname ADJ6/7/Adj6_Event:
  ADJ6 LinkDown Information :
    IPv6 Address   : N/A
    ulRouteIfIndex : Pos1/0
```

**// Debugging information for a link-down event.**
Service flow redirection debugging commands

The data plane flow redirection module name is identified as "DPFLWRDR" in debugging messages.

The output description tables in this document only contain fields and messages that require an explanation.

debugging flow-redirect

Use debugging flow-redirect to enable debugging for service flow redirection.

Use undo debugging flow-redirect to disable debugging for service flow redirection.

Syntax

```
debugging flow-redirect { all | event | packet }
undo debugging flow-redirect { all | event | packet }
```

Default

Debugging for service flow redirection is disabled.

Views

User view

Parameters

- **all**: Specifies all types of debugging for service flow redirection.
- **event**: Specifies event debugging.
- **packet**: Specifies packet of debugging.

Usage guidelines

Table 32 describes output fields and messages for the debugging flow-redirect event command.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MBUF released (reason-code)</td>
<td>The data plane flow redirection module released the MBUF. The reason-code argument represents the reason:</td>
</tr>
<tr>
<td></td>
<td>• 1—Failed to apply for memory for the additional packet information of the session when processing outbound packets.</td>
</tr>
<tr>
<td></td>
<td>• 2—Failed to apply for memory for the additional information of the session when processing outbound packets.</td>
</tr>
<tr>
<td></td>
<td>• 3—Failed to add a Layer 2 packet header for packets initiatedly sent from other systems.</td>
</tr>
<tr>
<td>Received an original packet.</td>
<td>The data plane flow redirection module received an original packet returned from another system.</td>
</tr>
</tbody>
</table>

Table 33 describes output fields and messages for the debugging flow-redirect packet command.
Table 33 Output from the debugging flow-redirect packet command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Thread thread-id, received xxx packets: (src-ip-addr)-&gt;(dst-ip-addr), mm bytes.</td>
<td>Thread thread-id received mm bytes of xxx packets. The packet source IP address is src-ip-addr, destination IP address is dst-ip-addr. The xxx indicates the type of the received packets, including TCP RST, TCP RST, and other.</td>
</tr>
<tr>
<td>Thread thread-id, the packet from source-zone to destination-zone is permitted to be redirected: (src-ip-addr)-&gt;(dst-ip-addr), mm bytes, ACL acl-number.</td>
<td>Thread thread-id received a packet from source-zone to destination-zone, which is permitted to be redirected. The packet's source IP address is src-ip-addr, destination IP address is dst-ip-addr, length is mm bytes, and the ACL used is acl-number.</td>
</tr>
<tr>
<td>Thread thread-id, the packet from source-zone to destination-zone is denied to be redirected: (src-ip-addr)-&gt;(dst-ip-addr), mm bytes.</td>
<td>Thread thread-id received a packet from source-zone to destination-zone, which is not allowed to be redirected. The packet's source IP address is src-ip-addr, destination IP address is dst-ip-addr, and length is mm bytes.</td>
</tr>
<tr>
<td>Thread thread-id, the packet from source-zone to destination-zone is a no-session packet and does not redirect: (src-ip-addr)-&gt;(dst-ip-addr), mm bytes.</td>
<td>Thread thread-id received a packet from source-zone to destination-zone. The thread did not find a session for the packet and did not redirect the packet. The packet's source IP address is src-ip-addr, destination IP address is dst-ip-addr, and length is mm bytes.</td>
</tr>
</tbody>
</table>

Examples

The output in the following examples was created when the firewall receives packets from zone Untrust to zone Trust (which corresponds to segment ID 1) under the following conditions:

- An advanced security policy is configured on the firewall for in-depth security inspection.
- The packets match ACL 3005 referenced by the policy.

# Enable packet debugging for service flow redirection.
<Sysname> debugging flow-redirect packet

*Jun 25 10:38:49:761 2007 Sysname DPFLWRDR/7/DEBUG:
Thread 1, the packet from untrust to trust is permitted to redirect: (136.65.86.4)->(15.34.96.56), 852 bytes, ACL 3005.

// Thread 1 received a packet from zone Untrust to zone Trust, which is permitted to be redirected. The packet's source IP address is 136.65.86.4, destination IP address is 15.34.96.56, length is 852 bytes, and the ACL used is 3005.

# Enable event debugging for service flow redirection.
<Sysname> debugging flow-redirect event

*Jun 25 10:38:49:761 2007 Sysname DPFLWRDR/7/DEBUG:
Received an original packet.

// The DPFLWRDR module received an original packet returned from another system.
AFT debugging commands

The output description tables in this document only contain fields and messages that require an explanation.

debugging aft

Use `debugging aft` to enable AFT debugging.
Use `undo debugging aft` to disable AFT debugging.

Syntax

```
debugging aft { alg | all | event | packet } [ interface interface-type interface-number ]
undo debugging aft { alg | all | event | packet } [ interface interface-type interface-number ]
```

Default

AFT debugging is disabled.

Views

User view

Default command level

1: Monitor level

Parameters

- `alg`: Specifies AFT ALG debugging.
- `all`: Specifies all types of AFT debugging.
- `event`: Specifies AFT event debugging.
- `packet`: Specifies AFT packet debugging.
- `interface interface-type interface-number`: Specifies an interface by its type and number.

Usage guidelines

Table 34 describes output fields and messages for the `debugging aft alg` command.

**Table 34: Output from the debugging aft alg command (only for multi-core devices)**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Invalid FTP command type.</td>
<td>Invalid FTP command type.</td>
</tr>
<tr>
<td>Failed to process command PORT.</td>
<td>AFT failed to process PORT messages.</td>
</tr>
<tr>
<td>Failed to process command PASV.</td>
<td>AFT failed to process PASV messages.</td>
</tr>
<tr>
<td>Failed to process command 227.</td>
<td>AFT failed to process response 227.</td>
</tr>
<tr>
<td>Failed to process command EPRT.</td>
<td>AFT failed to process EPRT messages.</td>
</tr>
<tr>
<td>Failed to process command 229.</td>
<td>AFT failed to process response 229.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------------------------------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Failed to update FTP data.</td>
<td>AFT failed to update FTP data.</td>
</tr>
<tr>
<td>The SessionInfo does not exist.</td>
<td>Session information did not exist.</td>
</tr>
<tr>
<td>The SessionAttachInfo does not exist.</td>
<td>Session attach information did not exist.</td>
</tr>
<tr>
<td>Failed to translate ICMPv6 type and code.</td>
<td>AFT failed to translate the ICMPv6 type and code fields.</td>
</tr>
<tr>
<td>Failed to translate ICMPv6 inner IPv6 header.</td>
<td>AFT failed to translate the inner IPv6 header of an ICMPv6 packet.</td>
</tr>
<tr>
<td>Failed to translate ICMPv6 data.</td>
<td>AFT failed to translate the ICMPv6 data field.</td>
</tr>
<tr>
<td>Failed to translate ICMPv4 type and code.</td>
<td>AFT failed to translate the ICMPv4 type and code fields.</td>
</tr>
<tr>
<td>Failed to translate ICMPv4 inner IPv4 header.</td>
<td>AFT failed to translate the inner IPv4 header of an ICMPv4 packet.</td>
</tr>
<tr>
<td>Failed to translate ICMPv4 data.</td>
<td>AFT failed to translate the ICMPv4 data field.</td>
</tr>
<tr>
<td>Failed to process DNS-ALG from IPv6 to IPv4.</td>
<td>AFT failed to process IPv6-to-IPv4 DNS-ALG.</td>
</tr>
<tr>
<td>Failed to process DNS-ALG from IPv4 to IPv6.</td>
<td>AFT failed to process IPv4-to-IPv6 DNS-ALG.</td>
</tr>
<tr>
<td>Failed to process DNS question.</td>
<td>AFT failed to process DNS queries.</td>
</tr>
<tr>
<td>Failed to process DNS answer RR.</td>
<td>AFT failed to process the DNS answer resource record section.</td>
</tr>
<tr>
<td>Failed to process DNS authority RR.</td>
<td>AFT failed to process the DNS authoritative answer resource record section.</td>
</tr>
<tr>
<td>Failed to process DNS additional information RR.</td>
<td>AFT failed to process the DNS additional information resource record section.</td>
</tr>
<tr>
<td>Failed to copy data from buffer to MBuf.</td>
<td>AFT failed to copy data from buffer to MBuf.</td>
</tr>
<tr>
<td>Failed to translate domain name in DNS question of a PTR query or response from IPv4 to IPv6.</td>
<td>AFT failed to translate the domain name in an IPv4-to-IPv6 DNS PTR query or response.</td>
</tr>
<tr>
<td>Failed to translate domain name in DNS question of a PTR query or response from IPv6 to IPv4.</td>
<td>AFT failed to translate the domain name in an IPv6-to-IPv4 DNS PTR query or response.</td>
</tr>
<tr>
<td>Failed to process DNS resource data.</td>
<td>AFT failed to process DNS resource data.</td>
</tr>
</tbody>
</table>

Table 35 describes output fields and messages for the `debugging aft event` command.

**Table 35 Output from the debugging aft event command (only for multi-core devices)**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Invalid Mbuf.</td>
<td>Invalid Mbuf.</td>
</tr>
<tr>
<td>The packet is not the first fragment, dropping packet.</td>
<td>AFT discarded non-first-fragment packets.</td>
</tr>
<tr>
<td>Cannot get the outgoing interface. An ICMPv6 error generated.</td>
<td>Outgoing interface was not found, and an ICMPv6 error message was generated.</td>
</tr>
<tr>
<td>The outgoing interface index is invalid, dropping packet.</td>
<td>AFT dropped the packet because the index to the outgoing interface was invalid.</td>
</tr>
<tr>
<td>The outgoing interface is disabled, dropping packet.</td>
<td>The outgoing interface dropped the packet because ATP was disabled on the interface.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Failed to get the packet information, dropping packet.</td>
<td>AFT failed to read the packet and dropped it.</td>
</tr>
<tr>
<td>The SessionInfo or AttachInfo does not exist, dropping packet.</td>
<td>AFT dropped the packet because the SessionInfo or AttachInfo field of the packet did not exist.</td>
</tr>
<tr>
<td>Invalid IPv4 address, dropping packet.</td>
<td>AFT dropped the packet because its IPv4 address was invalid.</td>
</tr>
<tr>
<td>Unexpired source route option found. An ICMPv6 error generated.</td>
<td>Unexpired source route option was found, and an ICMPv6 error message was generated.</td>
</tr>
<tr>
<td>Hop Limit exceeded!</td>
<td>The hop limit of the packet was reached.</td>
</tr>
<tr>
<td>ICMPv6 fragment packet, dropping packet.</td>
<td>AFT dropped the fragmented ICMPv6 packet.</td>
</tr>
<tr>
<td>Authentication header or Encapsulating Security Payload header found.</td>
<td>Authentication header or Encapsulating Security Payload header was found.</td>
</tr>
<tr>
<td>The first TCP packet is non-SYN, dropping packet.</td>
<td>AFT dropped the first TCP packet because it was not a SYN packet.</td>
</tr>
<tr>
<td>Cannot get the mapped IPv4 source address, dropping packet.</td>
<td>AFT dropped the packet because no mapped IPv4 source address was found.</td>
</tr>
<tr>
<td>Dropping an IPv4 fragment packet.</td>
<td>AFT dropped the IPv4 packet because it was a fragment.</td>
</tr>
<tr>
<td>Cannot get the outgoing interface. An ICMPv4 error generated.</td>
<td>Outgoing interface was not found, and an ICMPv4 error message was generated.</td>
</tr>
<tr>
<td>Option field error.</td>
<td>Option field error was found.</td>
</tr>
<tr>
<td>Unexpired source route option found. An ICMPv4 error generated.</td>
<td>Unexpired source route option was found, and an ICMPv4 error message was generated.</td>
</tr>
<tr>
<td>TTL exceeded!</td>
<td>The hop limit of the packet was reached.</td>
</tr>
<tr>
<td>Cannot get the mapped IPv6 source address, dropping packet.</td>
<td>AFT dropped the packet because no mapped IPv6 source address was found.</td>
</tr>
<tr>
<td>The mapped IPv6 source address does not contain a configured prefix, dropping packet.</td>
<td>AFT dropped the packet because the mapped IPv6 source address did not contain a configured prefix.</td>
</tr>
<tr>
<td>Error occurred during ALG processing for IPv6 packet.</td>
<td>Error occurred during ALG processing an IPv6 packet.</td>
</tr>
<tr>
<td>Error occurred during ALG processing for IPv4 packet.</td>
<td>Error occurred during ALG processing an IPv4 packet.</td>
</tr>
<tr>
<td>Error occurred during creating V6 header and calculating checksum.</td>
<td>Error occurred during creating an IPv6 header and calculating checksum.</td>
</tr>
</tbody>
</table>

Table 36 describes output fields and messages for the `debugging aft packet` command.

Table 36 Output from the `debugging aft packet` command (only for multi-core devices)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pro</td>
<td>Protocol number of the packet.</td>
</tr>
<tr>
<td>TTL</td>
<td>TTL value.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| IPv6 packet is translated to IPv4 packet. | AFT translated the IPv6 packet into an IPv4 packet. Before address translation:  
  - ipv6-address1 — Source IPv6 address.  
  - port1 — Source port,  
  - ipv6-address2 — Destination IPv6 address.  
  - port2 — Destination port.  
After address translation:  
  - ipv4-address1 — Source IPv4 address.  
  - port3 — Source port.  
  - ipv4-address2 — Destination IPv4 address.  
  - port4 — Destination port.  
| IPv4 packet is translated to IPv6 packet. | AFT translated the IPv4 packet into an IPv6 packet. Before address translation:  
  - ipv4-address1 — Source IPv4 address.  
  - port1 — Source port.  
  - ipv4-address2 — Destination IPv4 address.  
  - port2 — Destination port.  
After address translation:  
  - ipv6-address1 — Source IPv6 address.  
  - port3 — Source port.  
  - ipv6-address2 — Destination IPv6 address.  
  - port4 — Destination port.  

Examples

# Enable the AFT debugging on the device with AFT enabled. When an IPv6 host pings an IPv4 host, output similar to the following example is generated:

```
<Sysname> debugging aft packet
*Dec 22 15:26:08:494 2010 Sysname AFT/7/PACKET:
  Pro : ICMP
  TTL : 63
  IPv6 packet is translated to IPv4 packet.
  (6:0:FF06:606:200:: : 43996 - 2000:0:404:402:: : 0)
  (6.6.6.2 : 43996 - 4.4.4.2 : 0)
// AFT received an ICMPv6 packet and translated the packet into an ICMPv4 packet.
```

# Enable AFT event debugging. Output similar to the following example is generated when an IPv6 host attempts to establish an FTP connection with an IPv4 host under the condition that AFT is disabled on the interface connected to the IPv6 network:

```
<Sysname> debugging aft packet
*Dec 22 15:26:08:494 2010 Sysname AFT/7/PACKET:
  Pro : ICMP
  TTL : 254
  IPv4 packet is translated to IPv6 packet.
  (4.4.4.2 : 0 - 6.6.6.2 : 43996)
// AFT received an ICMPv4 packet and translated the packet into an ICMPv6 packet.
```
debugging aft event

*Mar 10 14:34:02:547 2010 Sysname AFT/7/EVENT:
The outgoing interface is disabled, dropping packet.

// AFT dropped the packet because it was disabled on the outgoing interface.
ALG debugging commands

The output description tables in this document only contain fields and messages that require an explanation.

debugging alg

Use debugging alg to enable ALG debugging.
Use undo debugging alg to disable ALG debugging.

Syntax

debugging alg { event | packet }
undo debugging alg { event | packet }

Default
ALG debugging is disabled.

Views
User view

Default command level
2: System level

Parameters

event: Specifies debugging for events.
packet: Specifies debugging for packets.

Usage guidelines

Table 37 describes output fields and messages for the debugging alg event command.

Table 37 Output from the debugging alg event command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FTP FSM has changed from A to B.</td>
<td>The FTP state machine was changed from state A to state B. The states include: • FTP READY—Ready. • FTP_WAIT_USER_ACK—Waiting for acknowledgment of username. • FTP_USER_LOGGED—User login state. • FTP_WAIT_PASS_ACK—Waiting for acknowledgment of user password. • FTP_CONXN_UP—Connection established.</td>
</tr>
<tr>
<td>FTP FSM has not changed.</td>
<td>The state of the FTP state machine was not changed.</td>
</tr>
</tbody>
</table>
### Field Description

**NAT information:**
- **interface-type interface-number** — NAT applied interface.
- **VPN** — VPN with the index of vpnindex.
- **Pro** — NAT protocol.
- **Direction** — NAT direction (in/out).
- **original ip** — Original IP address.
- **original port** — Original port.
- **new ip** — IP address after NAT.
- **new port** — Port after NAT.

<table>
<thead>
<tr>
<th>Alloc IPC_RPC_MSG error.</th>
<th>ALG failed to apply for the RPC message.</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALG receive some error IPC message!</td>
<td>ALG received an incorrect IPC message.</td>
</tr>
<tr>
<td>ALG send config to No. slot timeout.</td>
<td>ALG failed to assign the configuration to the interface card on the specified slot.</td>
</tr>
<tr>
<td>Fail to Add Relation Table of QQ/MSN</td>
<td>ALG failed to create a QQ/MSN relation table.</td>
</tr>
<tr>
<td>Fail to add Relation Table of FTP</td>
<td>ALG failed to create an FTP relation table.</td>
</tr>
<tr>
<td>Fail to Attach ALG AttachInfo</td>
<td>ALG failed to attach ALG extended information.</td>
</tr>
</tbody>
</table>

#### NBT Session FSM has changed from A to B.

- **NBT READY** — The connection has been established after TCP challenge handshake, but the NBT session is not started.
- **NBT_WAIT_SESSION_ACK** — The client is waiting for the session response from the server.
- **NBT_SESSION** — The NBT session is being conducted and data is being transmitted.

#### RTSP FSM has changed from A to B.

- **RTSP INIT** — Initial state. The Setup packet is not sent yet.
- **RTSP READY** — Ready to transmit media streams.
- **RTSP PLAYING** — Playing state.

#### ILS FSM has changed from A to B.

- **ILS_WAIT_REQUEST** — Waiting for Request.
- **ILS_WAIT_RESPONSE** — Waiting for Response.
- **ILS_OPERATE_COMPLETE** — An operation completed.

#### H.225 FSM has changed from A to B.

- **H225_CALL_TCP_READY** — The TCP connection has been established, but Setup packet is not sent yet.
- **H225_CALL_DELIVERED** — The calling party sends the Setup packet, and waits for the connect response from the peer.
- **H225_CALL_ACTIVE** — The call is being conducted.
- **H225_CALL_RELEASED** — The call ends.
H.245 FSM has changed from A to B.

The H.245 state machine supports the following states:

- **H245_TCP_READY**—The TCP connection has been established, but H.245 packet is not sent yet.
- **H245_OPEN**—Begins capability exchange.
- **H245_CLOSE**—The H.245 session ends.

SIP FSM has changed from A to B.

The SIP state machine supports the following states:

- **SIP_READY**—Ready state. The party is ready to receive and initiate multimedia session.
- **SIP_CALL**—Calling state. The party is waiting for the acknowledgement from the calling party for the request.
- **SIP_COMMUNICATION**—The multimedia session has been established and is in the communication.
- **SIP_WAIT_RESPONSE**—Wait for response for non-INVITe request packets, such as REGISTER, OPTIONS, REFER, SUBSCRIBE, NOTIFY, MESSAGE, INFO, CANCEL, and BYE.

SCCP FSM has changed from A to B.

The SCCP state machine supports the following states:

- **SCCP_UNREGISTER**—The TCP connection has been established, but the SCCP client is not registered.
- **SCCP_REGISTERED**—The SCCP client is registered.

Table 38 describes output fields and messages for the debugging alg packet command.

### Table 38 Output from the debugging alg packet command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FTP Decode Error.</td>
<td>FTP packet decoding error.</td>
</tr>
<tr>
<td>FTP Encode Error.</td>
<td>FTP packet encoding error.</td>
</tr>
<tr>
<td>DNS Decode Error.</td>
<td>DNS packet decoding error.</td>
</tr>
<tr>
<td>DNS Encode Error.</td>
<td>DNS packet encoding error.</td>
</tr>
<tr>
<td>FTP FSM error, drop packet.</td>
<td>FTP state machine error. ALG discarded the packet.</td>
</tr>
<tr>
<td>Mbuf prepend head failed.</td>
<td>ALG failed to make MBUF header consecutive.</td>
</tr>
<tr>
<td>Received a NBT type REQUEST.</td>
<td>NBT request types:</td>
</tr>
<tr>
<td></td>
<td>• NAME REGISTRATION.</td>
</tr>
<tr>
<td></td>
<td>• NAME OVERWRITE.</td>
</tr>
<tr>
<td></td>
<td>• NAME REFRESH.</td>
</tr>
<tr>
<td></td>
<td>• NAME RELEASE.</td>
</tr>
<tr>
<td></td>
<td>• NAME QUERY.</td>
</tr>
<tr>
<td></td>
<td>• NODE STATUS.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| Received a NBT type RESPONSE. | NBT response types:  
  - POSITIVE NAME REGISTRATION.  
  - NEGATIVE NAME REGISTRATION.  
  - POSITIVE NAME RELEASE.  
  - NEGATIVE NAME RELEASE.  
  - POSITIVE NAME QUERY.  
  - NEGATIVE NAME QUERY.  
  - REDIRECT NAME QUERY.  
  - WAIT FOR ACKNOWLEDGEMENT.  
  - NODE STATUS.  
  - END-NODE CHALLENGE REGISTRATION. |
| NBT session FSM error, dropping packet. | NBT session service state machine error. ALG discarded the packet. |
| RTSP FSM error, dropping packet. | RTSP state machine error. ALG discarded the packet. |
| ILS FSM error, dropping packet. | ILS state machine error. ALG discarded the packet. |
| Received a H.225 type packet. | H.225 packet types:  
  - Setup.  
  - Call proceeding.  
  - Alerting.  
  - Connect.  
  - Progress.  
  - Facility. |
| No translation record, H.225 type address field is not translated. | H.225 address types:  
  - h245Address.  
  - srcCallSignalAddress.  
  - destCallSignalAddress.  
  - alternativeAddress.  
  - fastStart mediaChannel.  
  - fastStart mediaControlChannel.  
  - fastStart localAreaAddress.  
  - h245Control mediaChannel.  
  - h245Control mediaControlChannel.  
  - h245Control localAreaAddress. |
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Received a SIP type REQUEST.</td>
<td>SIP request types:</td>
</tr>
<tr>
<td></td>
<td>• INVITE.</td>
</tr>
<tr>
<td></td>
<td>• ACK.</td>
</tr>
<tr>
<td></td>
<td>• OPTIONS.</td>
</tr>
<tr>
<td></td>
<td>• BYE.</td>
</tr>
<tr>
<td></td>
<td>• CANCEL.</td>
</tr>
<tr>
<td></td>
<td>• REGISTER.</td>
</tr>
<tr>
<td></td>
<td>• INFO.</td>
</tr>
<tr>
<td></td>
<td>• PRACK.</td>
</tr>
<tr>
<td></td>
<td>• SUBSCRIBE.</td>
</tr>
<tr>
<td></td>
<td>• NOTIFY.</td>
</tr>
<tr>
<td></td>
<td>• UPDATE MESSAGE.</td>
</tr>
<tr>
<td></td>
<td>• UNSUBSCRIBE.</td>
</tr>
<tr>
<td></td>
<td>• REFER.</td>
</tr>
<tr>
<td></td>
<td>• PUBLISH.</td>
</tr>
<tr>
<td></td>
<td>• FEATURE.</td>
</tr>
<tr>
<td>ALG received a SIP response. The packet type can be:</td>
<td>• INFORMATIONAL.</td>
</tr>
<tr>
<td>Received a SIP type RESPONSE.</td>
<td>• SUCCESSFUL.</td>
</tr>
<tr>
<td></td>
<td>• REDIRECTION.</td>
</tr>
<tr>
<td></td>
<td>• CLIENT ERROR.</td>
</tr>
<tr>
<td></td>
<td>• SERVER ERROR.</td>
</tr>
<tr>
<td></td>
<td>• GLOBAL FAILURE.</td>
</tr>
<tr>
<td>SCCP FSM error.</td>
<td>SCCP state machine error.</td>
</tr>
<tr>
<td>No translation record, SCCP type packet was not translated.</td>
<td>ALG failed to translate the SCCP packet. The packet type can be:</td>
</tr>
<tr>
<td></td>
<td>• Register.</td>
</tr>
<tr>
<td></td>
<td>• IPPort.</td>
</tr>
<tr>
<td></td>
<td>• OpenReceiveChannelAck.</td>
</tr>
<tr>
<td></td>
<td>• StartMediaTransmission.</td>
</tr>
<tr>
<td></td>
<td>• OpenMultiMediaReceiveChannelAck.</td>
</tr>
<tr>
<td></td>
<td>• StartMultiMediaTransmission.</td>
</tr>
<tr>
<td>Failed to attach ALG type attachinfo of SCCP.</td>
<td>ALG failed to attach information to the SCCP packet. The packet type can be:</td>
</tr>
<tr>
<td></td>
<td>• Register.</td>
</tr>
<tr>
<td></td>
<td>• IPPort.</td>
</tr>
<tr>
<td></td>
<td>• OpenReceiveChannelAck.</td>
</tr>
<tr>
<td></td>
<td>• StartMediaTransmission.</td>
</tr>
<tr>
<td></td>
<td>• OpenMultiMediaReceiveChannelAck.</td>
</tr>
<tr>
<td></td>
<td>• StartMultiMediaTransmission.</td>
</tr>
</tbody>
</table>
### Field Description

**Received SCCP type packet.**

ALG received an SCCP packet. The packet type can be:
- Register.
- IPPort.
- OpenReceiveChannelAck.
- StartMediaTransmission.
- RegisterAck.
- Keepalive.
- UnRegister.
- KeepaliveACK.
- OpenMultiMediaReceiveChannelAck.
- StartMultiMediaTransmission.

**Received SCCP type CallState packet.**

ALG received an SCCP CallState packet. The packet type can be:
- OffHook.
- OnHook.
- RingOut.
- RingIn.
- Connected.
- Busy.
- Congestion.
- Hold.
- CallWaiting.
- CallTransfer.
- CallPark.
- Proceed.
- CallRemoteMultiline.
- InvalidNumber.

**Received SCCP packet (Type: id).**

ALG received an SCCP packet with the type ID of id.

### Examples

# Enable ALG event debugging on the device with FTP ALG enabled. Output similar to the following example is generated when an FTP client on the private network performs these tasks:

- Accesses an FTP server located on the public network.
- Uses the correct username and password to log in to the sever.
- Sends a PORT command for data transfer.

```plaintext
<Sysname> debugging alg event
*0.856360 Sysname ALG/8/EVENTS:
FTP FSM has changed from FTP_READY to FTP_WAIT_USER_ACK
FTP FSM has changed from FTP_WAIT_USER_ACK to FTP_USER_LOGGED
FTP FSM has changed from FTP_USER_LOGGED to FTP_WAIT_PASS_ACK
FTP FSM has changed from FTP_WAIT_PASS_ACK to FTP_CONNECTUP

// Transit process of the FTP state machine at the user's login.
(Ethernet0/1/2) from VPN : 1, Pro : TCP, Direction : OUT
(1.1.1.1: 1025 ) ----> (5.5.5.7: 12282 )
```

93
The ALG module obtained NAT information in the packet payload after the PORT command was executed.

# Enable ALG packet debugging on the device with DNS ALG enabled. Output similar to the following example is generated when a host on the private network accesses a DNS server on the public network in type-A query mode. In this mode, the host queries an IPv4 address by domain name.

```
<Sysname> debugging alg packet
  *0.418080 Sysname ALG/8/PACKETS: Receive a DNS query packet.
```

// The ALG module received a DNS query packet.

# Enable ALG packet debugging on the device with NBT ALG enabled. Output similar to the following example is generated when a host on the private network pings the hostname of another host on the public network.

```
<Sysname> debugging alg packet
  *Aug 24 10:50:58:500 2007 Sysname DPALG/7/debug:
    Received a NBT NAME QUERY REQUEST.
```

// The ALG module received an NBT query request.

# Enable ALG packet debugging on the device with SQLNET ALG enabled. Output similar to the following example is generated when an Oracle client on the public network tests the database connection in its network configuration wizard.

```
<Sysname> debugging alg packet
  *Aug 24 10:50:58:500 2007 Sysname DPALG/7/debug:
    Received a SQLNET CONNECT REQUEST.
```

// The ALG module received an SQLNET connection request.

```
<Sysname> debugging alg packet
  *Aug 24 10:51:58:500 2007 Sysname DPALG/7/debug:
    Received a SQLNET REDIRECT RESPONSE.
```

// The ALG module received an SQLNET redirect response.

# Enable ALG event debugging on the device with RTSP ALG enabled. Output similar to the following example is generated when an RTSP client opens the URL of a media file under these conditions:

- The RTSP client is RealPlayer software on the private network.
- The URL contains the IP address of an RTSP server and the name of the media file.

```
<Sysname> debugging alg event
  *Aug 24 10:50:58:500 2007 Sysname DPALG/7/debug:
    From VPN : 1, Pro : TCP
    Direction : OUT
    ( 1.1.1.2: 1000 ) ----> ( 2.2.2.9: 2010 )
```

// After the dclient sent the setup command, the ALG module translated the address in the packet payload.

# Enable ALG packet debugging on the device with ILS ALG enabled. Output similar to the following example is generated when NetMeeting on the private network sends an AddRequest message to a public network server for registration.

```
<Sysname> debugging alg packet
  *Aug 24 10:50:58:500 2007 Sysname DPALG/7/debug:
    Received an ILS AddRequest packet.
```

// The ALG module received an ILS registration request.
# Enable ALG event debugging on the device with H.323 ALG enabled. Output similar to the following example is generated when an H.323 terminal on the private network calls another H.323 terminal on the public network.

```<Sysname> debugging alg event
*Aug 24 10:50:58:500 2007 Sysname DPALG/7/debug:
H.225 FSM has changed from H225_CALL_TCP_READY to H225_CALL_DELIVERED.
```

// The ALG module received a setup packet from the calling party and its state changed from H225_CALL_TCP_READY to H225_CALL_DELIVERED.

```*Aug 24 10:51:02:500 2007 Sysname DPALG/7/debug:
H.225 FSM has changed from H225_CALL_DELIVERED to H225_CALL_ACTIVE.
```

// The ALG module received a connect packet from the called party and its state changed from H225_CALL_DELIVERED to H225_CALL_ACTIVE.
ARP debugging commands

The output description tables in this document only contain fields and messages that require an explanation.

debugging arp packet

Use debugging arp packet to enable ARP packet debugging.

Use undo debugging arp packet to disable ARP packet debugging.

Syntax

debugging arp packet [ ipacl ipacl-number | macacl macacl-number ] *

undo debugging arp packet

Default

ARP packet debugging is disabled.

Views

User view

Default command level

1: Monitor level

Parameters

ipacl: Checks the sender and target IP addresses of ARP packets.

ipacl-number: Specifies an advanced ACL by its number, in the range of 3000 to 3999.

macacl: Checks the sender and target MAC addresses of ARP packets.

macacl-number: Specifies a Layer 2 ACL by its number, in the range of 4000 to 4999.

Usage guidelines

Table 39 describes output fields and messages for the debugging arp packet command.

Table 39 Output from the debugging arp packet command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>arp_send:Send an ARP Packet</td>
<td>ARP sent an ARP packet.</td>
</tr>
<tr>
<td>arp_rcv:Receive an ARP Packet</td>
<td>ARP received an ARP packet.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Operation</th>
<th>Packet type:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• 1—ARP request.</td>
</tr>
<tr>
<td></td>
<td>• 2—ARP reply.</td>
</tr>
</tbody>
</table>

Examples

# Enable ARP packet debugging. When the device is pinged from a directly connected device, output similar to the following example is generated:

<Sysname> debugging arp packet
debugging arp status

Use **debugging arp status** to enable ARP status debugging.

Use **undo debugging arp status** to disable ARP status debugging.

**Syntax**

**debugging arp status**

**undo debugging arp status**

**Default**

ARP status debugging is disabled.

**Views**

User view

**Default command level**

1: Monitor level

**Usage guidelines**

You can use this command to display the status of ARP entries.

Table 40 describes output fields and messages for the **debugging arp status** command.
### Table 40 Output from the debugging arp status command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARP Item Status Change</td>
<td>Status change of the ARP entry.</td>
</tr>
<tr>
<td>eth_addr</td>
<td>Hardware address of the ARP entry.</td>
</tr>
<tr>
<td>ip_addr</td>
<td>IP address of the ARP entry.</td>
</tr>
</tbody>
</table>

**Example**

```
# Enable ARP status debugging. When the device pings a directly connected device, output similar to the following example is generated:
<Sysname> debugging arp status
<Sysname> ping -c 1 10.11.113.110
  PING 10.11.113.110: 56 data bytes, press CTRL_C to break
  *Apr 19 17:02:22:153 2006 Sysname ARP/8/arp_status_change:;ARP Item Status Change :
  eth_addr : 000f-1f9b-7fad, ip_addr : 10.11.113.110, INITIALIZE -> NO_AGE
  Reply from 10.11.113.110: bytes=56 Sequence=1 ttl=128 time=79 ms

  --- 10.11.113.110 ping statistics ---
  // ARP changed the status of the ARP entry with IP address 10.11.113.110 from INITIALIZE to NO_AGE.
```
ARP snooping debugging commands

The output description tables in this document only contain fields and messages that require an explanation.

debugging arp-snooping

Use **debugging arp-snooping** to enable ARP snooping debugging.
Use **undo debugging arp-snooping** to disable ARP snooping debugging.

**Syntax**

Centralized devices:
```
debugging arp-snooping { all | error | info }
undo debugging arp-snooping { all | error | info }
```

Distributed devices:
```
debugging arp-snooping { all | error | info | sync }
undo debugging arp-snooping { all | error | info | sync }
```

**Default**

ARP snooping debugging is disabled.

**Views**

User view

**Default command level**
1: Monitor level

**Parameters**

- **all**: Specifies all types of ARP snooping debugging.
- **error**: Specifies ARP snooping error debugging.
- **info**: Specifies ARP snooping information debugging.
- **sync**: Specifies ARP snooping data synchronization debugging.

**Usage guidelines**

Table 41 describes output fields and messages for the **debugging arp-snooping error** command.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Failed to vlan-mapping conversion when sending packet</td>
<td>VLAN mapping failed.</td>
</tr>
<tr>
<td>Failed to send packet by VLAN component</td>
<td>ARP failed to send packets by VLAN component.</td>
</tr>
<tr>
<td>VLAN component is null</td>
<td>ARP failed to send the packet because the VLAN component was null.</td>
</tr>
</tbody>
</table>
Table 42 describes output fields and messages for the **debugging arp-snooping info** command.

**Table 42 Output from the debugging arp-snooping info command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Receive an ARP and begin to modify existed entry</td>
<td>ARP updated the existing entry.</td>
</tr>
<tr>
<td>Receive an ARP and begin to add one entry</td>
<td>ARP added an ARP entry.</td>
</tr>
<tr>
<td>TCN VLAN bitmap:</td>
<td></td>
</tr>
<tr>
<td>xx xx xx xx</td>
<td></td>
</tr>
<tr>
<td>Port num: n</td>
<td></td>
</tr>
<tr>
<td>Port list:</td>
<td></td>
</tr>
<tr>
<td>0xn 0xn 0xn 0xn 0xn 0xn 0xn 0xn</td>
<td></td>
</tr>
<tr>
<td>0xn 0xn 0xn 0xn 0xn 0xn 0xn 0xn</td>
<td></td>
</tr>
<tr>
<td>Content of the TCN message.</td>
<td></td>
</tr>
</tbody>
</table>

Table 43 describes output fields and messages for the **debugging arp-snooping sync** command.

**Table 43 Output from the debugging arp-snooping sync command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Begin to send arp snooping IPC message:</td>
<td></td>
</tr>
<tr>
<td>Command type: 0xn</td>
<td>Data sent in the ARP snooping distributed IPC synchronization, including the command type, source card number, destination card group number, and length of the command.</td>
</tr>
<tr>
<td>Src board: 0xn</td>
<td></td>
</tr>
<tr>
<td>Dst board group: 0xn</td>
<td></td>
</tr>
<tr>
<td>Command count: n</td>
<td></td>
</tr>
<tr>
<td>Command length: 0xn</td>
<td></td>
</tr>
<tr>
<td>Begin to sync arp snooping configuration to inserted slot</td>
<td>ARP began distributed VLAN bitmap synchronization.</td>
</tr>
<tr>
<td>Receive arp snooping configuration from mainboard</td>
<td>ARP received a distributed VLAN bitmap synchronization message.</td>
</tr>
<tr>
<td>Receive TCN message with port number: n</td>
<td>The MPU received a TCN message.</td>
</tr>
<tr>
<td>Send TCN message with port number: n</td>
<td>ARP synchronized TCN from the interface card to the MPU.</td>
</tr>
<tr>
<td>Begin to create sync timer</td>
<td>ARP created the data synchronization timer.</td>
</tr>
</tbody>
</table>

**Examples**

```
# Enable ARP snooping information debugging. The output in this example was created when ARP snooping and ARP fast-reply are both enabled.
<Sysname> debugging arp-snooping info
<Sysname> terminal debugging
% Current terminal debugging is on
// ARP snooping information debugging was enabled.
* Jul 30 09:05:39:844 2007 Sysname ARP/7/Debug_ARP_Snooping_Information:
ARP_Snooping Information:
Receive an ARP and begin to modify existed entry
// ARP snooping received an ARP packet and updated the existing ARP snooping entry.
```
ARP snooping begin to forward the packet

// ARP snooping forwarded the ARP packet.

ARP snooping received an ARP reply and creates a new entry to save the resolution result.
The output description tables in this document only contain fields and messages that require an explanation.

**debugging arp fast-reply**

Use `debugging arp fast-reply` to enable ARP fast-reply debugging.

Use `undo debugging arp fast-reply` to disable ARP fast-reply debugging.

**Syntax**

```plaintext
debugging arp fast-reply { all | error | packet | info }
undo debugging arp fast-reply { all | error | packet | info }
```

**Default**

ARP fast-reply debugging is disabled.

**Views**

User view

**Default command level**

1: Monitor level

**Parameters**

all: Specifies all types of ARP fast-reply debugging.

error: Specifies ARP fast-reply error debugging.

packet: Specifies ARP fast-reply packet debugging.

info: Specifies ARP fast-reply information debugging.

**Usage guidelines**

Table 44 describes output fields and messages for the `debugging arp fast-reply error` command.

**Table 44 Output from the debugging arp fast-reply error command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Failed to send packet by driver</td>
<td>ARP failed to send an ARP packet by driver.</td>
</tr>
<tr>
<td>Invalid ifindex or interface type</td>
<td>Invalid interface index or interface type.</td>
</tr>
</tbody>
</table>

Table 45 describes output fields and messages for the `debugging arp fast-reply packet` command.
Table 45 Output from the debugging arp fast-reply packet command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Receive ARP packet:</td>
<td>ARP received ARP packets. The received ARP packets include the following information:</td>
</tr>
<tr>
<td>Src Interface : ifName</td>
<td>• Sending interface.</td>
</tr>
<tr>
<td>VLAN ID : vlan id</td>
<td>• VLAN to which the sending interface belongs.</td>
</tr>
<tr>
<td>SrcArpMac : xxxxxxxxxx</td>
<td>• Sender IP and MAC addresses.</td>
</tr>
<tr>
<td>DstArpMac : xxxxxxxxxx</td>
<td>• Target IP and MAC addresses.</td>
</tr>
<tr>
<td>SrcEthMac : xxxxxxxxxx</td>
<td>• Source and destination Ethernet MAC addresses.</td>
</tr>
<tr>
<td>DstEthMac : xxxxxxxxxx</td>
<td>• Packet type.</td>
</tr>
<tr>
<td>PacketType : packetType</td>
<td></td>
</tr>
</tbody>
</table>

Send ARP packet:

<table>
<thead>
<tr>
<th>VLAN ID : vlan id</th>
<th>ARP sent ARP packets. The sent ARP packets include the following information:</th>
</tr>
</thead>
<tbody>
<tr>
<td>SrcArpMac : xxxxxxxxxx</td>
<td>• VLAN to which the sending interface belongs.</td>
</tr>
<tr>
<td>DstArpMac : xxxxxxxxxx</td>
<td>• Sender IP and MAC addresses.</td>
</tr>
<tr>
<td>SrcEthMac : xxxxxxxxxx</td>
<td>• Target IP and MAC addresses.</td>
</tr>
<tr>
<td>DstEthMac : xxxxxxxxxx</td>
<td>• Source and destination Ethernet MAC addresses.</td>
</tr>
<tr>
<td>PacketType : packetType</td>
<td></td>
</tr>
</tbody>
</table>

Table 46 describes output fields and messages for the `debugging arp fast-reply info` command.

Table 46 Output from the debugging arp fast-reply info command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Self ip address</td>
<td>IP conflict detected.</td>
</tr>
<tr>
<td>Get info from DHCP Snooping:</td>
<td>ARP obtained information from a DHCP snooping entry.</td>
</tr>
<tr>
<td>VLAN ID: vlanid</td>
<td></td>
</tr>
<tr>
<td>Port: ifName</td>
<td></td>
</tr>
<tr>
<td>IP Addr: x.x.x.x</td>
<td></td>
</tr>
<tr>
<td>MAC Addr: xxxxxxxxxx</td>
<td></td>
</tr>
<tr>
<td>Get info from ARP Snooping:</td>
<td>ARP obtained information from an ARP snooping entry.</td>
</tr>
<tr>
<td>VLAN ID: vlanid</td>
<td></td>
</tr>
<tr>
<td>Port: ifName</td>
<td></td>
</tr>
<tr>
<td>IP Addr: x.x.x.x</td>
<td></td>
</tr>
<tr>
<td>MAC Addr: xxxxxxxxxx</td>
<td></td>
</tr>
<tr>
<td>ARP fast-reply deliver packet to lower priority feature</td>
<td>ARP delivered the packet to the feature with a lower priority.</td>
</tr>
</tbody>
</table>

Examples

```
# Enable all types of ARP fast-reply debugging. The output in this example was created when ARP snooping and ARP fast-reply are both enabled:
<Sysname> debugging arp fast-reply all
<Sysname> terminal debugging
% Current terminal debugging is on
// All types of debugging for ARP fast-reply were enabled.
*Sep 12 11:40:56:62 2007 Sysname ARP/7/Debug_ARP_FastReply_Packet:
ARP_FastReply Packet:
Receive ARP packet:
Src Interface : Ethernet1/2 Vlan ID :1
SrcArpMac :0000-0000-0012 SrcIp :6.0.0.2
```

103
// ARP received an ARP request.
*Sep 12 11:42:33:969 2007 Sysname ARP/7/Debug_ARP_FastReply_Information:
ARP_FastReply Information:
Get info from ARP Snooping:
VLAN ID: 1,    Port: Ethernet1/2
IP Addr: 6.0.0.5,    MAC Addr: 0000-0000-0012,
// ARP found a matching entry in the ARP snooping table.
*Sep 12 14:03:20:437 2007 Sysname ARP/7/Debug_ARP_FastReply_Packet:
ARP_FastReply Packet:
Send ARP packet:     Vlan ID :1
SrcArpMac :0000-0000-0012    SrcIp :6.0.0.2
DstArpMac :0000-0000-0000    DstIp :6.0.0.5
SrcEthMac :0000-0000-0012    DstEthMac :ffff-ffff-ffff
PacketType :REQUEST
// ARP forwarded the ARP request.
ATM debugging commands

The output description tables in this document only contain fields and messages that require an explanation.

**debugging atm all**

Use **debugging atm all** to enable all types of ATM debugging.

Use **undo debugging atm all** to disable all types of ATM debugging.

**Syntax**

```plaintext
deedgling atm all
undo debugging atm all
```

**Default**

All types of ATM debugging are disabled.

**Views**

User view

**Default command level**

1: Monitor level

**Usage guidelines**

When all types of ATM debugging are enabled, a large amount of output information is generated. ATM debugging may also disable users from managing network devices through terminals and may decrease the packet forwarding performance considerably.

Besides outputting debugging information such as ATM errors, OAM, events, and packets, this command also displays information exchanged between upper-layer software and hardware interfaces (for example, PVC parameters information sent to the hardware when a PVC is created). Run this command if you need to view the information exchanged through drive interfaces between platform software and bottom-level hardware.

**Related commands**

- **debugging atm error**
- **debugging atm oam**
- **debugging atm event**
- **debugging atm packet**

**debugging atm error**

Use **debugging atm error** to enable debugging for ATM errors.

Use **undo debugging atm error** to disable debugging for ATM errors.

**Syntax**

```plaintext
deedgling atm error [ interface interface-type interface-number [ pvc { pvc-name | vpi/vci } ] ]
```

105
undo debugging atm error [ interface interface-type interface-number [ pvc { pvc-name | vpi/vci } ] ]

**Default**
Debugging for ATM errors is disabled.

**Views**
User view

**Default command level**
1: Monitor level

**Parameters**

`interface-type interface-number`: Specifies an interface by its type and number.

`pvc-name`: PVC name, a string of 1 to 16 characters. A PVC name is unique on an ATM interface and is not case-sensitive. It cannot be a valid VPI/VCI pair (such as `1/20`).

`vpi/vci`: ATM virtual path identifier (VPI) and ATM virtual channel identifier (VCI). The `vpi` argument ranges from 0 to 255. The range of the `vci` argument varies with interface types, as listed in Table 47. Normally, VCI 0 through VCI 31 are reserved for special purposes.

**Table 47 Interface type and VCI range**

<table>
<thead>
<tr>
<th>Interface type</th>
<th>VCI range</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADSL 2+</td>
<td>0 to 255</td>
</tr>
<tr>
<td>G.SHDSL</td>
<td>0 to 255</td>
</tr>
<tr>
<td>ATM OC3</td>
<td>0 to 1023</td>
</tr>
<tr>
<td>ATM E1</td>
<td>0 to 511</td>
</tr>
<tr>
<td>ATM T1</td>
<td>0 to 511</td>
</tr>
<tr>
<td>ATM E3</td>
<td>0 to 1023</td>
</tr>
<tr>
<td>ATM T3</td>
<td>0 to 1023</td>
</tr>
</tbody>
</table>

**Usage guidelines**
If no interface is specified, the operation is performed on all interfaces.

If you do not specify the PVC name or VPI/VCI pair, the operation applies to all PVCs.

Table 48 describes output fields and messages for the `debugging atm error` command.

**Table 48 Output from the debugging atm error command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>interface name</td>
<td>Interface name.</td>
</tr>
<tr>
<td>pvc VPI/VCI</td>
<td>VPI/VCI pair.</td>
</tr>
<tr>
<td>error code</td>
<td>Error description and error code.</td>
</tr>
</tbody>
</table>

**Examples**

# Enable all types of debugging for ATM errors on all ATM interfaces on Router B. Output similar to the following example is generated when the following conditions exist:

- Router A and Router B are connected through their ATM interfaces.
- Only one side of the connection is assigned an IP address.

  Router A is configured as follows:
  
  ```
  <RouterA> system-view
  [RouterA] interface Atm5/0
  [RouterA-Atm5/0] pvc 10/33
  [RouterA-atm-pvc-Atm1/10/33] map ip inarp
  ```

- Router B is configured as follows:

  ```
  <RouterB> system-view
  [RouterB] interface Atm6/0
  [RouterB-Atm6/0] pvc 10/33
  [RouterB-atm-pvc-Atm1/10/33] map ip inarp
  [RouterB-atm-pvc-Atm1/10/33] quit
  [RouterB-Atm6/0] ip address 100.1.1.2 255.255.255.0
  ```

  ```
  <RouterB> debugging atm error
  *0.409432803 RouterB ATM/8/debug8: Atm5/0 pvc 10/33 err: send inarp fail, intf no ip(code 48)
  ```

  // ATM failed to send InARP packets due to an error with error code 48 (which indicates no IP address is found).

**debugging atm event**

Use `debugging atm event` to enable debugging for ATM events.

Use `undo debugging atm event` to disable debugging for ATM events.

**Syntax**

```
debugging atm event [ interface interface-type interface-number [ pvc { pvc-name [ vpi/vci ] | vpi/vci } ] ]
undo debugging atm event [ interface interface-type interface-number [ pvc { pvc-name [ vpi/vci ] | vpi/vci } ] ]
```

**Default**

Debugging for ATM events is disabled.

**Views**

User view

**Default command level**

1: Monitor level

**Parameters**

- `interface-type interface-number`: Specifies an interface by its type and number.
- `pvc-name`: PVC name, a string of 1 to 16 characters. A PVC name is unique on an ATM interface and is not case-sensitive. It cannot be a valid VPI/VCI pair (such as 1/20).
- `vpi/vci`: ATM virtual path identifier (VPI) and ATM virtual channel identifier (VCI). The `vpi` argument ranges from 0 to 255. The range of the `vci` argument varies with interface types, as listed in Table 47. Normally, VCI 0 through VCI 31 are reserved for special purposes.

**Usage guidelines**

If no interface is specified, the operation applies to all ATM interfaces.
If you do not specify the PVC name or VPI/VCI pair, the operation applies to all PVCs.

Table 49 describes output fields and messages for the `debugging atm event` command.

### Table 49 Output from the debugging atm event command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>interface name</td>
<td>Interface name.</td>
</tr>
<tr>
<td>pvc VPI/VCI</td>
<td>VPI/VCI pair.</td>
</tr>
<tr>
<td>event</td>
<td>Event description.</td>
</tr>
<tr>
<td>no local ip, in inarp msg ignored</td>
<td>An InARP packet was dropped because the local IP address is not available.</td>
</tr>
<tr>
<td>inarp timeout, map cleared for no reply</td>
<td>An InARP packet timed out.</td>
</tr>
<tr>
<td>sub-interface deleted</td>
<td>A sub-interface was removed.</td>
</tr>
<tr>
<td>sub-interface down</td>
<td>A sub-interface went down.</td>
</tr>
<tr>
<td>sub-interface up</td>
<td>A sub-interface went up.</td>
</tr>
<tr>
<td>Received wrong oam ping reply</td>
<td>An OAM cell was received.</td>
</tr>
<tr>
<td>OAM change state to up</td>
<td>The OAM state changed to up.</td>
</tr>
<tr>
<td>OAM change state to down</td>
<td>The OAM state changed to down.</td>
</tr>
</tbody>
</table>

### Examples

```bash
# Enable debugging for ATM events on all ATM interfaces on Router A. Output similar to the following example is generated when ATM 5/0 is shut down on Router A under the following conditions:
- Router A and Router B are connected through their ATM interfaces.
- Router A is configured as follows:
  ```bash
  <RouterA> system-view
  [RouterA] interface Atm5/0
  [RouterA-Atm5/0] pvc 10/33
  [RouterA-atm-pvc-Atm1/10/33] map ip inarp
  [RouterA-atm-pvc-Atm1/10/33] quit
  [RouterA-Atm5/0] ip address 100.1.1.1 255.255.255.0
  ```
- Router B is configured as follows:
  ```bash
  <RouterB> system-view
  [RouterB] interface Atm6/0
  [RouterB-Atm6/0] pvc 10/33
  [RouterB-atm-pvc-Atm1/10/33] map ip inarp
  [RouterB-atm-pvc-Atm1/10/33] quit
  [RouterB-Atm6/0] ip address 100.1.1.2 255.255.255.0
  ```
- <RouterA> debugging atm event
  ```bash
  *0.413674544 RouterA ATM/8/debug8: Atm5/0 pvc 10/33 event: OAM change state to down
  // The state of PVC 10/33 on ATM 5/0 interface turned to down.
  ```
```
debugging atm oam

Use **debugging atm oam** to enable debugging for ATM OAM, which displays information about the OAM packets sent/received on an ATM interface.

Use **undo debugging atm oam** to disable debugging for ATM OAM.

**Syntax**

```
debugging atm oam [ interface interface-type interface-number ]
undo debugging atm oam [ interface interface-type interface-number ]
```

**Default**

Debugging for ATM OAM is disabled.

**Views**

User view

**Default command level**

1: Monitor level

**Parameters**

```
interface-type interface-number: Specifies an interface by its type and number.
```

**Usage guidelines**

If no interface is specified, the operation applies to all ATM interfaces.

Table 50 describes output fields and messages for the **debugging atm oam** command.

**Table 50 Output from the debugging atm oam command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>interface name</td>
<td>Interface name.</td>
</tr>
<tr>
<td>packet input / output</td>
<td>Information about input/output packets.</td>
</tr>
<tr>
<td>packet length</td>
<td>Packet length.</td>
</tr>
<tr>
<td>partial data</td>
<td>Partial packet data.</td>
</tr>
</tbody>
</table>

**Examples**

```
# Enable debugging for ATM OAM.
<Sysname> debugging atm oam
*0.414264551 Sysname DRVDBG/8/debugging:
(Atm5/0)PHY/PKT:
    Packet Output, Packet Len =48,Partial data as follows:
    18 01 00 00 00 AA FF FF FF FF FF FF FF FF FF
    FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
    6A 6A 6A 6A 6A 6A 6A 6A 6A 6A 6A 6A 6A 6A 00 00

// ATM sent an OAM packet through ATM 5/0 interface.
```

debugging atm packet

Use **debugging atm packet** to enable debugging for ATM packets.
Use **undo debugging atm packet** to disable debugging for ATM packets.

**Syntax**

```plaintext
debugging atm packet [ interface interface-type interface-number [ pvc { pvc-name [ vpi/vci ] | vpi/vci } ] ]

undo debugging atm packet [ interface interface-type interface-number [ pvc { pvc-name [ vpi/vci ] | vpi/vci } ] ]
```

**Default**

Debugging for ATM packets is disabled.

**Views**

User view

**Default command level**

1: Monitor level

**Parameters**

- **interface-type interface-number**: Specifies an interface by its type and number.
- **pvc-name**: PVC name, a string of 1 to 16 characters. A PVC name is unique on an ATM interface and is not case-sensitive. It cannot be a valid VPI/VCI pair (such as 1/20).
- **vpi/vci**: ATM virtual path identifier (VPI) and ATM virtual channel identifier (VCI). The **vpi** argument ranges from 0 to 255. The range of the **vci** argument varies with interface types, as listed in Table 47. Normally, VCI 0 through VCI 31 are reserved for special purposes.

**Usage guidelines**

If no interface is specified, the operation applies to all ATM interfaces.

If you do not specify the PVC name or VPI/VCI pair, the operation applies to all PVCs.

After debugging for ATM packets is enabled, large amount of output information is generated when the device sends or receives a packet. Debugging for ATM packets might also disable users from managing network devices through terminals and might greatly decrease forwarding performance.

**Table 51** describes output fields and messages for the **debugging atm packet** command.

**Table 51 Output from the debugging atm packet command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>interface name</td>
<td>Interface name.</td>
</tr>
<tr>
<td>pvc VPI/VCI</td>
<td>VPI/VCI pair.</td>
</tr>
<tr>
<td>in / out</td>
<td>Packet direction.</td>
</tr>
<tr>
<td>packet error</td>
<td>Packet and error cause. This field appears when a packet error occurs.</td>
</tr>
<tr>
<td>packet type</td>
<td>Packet type.</td>
</tr>
<tr>
<td>encapsulation type</td>
<td>Encapsulation type.</td>
</tr>
<tr>
<td>packet length</td>
<td>Packet length.</td>
</tr>
<tr>
<td>packet data</td>
<td>Partial packet data.</td>
</tr>
</tbody>
</table>
Examples

# Enable debugging for ATM packets on all ATM interfaces.
<Sysname> debugging atm packet

......
*0.263288 Sysname ATM/7/debug8: Atm5/0 pvc 10/33 out ip pkt, snap, 92
*0.263288 Sysname ATM/7/debug8:   AA AA 03 00 00 00 08 00 45 00 00 54 00 05 00 00
*0.263288 Sysname ATM/7/debug8:   FF 01 F1 9E 64 01 01 01 64 01 01 02 08 00 53 9D

// ATM sent an IP packet through PVC 10/33 on ATM 5/0 interface. The packet is 92 bytes in length, with the encapsulation type SNAP.
Attack detection and protection debugging commands

The output description tables in this document only contain fields and messages that require an explanation.

debugging attack-defense

Use `debugging attack-defense` to enable attack detection and protection debugging.

Use `undo debugging attack-defense` to disable attack detection and protection debugging.

**Syntax**

```
debugging attack-defense { all | error | event }
undo debugging attack-defense { all | error | event }
```

**Default**

No attack detection and protection debugging is enabled.

**Views**

User view

**Default command level**

1: Monitor level

**Parameters**

- **all**: Specifies all types of attack detection and protection debugging.
- **error**: Specifies error debugging of attack detection and protection.
- **event**: Specifies event debugging of attack detection and protection.

**Usage guidelines**

Table 52 describes output fields and messages for the `debugging attack-defense event` command.
### Table 52 Output from the debugging attack-defense event command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attack begin.</td>
<td>Information about the detected attack:</td>
</tr>
<tr>
<td>Attack type: type</td>
<td>• <strong>Attack type</strong>—The attack type, Scan, UDP Flood, ICMP Flood, or SYN Flood.</td>
</tr>
<tr>
<td>Interface: interface-type interface-number</td>
<td>• <strong>Interface</strong>—Interface where attack protection is configured.</td>
</tr>
<tr>
<td>Action: action</td>
<td>• <strong>Action</strong>—Action against the attack. Available action parameters are:</td>
</tr>
<tr>
<td>IP address: ip-address</td>
<td>- none—Takes no actions but outputs the attack alarm logs.</td>
</tr>
<tr>
<td>Attack end.</td>
<td>- drop packet—Also drops attack packets.</td>
</tr>
<tr>
<td></td>
<td>- send RST to destination host—Also sends an RST message to the destination host.</td>
</tr>
<tr>
<td></td>
<td>- drop packet and add source host to blacklist—Also drops attack packets and adds the attacker’s IP address to the blacklist.</td>
</tr>
<tr>
<td></td>
<td>• <strong>IP address</strong>—Attacker’s IP address for scanning attacks or the victim IP address for flood attacks.</td>
</tr>
</tbody>
</table>

### Table 53 describes output fields and messages for the debugging attack-defense event command.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Failed to add ip-address to blacklist for already existing.</td>
<td>The attack protection module failed to add ip-address to the blacklist for the entry already exists.</td>
</tr>
<tr>
<td>Failed to add ip-address to blacklist.</td>
<td>The attack protection module failed to add ip-address to the blacklist due to insufficient hardware or software resources.</td>
</tr>
<tr>
<td>Failed to send attack-type log to IC.</td>
<td>The attack types include Land, Smurf, Fraggle, Winnuke, ICMP Redirect, ICMP Unreachable, Tracert, TCP Flag, Large ICMP, Source Route, Route Record, Scan, UDP Flood, ICMP Flood, and SYN Flood.</td>
</tr>
</tbody>
</table>
Examples

# Enable attack protection error debugging. Output similar to the following example is generated when the device detects a scanning attack sourced from 1.1.1.1 under the following conditions:

1. A scanning attack protection policy is configured on the device.
2. A blacklist entry is manually added for the IP address 1.1.1.1.

```plaintext
<Sysname> debugging attack-defense error
*Apr 10 21:07:53:78 2008 Sysname ATTACK/7/Attack-defend Error:
Failed to add 1.1.1.1 to blacklist for already existing.
```

// The device failed to add IP address 1.1.1.1 to the blacklist because the entry was already added manually.

# Enable attack protection event debugging. Output similar to the following example is generated when the device detects a Land attack under the following conditions:

- A scanning attack protection policy is configured to prevent Land attacks on the device.
- The blacklist function is enabled.

```plaintext
<Sysname> debugging attack-defense event
*Apr 10 21:07:53:125 2008 Sysname ATTACK/7/Attack-defend Event:
Attack begin.
Attack type: Scan
Interface: GigabitEthernet1/1
Action: drop packet and add source host to blacklist
IP address: 1.1.1.1
```

// The device detected a scanning attack on GigabitEthernet 1/1. The predefined protection action is to add the attacker's IP address to the blacklist, and to drop packets from the attacker.

```plaintext
*Apr 10 21:07:53:172 2008 Sysname ATTACK/7/Attack-defend Error:
Success to add 1.1.1.1 to blacklist, aging time is 30(s).
```

// The device added 1.1.1.1 to the blacklist. The entry's aging time is 30 seconds.

```plaintext
*Apr 10 21:07:53:141 2008 Sysname ATTACK/7/Attack-defend Event:
Attack end.
Attack type: Scan
Interface: GigabitEthernet1/1
Action: none
IP address: 1.1.1.1
```

// The blacklist function took effect. GigabitEthernet 1/1 received no packets from 1.1.1.1. The scanning attack was considered over.

```plaintext
*Apr 10 21:07:53:157 2008 Sysname ATTACK/7/Attack-defend Event:
Single packet attack.
Attack type: Land
Interface: GigabitEthernet1/1
Action: none
Source IP address: 1.1.1.1
Destination IP address: 1.1.3.2
```

// The device detected a Land attack on GigabitEthernet 1/1. The device only sent attack alarm logs to the information center.
debugging blacklist

Use `debugging blacklist` to enable blacklist debugging.
Use `undo debugging blacklist` to disable blacklist debugging.

Syntax

```
debugging blacklist { all | error | event }
undo debugging blacklist { all | error | event }
```

Default

No blacklist debugging is enabled.

Views

User view

Default command level

1: Monitor level

Parameters

- `all`: Specifies all types of blacklist debugging.
- `error`: Specifies blacklist error debugging.
- `event`: Specifies blacklist event debugging.

Usage guidelines

Table 54 describes output fields and messages for the `debugging blacklist error` command.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Failed to enqueue when deleting blacklist item. Item IP: <code>ip-address</code></td>
<td>Blacklist failed to add a blacklist entry to the deleted entry queue when deleting the entry.</td>
</tr>
<tr>
<td>Failed to create new blacklist.</td>
<td>Blacklist failed to create a blacklist entry.</td>
</tr>
<tr>
<td>Blacklist is full.</td>
<td>Number of blacklist entries has reached the upper limit.</td>
</tr>
</tbody>
</table>

Table 55 describes output fields and messages for the `debugging blacklist event` command.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Drop packet for matching the blacklist. IP address: <code>ip-address</code></td>
<td>Blacklist dropped a packet matching the blacklist. <code>ip-address</code> indicates the source IP address of the packet.</td>
</tr>
</tbody>
</table>

Examples

```
# Enable blacklist error debugging. Output similar to the following example is generated when the device detects a scanning attack under the following conditions:

• A scanning attack protection policy is configured on the device.
• The blacklist function is enabled for scanning attack protection.
• The number of blacklist entries has reached the upper limit.
```
<Sysname>debugging blacklist error
*Apr 10 20:47:54:734 2008 Sysname BLS/7/BLS Error:
Failed to create new blacklist.

// The device failed to add a new entry to the blacklist because the blacklist is full.

# Enable blacklist event debugging. Output similar to the following example is generated when the device detects a scanning attack under the following conditions:

• A scanning attack protection policy is configured on the device.
• The blacklist function is enabled for scanning attack protection.

<Sysname> debugging blacklist event
*Apr 10 20:47:54:750 2008 Sysname BLS/7/BLS Event:
Drop packet for matching the blacklist.
IP address: 1.1.1.1

// The device dropped the packets whose source IP address is in the blacklist.

debugging flow-statistics

Use `debugging flow-statistics` to enable traffic statistics debugging.

Use `undo debugging flow-statistics` to disable traffic statistics debugging.

Syntax

```
debugging flow-statistics { all | error | event }
undo debugging flow-statistics { all | error | event }
```

Default

No traffic statistics debugging is enabled.

Views

User view

Default command level

1: Monitor level

Parameters

`all`: Specifies all types of traffic statistics debugging.
`error`: Specifies error debugging of traffic statistics.
`event`: Specifies event debugging of traffic statistics.

Usage guidelines

Table 56 describes output fields and messages for the `debugging flow-statistics error` command.

**Table 56 Output from the debugging flow-statistics error command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP statistic list is full.</td>
<td>Number of IP traffic statistics node entries has reached the upper limit.</td>
</tr>
<tr>
<td>Failed to get time value.</td>
<td>The attack protection module failed to get the system time.</td>
</tr>
</tbody>
</table>
Table 57 describes output fields and messages for the **debugging flow-statistics event** command.

### Table 57 Output from the debugging flow-statistics event command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Operation: create source IP statistic node | The attack protection module created an IP traffic statistics node.  
  - **IP address**—Source IP address of the statistics object.  
  - **VPN Instance ID**—VPN instance to which the traffic belongs. |
| IP address: source-ip         |                                                                                                                                          |
| VPN Instance ID: vpn-instance-id |                                                                                                                                          |
| Operation: create destination IP statistic node | The attack protection module created an IP traffic statistics node.  
  - **IP address**—Destination IP address of the statistics object.  
  - **VPN Instance ID**—VPN instance to which the traffic belongs. |
| IP address: destination-ip    |                                                                                                                                          |
| VPN Instance ID: vpn-instance-id |                                                                                                                                          |

### Examples

**# Enable traffic statistics event debugging.** When an interface with traffic accounting by source IP address configured receives packets, output similar to the following example is generated:

```plaintext
<Sysname> debugging flow-statistics event
*Apr 10 21:07:53:203 2008 Sysname ATTACK/7/Flow-statistic Event:
  Operation: create source IP statistic node
  IP address: 1.1.1.1
  VPN Instance ID: 25
// The device created an IP traffic statistics node. The source IP address of the statistics object is 1.1.1.1, and the traffic belongs to VPN instance 25.
```

**# Enable traffic statistics error debugging.** The output in this example was created when the following conditions exist:

- Traffic accounting by source IP address is configured on an interface.
- The number of packets that the interface receives from a source IP address exceeds the threshold.

```plaintext
*Apr 10 21:07:53:203 2008 Sysname ATTACK/7/Flow-statistic Error:
  IP statistic list is full.
// Number of entries of the IP traffic statistics node reached the upper limit.
```

### debugging tcp-proxy

Use **debugging tcp-proxy** to enable TCP proxy debugging.

Use **undo debugging tcp-proxy** to disable TCP proxy debugging.

#### Syntax

```
debugging tcp-proxy { all | dhbk | error | event | packet | timer }
debugging tcp-proxy { all | dhbk | error | event | packet | timer }
```

#### Default

TCP proxy debugging is disabled.
Views

User view

Default command level

1: Monitor level

Parameters

**all**: Specifies all types of TCP proxy debugging.

**dhhbk**: Specifies TCP proxy stateful failover debugging.

**error**: Specifies TCP proxy error debugging.

**event**: Specifies TCP proxy event debugging.

**packet**: Specifies TCP proxy packet debugging.

**timer**: Specifies TCP proxy timer debugging.

Usage guidelines

Table 58 describes output fields and messages for the **debugging tcp-proxy dhhbk** command.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>

Examples

# Enable TCP proxy event debugging. Output similar to the following example is generated when the receiving packet rate on an interface exceeds the threshold under the following conditions:

- An attack protection policy against SYN flood attacks is applied to the interface.
- The protection action is to trigger TCP proxy to add the victim IP address to the protected IP list.
- TCP proxy is enabled on the interface.

```
<Sysname> debugging tcp-proxy event
*Jun 28 09:40:56:763 2010 Sysname TCPPROXY/7/ EVENT: (3.3.3.3:0); Added a dynamic protected IP entry.
```
Basic IP routing debugging commands

The output description tables in this document only contain fields and messages that require an explanation.

Enabling any debugging command in this chapter might affect system performance, especially when the system is busy. Therefore, disable the debugging once the debugging operation is complete.

decommissioning rm all

Use `debugging rm all` to enable all route management (RM) debugging. Use `undo debugging rm all` to disable all RM debugging.

**Syntax**
```
debugging rm all
undo debugging rm all
```

**Default**
All RM debugging is disabled.

**Views**
User view

**Default command level**
1: Monitor level

**Examples**
```
# Enable all RM debugging.
<Sysname> debugging rm all
```

debugging rm backup

Use `debugging rm backup` to enable RM data backup debugging. Use `undo debugging rm backup` to disable RM data backup debugging.

**Syntax**
```
debugging rm backup
undo debugging rm backup
```

**Default**
Data backup debugging is disabled.

**Views**
User view

**Default command level**
1: Monitor level
Usage guidelines

Table 59 describes output fields and messages for the `debugging rm backup` command.

### Table 59 Output from the debugging rm backup command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Module XXX</td>
<td>Module name.</td>
</tr>
<tr>
<td>backup data(XXX)</td>
<td>Backup data size in bytes.</td>
</tr>
<tr>
<td>routine backup</td>
<td>Scheduled backup.</td>
</tr>
<tr>
<td>realtime backup</td>
<td>Realtime backup.</td>
</tr>
<tr>
<td>batch backup</td>
<td>Batch backup.</td>
</tr>
</tbody>
</table>

**Examples**

`#` Enable RM data backup debugging. When you configure an IPv4 static route, output similar to the following example is generated:

```plaintext
<Sysname> debugging rm backup
*Jan 9 16:52:03:500 2010 Sysname RM/3/RMDEBUG:
  RM Backup: Receive HA routine backup notification
<Sysname> system-view
[Sysname] ip route-static 1.1.1.1 32 null 0
*Jan 9 16:52:05:875 2010 Sysname RM/3/RMDEBUG:
  RM Backup: Module USR sends backup data(284) to SMB.
*Jan 9 16:52:05:875 2010 Sysname RM/3/RMDEBUG:
  RM Backup: RM send realtime backup data success.

// RM sent the static route backup data.
```

**debugging rm bfd**

Use `debugging rm bfd` to enable RM BFD debugging.

Use `undo debugging rm bfd` to disable RM BFD debugging.

**Syntax**

```plaintext
debugging rm bfd
undo debugging rm bfd
```

**Default**

RM BFD debugging is disabled.

**Views**

User view

**Default command level**

1: Monitor level

**Usage guidelines**

Table 60 describes output fields and messages for the `debugging rm bfd` command.
Table 60 Output from the debugging rm bfd command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>InstanceID</td>
<td>Instance ID.</td>
</tr>
<tr>
<td>MsgType</td>
<td>Message type.</td>
</tr>
<tr>
<td>ConnectType</td>
<td>Session type.</td>
</tr>
<tr>
<td>IntfIdx</td>
<td>Interface index.</td>
</tr>
<tr>
<td>AppProto</td>
<td>Protocol type.</td>
</tr>
<tr>
<td>SrcIPAddr</td>
<td>Source IP address of the BFD session.</td>
</tr>
<tr>
<td>DstIPAddr</td>
<td>Destination IP address of the BFD session.</td>
</tr>
</tbody>
</table>

Examples

# Enable RM BFD debugging. When you configure an IPv4 static route with BFD enabled, output similar to the following example is generated:

<Sysname> debugging rm bfd
<Sysname> system-view
[Sysname] ip route-static 1.1.1.2 32 Ethernet 1/1 10.11.1.4 bfd control-packet
*Jan 9 15:53:04:672 2010 Sysname RM/3/RMDEBUG:
  RM->BFD ::InstanceID = 0x0, MsgType = 0x0, ConnectType=0x1 ,IntfIdx = 0xf0008, AppProto=0x80, SrcIPAddr =10.11.1.3, DstIPAddr =10.11.1.4
  // RM notified BFD to establish a session.

doublegging rm ipv4

Use debugging rm ipv4 to enable RM IPv4 route debugging.
Use undo debugging rm ipv4 to disable RM IPv4 route debugging.

Syntax

doublegging rm ipv4 { im | msr | rcom [ ip-prefix prefix-name ] | rr | rtmsg | urt [ ip-prefix prefix-name ] | usr }
undo doublegging rm ipv4 { im | msr | rcom | rr | rtmsg | urt | usr }

Default

RM IPv4 route debugging is disabled.

Views

User view

Default command level

1: Monitor level

Parameters

im: Specifies IPv4 routing interface debugging.
msr: Specifies IPv4 multicast static route debugging.
rcom: Specifies IPv4 route sending debugging.
ip-prefix prefix-name: References an IP prefix list.
**rr**: Specifies IPv4 recursive route debugging.

**rtmsg**: Specifies IPv4 routing table message debugging.

**urt**: Specifies IPv4 routing table debugging.

**usr**: Specifies IPv4 unicast static route debugging.

**Usage guidelines**

Table 61 describes output fields and messages for the `debugging rm ipv4 im` command.

**Table 61 Output from the debugging rm ipv4 im command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IfnetIdx</td>
<td>Interface index.</td>
</tr>
<tr>
<td>IfState</td>
<td>Interface state.</td>
</tr>
<tr>
<td>Slot</td>
<td>Slot number.</td>
</tr>
<tr>
<td>IfName</td>
<td>Interface name.</td>
</tr>
<tr>
<td>IfType</td>
<td>Interface type.</td>
</tr>
<tr>
<td>MsgType</td>
<td>Message type.</td>
</tr>
<tr>
<td>InstID</td>
<td>Interface instance ID.</td>
</tr>
<tr>
<td>Bandwidth</td>
<td>Interface bandwidth.</td>
</tr>
<tr>
<td>Baudrate</td>
<td>Baud rate.</td>
</tr>
<tr>
<td>MTU</td>
<td>Maximum transmission unit (MTU).</td>
</tr>
</tbody>
</table>

Table 62 describes output fields and messages for the `debugging rm ipv4 rcom` command.

**Table 62 Output from the debugging rm ipv4 rcom command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NH</td>
<td>Next hop.</td>
</tr>
<tr>
<td>If</td>
<td>Interface index.</td>
</tr>
<tr>
<td>Addr/Mask</td>
<td>Interface address/mask.</td>
</tr>
</tbody>
</table>

Table 63 describes output fields and messages for the `debugging rm ipv4 rr` command.

**Table 63 Output from the debugging rm ipv4 rr command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add/Del Relay Radix ExtNode</td>
<td>Add/delete the recursive next hop node.</td>
</tr>
<tr>
<td>Addr/Mask</td>
<td>Next hop address/mask.</td>
</tr>
<tr>
<td>DstIns/NHIns</td>
<td>Destination[next hop instance.</td>
</tr>
</tbody>
</table>

Table 64 describes output fields and messages for the `debugging rm ipv4 rtmsg` command.

**Table 64 Output from the debugging rm ipv4 rtmsg command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>received msg(XX,X)</td>
<td>Type/Protocol ID of the received message.</td>
</tr>
</tbody>
</table>
### Table 65 describes output fields and messages that for the **debugging rm ipv4 urt** command.

### Table 65 Output from the debugging rm ipv4 urt command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>for: dest/mask</td>
<td>Destination address/mask.</td>
</tr>
<tr>
<td>ExitIf</td>
<td>Index of the outgoing interface.</td>
</tr>
</tbody>
</table>

Examples

```
# Enable RM IPv4 route table debugging. When you add a static route whose destination address and
next hop are 21.1.1.0/24 and 43.43.43.40, respectively, to the routing table, output similar to the
following example is generated:
<Sysname> debugging rm ipv4 urt
*Jun 28 14:59:26:681 2007 29.43 RM/3/RMDEBUG:
 Add route TableID:1 Dest:21.1.1.0 Mask:255.255.255.0 Nexthop:43.43.43.40 Neighbor
 r:0.0.0.0 Protocol:Static
*Jun 28 14:59:26:682 2007 AR29.43 RM/3/RMDEBUG:
 RM RR: URT add route Dest : 21.1.1.0, Mask : 255.255.255.0, Nexthop : 43.43.43.40,
 Neighbor : 0.0.0.0 to wait queue success.
// RM added a static route whose destination address and next hop are 21.1.1.0/24 and 43.43.43.40,
respectively.

# Delete the static route.
*Jun 28 15:05:03:09 2007 29.43 RM/3/RMDEBUG:
 Delete route TableID:1 Dest:21.1.1.0 Mask:255.255.255.0 Nexthop:43.43.43.40 Neighbor
 0.0.0.0 Protocol:Static
*Jun 28 15:05:03:10 2007 AR29.43 RM/3/RMDEBUG:
 RM_URT_CalcActRoutes_H 15010100/ffffff00 bIsProto:0, useProtoID:0
// RM deleted the static route.
```

**debugging rm ipv6**

Use **debugging rm ipv6** to enable RM IPv6 route debugging.

Use **undo debugging rm ipv6 urt** to disable RM IPv6 route debugging.
Syntax

```
debugging rm ipv6 { im | rcom [ip-prefix prefix-name] | rr | urt [ip-prefix prefix-name] | usr }  
undo debugging rm ipv6 { im | rcom | rr | urt | usr }
```

Default

RM IPv6 route debugging is disabled.

Views

User view

Default command level

1: Monitor level

Parameters

- **im**: Specifies IPv6 routing interface debugging.
- **rcom**: Specifies IPv6 route sending debugging.
- **ip-prefix prefix-name**: References an IPv6 prefix list.
- **rr**: Specifies IPv6 recursive route debugging.
- **urt**: Specifies IPv6 routing table debugging.
- **usr**: Specifies IPv6 unicast static route debugging.

Usage guidelines

Table 66 describes output fields and messages for the `debugging rm ipv6 im` command.

**Table 66 Output from the debugging rm ipv6 im command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IfnetIdx</td>
<td>Interface index.</td>
</tr>
<tr>
<td>IfState</td>
<td>Interface state.</td>
</tr>
<tr>
<td>Slot</td>
<td>Slot number.</td>
</tr>
<tr>
<td>IfName</td>
<td>Interface name.</td>
</tr>
<tr>
<td>IfType</td>
<td>Interface type.</td>
</tr>
<tr>
<td>MsgType</td>
<td>Message type.</td>
</tr>
<tr>
<td>Bandwidth</td>
<td>Interface bandwidth.</td>
</tr>
<tr>
<td>Baudrate</td>
<td>Baud rate.</td>
</tr>
<tr>
<td>LogAddr</td>
<td>Interface address.</td>
</tr>
<tr>
<td>PrefixLen</td>
<td>Prefix length.</td>
</tr>
</tbody>
</table>

Table 67 describes output fields and messages for the `debugging rm ipv6 rcom` command.

**Table 67 Output from the debugging rm ipv6 rcom command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Route with Dest</td>
<td>Destination prefix of the route.</td>
</tr>
<tr>
<td>added to FIB6</td>
<td>Added to the FIB table.</td>
</tr>
</tbody>
</table>
Table 68 describes output fields and messages for the debugging rm ipv6 urt command.

Table 68: Output from the debugging rm ipv6 urt command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RM</td>
<td>Route management.</td>
</tr>
<tr>
<td>addition msg</td>
<td>Route addition message.</td>
</tr>
<tr>
<td>deletion msg</td>
<td>Route deletion message.</td>
</tr>
<tr>
<td>modification msg</td>
<td>Route modification message.</td>
</tr>
<tr>
<td>selfflag msg</td>
<td>Route flag setting message.</td>
</tr>
<tr>
<td>resetflag msg</td>
<td>Route flag resetting message.</td>
</tr>
<tr>
<td>setrelyflag msg</td>
<td>Route recursion flag setting message.</td>
</tr>
<tr>
<td>resetrelyflag msg</td>
<td>Route recursion flag resetting message.</td>
</tr>
<tr>
<td>Destination</td>
<td>Destination IPv6 address.</td>
</tr>
<tr>
<td>Neighbour</td>
<td>Neighbor address.</td>
</tr>
</tbody>
</table>

Examples

# Enable IPv6 route debugging.
<Sysname> debugging rm ipv6 urt

RM receive ipv6 route addition msg, the detail info is
   Destination : FE80::                   PrefixLength : 10
   NextHop     : ::                       Preference   : 0
   RelayNextHop: : :
   Neighbour   : ::                       Protocol     : Direct
   Interface   : NULL0                    Flags        : 10010

RM receive ipv6 route addition msg, the detail info is
   Destination : 123::123                 PrefixLength : 128
   NextHop     : ::1                      Preference   : 0
   RelayNextHop: : :
   Neighbour   : ::                       Protocol     : Direct
   Interface   : InLoopBack0              Flags        : 10050

RM receive ipv6 route addition msg, the detail info is
   Destination : 123::                    PrefixLength : 100
   NextHop     : 123::123                 Preference   : 0
   RelayNextHop: : :
   Neighbour   : ::                       Protocol     : Direct
   Interface   : Ethernet1/0              Flags        : 10000
//RM added IPv6 routes.

[Sysname-Ethernet1/0] undo ipv6 address

RM receive ipv6 route deletion msg, the detail info is
  Destination: FE80::  PrefixLength: 10
  NextHop: ::  Preference: 0
  RelayNextHop: ::
  Neighbour: ::  Protocol: Direct
  Interface: NULL0  Flags: 10010

RM receive ipv6 route deletion msg, the detail info is
  Destination: 123::123  PrefixLength: 128
  NextHop: ::1  Preference: 0
  RelayNextHop: ::
  Neighbour: ::  Protocol: Direct
  Interface: InLoopBack0  Flags: 10050

RM receive ipv6 route deletion msg, the detail info is
  Destination: 123::  PrefixLength: 100
  NextHop: 123::123  Preference: 0
  RelayNextHop: ::
  Neighbour: ::  Protocol: Direct
  Interface: Ethernet1/0  Flags: 10000

// RM deleted IPv6 routes.

debugging rm job

Use debugging rm job to enable RM job debugging.
Use undo debugging rm job to disable RM job debugging.

Syntax

debugging rm job [ jobfilter job-name ]
undo debugging rm job

Default

RM job debugging is disabled.

Views

User view

Default command level

1: Monitor level

Parameters

jobfilter job-name: References a job by its name.

Usage guidelines

Table 69 describes output fields and messages for the debugging rm job command.
Table 69 Output from the debugging rm job command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Job XXX</td>
<td>• <strong>SelActRt</strong>—Optimal route selection job.</td>
</tr>
<tr>
<td></td>
<td>• <strong>NotifyRoute</strong>—Route notification job.</td>
</tr>
</tbody>
</table>

Examples

# Enable RM job debugging. When you configure an IPv4 static route, output similar to the following example is generated:

```bash
<Sysname> debugging rm job
<Sysname> system-view
.Sysname] ip route-static 1.1.1.1 32 null 0
*Jan 9 16:15:17:969 2010 Sysname RM/3/RMDEBUG: Job SelActRt created (Pri:6,TID:3)
*Jan 9 16:15:17:969 2010 Sysname RM/3/RMDEBUG: Begin processing JOB SelActRt(6)
*Jan 9 16:15:17:969 2010 Sysname RM/3/RMDEBUG: Job NotifyRoute created (Pri:6,TID:5)
*Jan 9 16:15:17:969 2010 Sysname RM/3/RMDEBUG: Begin processing JOB NotifyRoute(6)
// RM created and executed the optimal route selection job and route notification job.
```

debugging rm policy

Use `debugging rm policy` to enable routing policy debugging.

Use `undo debugging rm policy` to disable routing policy debugging.

Syntax

```bash
devugging rm policy [ ip-prefix prefix-name ]
undo devugging rm policy
```

Default

Routing policy debugging is disabled.

Views

User view

Default command level

1: Monitor level

Parameters

- `ip-prefix prefix-name`: Matches an IP prefix list.

Usage guidelines

Table 70 describes output fields and messages for the `debugging rm policy` command.

Table 70 Output from the debugging rm policy command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prefix filter</td>
<td>IPv4 prefix filter.</td>
</tr>
<tr>
<td>Prefix filter6</td>
<td>IPv6 prefix filter.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>route-map 'XXX'</td>
<td>Routing policy XXX.</td>
</tr>
</tbody>
</table>

Examples

# Display routing policy debugging information matching IP prefix list test.
<Sysname> debugging rm policy ip-prefix test
[Sysname] ip as-path 10 permit test
*Aug 26 15:01:01:64 2006 Sysname RM/3/RMDEBUG:
   Aspath filter created for List ID : 10

// RM created AS path list 10.
[Sysname] undo ip as-path 10
*Aug 26 15:01:53:706 2006 Sysname RM/3/RMDEBUG:
Aspath filter with list ID :10 is deleted

// RM deleted AS path list 10.
[Sysname] ip community-list 101 permit test
*Aug 26 15:11:20:554 2006 Sysname RM/3/RMDEBUG:
Community-list filter created with list ID 101

// RM created IP community list 101.
[Sysname] undo ip community-list 101
*Aug 26 15:12:22:750 2006 Sysname RM/3/RMDEBUG:
Community-list filter deleted with list ID 101

// RM deleted IP community list 101.
[Sysname] ip extcommunity-list 100 permit rt 1:1
*Aug 26 15:15:25:554 2006 Sysname RM/3/RMDEBUG:
Extcommunity-list filter added successfully

// RM created IP extended community list 100.
[Sysname] undo ip extcommunity-list 100
*Aug 26 15:17:00:782 2006 Sysname RM/3/RMDEBUG:
Extcommunity-list filter deleted with list ID 100

// RM deleted IP extended community list 100.
[Sysname] ip ip-prefix test permit 1.1.1.1 24
*Aug 26 15:18:38:944 2006 Sysname RM/3/RMDEBUG:
Prefix filter 'test' created

// RM created IP prefix list test.
[Sysname] undo ip ip-prefix test index 10
Prefix filter 'test' deleted with seq-no 10

// RM deleted IP prefix list test.
[Sysname] ip ipv6-prefix test permit 1::1 128
*Aug 26 16:24:15:848 2006 Sysname RM/3/RMDEBUG:
Prefix filter6 'test' created

// RM created IPv6 prefix list test.
[Sysname] undo ip ipv6-prefix test index 10
*Aug 26 16:25:32:782 2006 Sysname RM/3/RMDEBUG:
Prefix filter6 'test' deleted with seq-no 10

// RM deleted IPv6 prefix list test.

[Sysname] route-policy test permit node 10

*Aug 26 15:34:27:460 2006 Sysname RM/3/RMDEBUG:
route-map 'test' created successfully

// RM created Routing policy test with node 11.

[Sysname] route-policy test permit node 11

*Aug 26 15:35:29:396 2006 Sysname RM/3/RMDEBUG:
route-map 'test' new sequence number 11 added!

// RM added Node 11 to routing policy test.

[Sysname] undo route-policy test

*Aug 26 15:38:21:368 2006 Sysname RM/3/RMDEBUG:
Route-map filter 'test' deleted successfully

// RM deleted routing policy test.

[Sysname-rip-1] filter-policy ip-prefix test export

*Aug 26 15:45:00:266 2006 Sysname RM/3/RMDEBUG:
Distribute policy creation successful

// RM created a route distribution policy using IP prefix list test.

[Sysname-rip-1] undo filter-policy export

*Aug 26 15:45:00:266 2006 Sysname RM/3/RMDEBUG:
Distribute policy deletion successful

// RM deleted the route distribution policy.

[Sysname-ripng-1] filter-policy ipv6-prefix test export

*Aug 26 16:26:32:630 2006 Sysname RM/3/RMDEBUG:
Distribute6 policy creation successful

// RM created an IPv6 route distribution policy.

[Sysname-ripng-1] undo filter-policy export

*Aug 26 16:26:32:630 2006 Sysname RM/3/RMDEBUG:
Distribute6 policy deletion successful

// RM deleted the IPv6 route distribution policy.

[Sysname-rip-1] import-route static route-policy test

*Aug 26 16:11:59:37 2006 Sysname RM/3/RMDEBUG:
Create policy ReDistribute successfully

// RM applied routing policy test for IP static route redistribution.

[Sysname-rip-1] undo import-route static

*Aug 26 16:12:22:728 2006 Sysname RM/3/RMDEBUG:
ReDistribute policy deleted successfully

// RM deleted routing policy test for IP static route redistribution.

[Sysname-ripng-1] import-route static route-policy test

*Aug 26 16:33:38:162 2006 Sysname RM/3/RMDEBUG:
Create policy ReDistribute6 successful

// RM applied routing policy test for IPv6 static route redistribution.

[Sysname-ripng-1] undo import-route static

*Aug 26 16:34:04:956 2006 Sysname RM/3/RMDEBUG:
debugging rm system

Use **debugging rm system** to enable RM system debugging.

Use **undo debugging rm system** to disable RM system debugging.

**Syntax**

```
debugging rm system
undo debugging rm system
```

**Default**

RM system debugging is disabled.

**Views**

User view

**Default command level**

1: Monitor level

**Usage guidelines**

Table 71 describes output fields and messages for the **debugging rm system** command.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Instance XXX</td>
<td>Instance name: VPN instance ID.</td>
</tr>
</tbody>
</table>

**Examples**

```
# Enable RM system debugging. When you create a VPN instance, output similar to the following example is generated:
<Sysname> debugging rm system
<Sysname> system-view
[Sysname] ip vpn-instance vpn1
[Sysname-vpn-instance-vpn1]
// RM created VPN instance 1.

[Sysname] undo ip vpn-instance vpn1
// RM deleted VPN instance 1.
```

debugging rm task

Use **debugging rm task** to enable RM task debugging.

Use **undo debugging rm task** to disable RM task debugging.
Syntax

debugging rm task
undo debugging rm task

Default

RM task debugging is disabled.

Views

User view

Default command level

1: Monitor level

devbugging rm timer

Use debugging rm timer to enable timer debugging.
Use undo debugging rm timer to disable timer debugging.

Syntax

devbugging rm timer [ timerfilter timer-name ]
undo debugging rm timer

Default

Timer debugging is disabled.

Views

User view

Default command level

1: Monitor level

Parameters

timerfilter timer-name: Name of a timer to be filtered.

Usage guidelines

Table 72 describes output fields and messages for the debugging rm timer command.

Table 72 Output from the debugging rm timer command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XXX timer</td>
<td>Timer name:</td>
</tr>
<tr>
<td></td>
<td>• GotQ—Route recursion GotQ timer.</td>
</tr>
<tr>
<td></td>
<td>• WaitQ—Route recursion WaitQ timer.</td>
</tr>
<tr>
<td></td>
<td>• HI Timer—BGP Holdtime timer.</td>
</tr>
<tr>
<td></td>
<td>• KA Timer—BGP Keepalive timer.</td>
</tr>
<tr>
<td></td>
<td>• HELLO Timer—OSPF. Hello timer.</td>
</tr>
<tr>
<td>timer created</td>
<td>RM created the timer.</td>
</tr>
<tr>
<td>Timer XXX expires</td>
<td>The timer expired.</td>
</tr>
<tr>
<td>XXX timer deleted</td>
<td>RM deleted the timer.</td>
</tr>
</tbody>
</table>
Examples

# Enable timer debugging. The output in this example was created when IPv4 static routes are configured.
<Sysname> debugging rm timer

// RM created WaitQ and GotQ timers.

// WaitQ and GotQ timers expired.
Basic MPLS debugging commands

The output description tables in this document only contain fields and messages that require an explanation.

Some information in this chapter is device type specific. Devices in this chapter are categorized depending on their IRF capability and support for interface cards that use independent processors for forwarding traffic, as shown in Table 1.

**Table 73 Device types**

<table>
<thead>
<tr>
<th>Device type</th>
<th>Interface cards with on-card processors</th>
<th>IRF capability</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Distributed devices</td>
<td>Yes</td>
<td>No</td>
<td>HP 6600 routers (except for 6602)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Yes (in standalone mode)</td>
<td>HP 12500 switches</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>HP 10500 switches</td>
</tr>
<tr>
<td>Distributed IRF devices</td>
<td>Yes</td>
<td>Yes (in IRF mode)</td>
<td>HP 12500 switches</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>HP 10500 switches</td>
</tr>
<tr>
<td>Centralized devices</td>
<td>No</td>
<td>No</td>
<td>HP MSR routers</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>HP 6602 router</td>
</tr>
<tr>
<td>Centralized IRF devices</td>
<td>No</td>
<td>Yes</td>
<td>HP 5800 switches</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>HP 5500 switches</td>
</tr>
</tbody>
</table>

**debugging mpls fast-forwarding**

Use `debugging mpls fast-forwarding` to enable debugging for MPLS fast forwarding.

Use `undo debugging mpls fast-forwarding` to disable debugging for MPLS fast forwarding.

**Syntax**

- `debugging mpls fast-forwarding`
- `undo debugging mpls fast-forwarding`

**Default**

Debugging for MPLS fast forwarding is disabled.

**Views**

- User view

**Default command level**

1: Monitor level

**Usage guidelines**

This command is supported only on devices with single-core CPUs.

**Table 74** describes output fields and messages for the `debugging mpls fast-forwarding` command.
Table 74 Output from the debugging mpls fast forwarding command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MPLS fast switching : IP Packet for swap 3 label bottom of stack !</td>
<td>The packet is an IP packet after label 3 was popped off.</td>
</tr>
<tr>
<td>MPLS fast switching : L2VPN Packet !</td>
<td>The packet forwarded is an L2VPN packet.</td>
</tr>
</tbody>
</table>

Examples

# Configure MPLS L3VPN basic capability. Enable debugging for MPLS fast forwarding on a P device.

<Sysname> debugging mpls fast-forwarding

*Sep 25 14:08:40:966 2006 2921B MFW/7/MPLSFW PACKET:
MPLS Fast Switch: Rcv MPLS Pkt from Interface Ethernet1/1.1, Label(s)=1106, EXP=0, TTL=254

// MPLS received an MPLS packet with the in-label 1106, EXP 0, and TTL 254.

*Sep 25 14:08:40:966 2006 2921B MFW/7/GPLSFW PACKET:
MFW_FastSwitch: Using Label: 1106 to search cache after cutting other labels.

// MPLS was searching the fast forwarding cache for label 1106.

*Sep 25 14:08:40:966 2006 2921B MFW/7/GPLSFW PACKET:
MFW_FastSwitch succeed sending MPLS or IP Packet with:
output Interface: Serial0/2, using Qos Physical Transmit Interface!

// MPLS successfully sent the packet through interface Serial 0/2.

... *

*Sep 25 14:08:45:338 2006 2921B MFW/7/GPLSFW PACKET:
MFW_FastSwitch:POP Label 1095 and Send IP Packet received from Interface :Ethernet1/1.1
IP normal forwarding success !

// MPLS popped label 1095 and successfully sent the packet through interface Ethernet 1/1.1.

deploying mpls ldp

Use **debugging mpls ldp** to enable debugging for LDP.

Use **undo debugging mpls ldp** to disable debugging for LDP.

Syntax

```
debugging mpls ldp { { advertisement | all | error | main | notification | pdu | session | socket | timer } |
| interface interface-type interface-number } | fec [{ [ vpn-instance vpn-instance-name ] fec mask ] | hsb |
| remote-peer [ remote-peer-name ] }

undo debugging mpls ldp { { advertisement | all | error | main | notification | pdu | session | socket |
| timer } [ interface interface-type interface-number ] | fec [{ [ vpn-instance vpn-instance-name ] fec mask ] |
| hsb | remote-peer [ remote-peer-name ] }
```

Default

All types of debugging for LDP are disabled.

Views

User view

Default command level

1: Monitor level
Parameters

advertisement: Specifies debugging for advertisement events.
all: Specifies debugging for all events.
error: Specifies debugging for error events.
main: Specifies debugging for main interface events related to the external modules.
notification: Specifies debugging for notification events.
pdu: Specifies debugging for PDU events.
session: Specifies debugging for session events.
socket: Specifies debugging for socket events.
timer: Specifies debugging for timer events.
interface interface-type interface-number: Specifies debugging for the specified LDP-enabled interface.
fec: Specifies debugging for FEC events.

vpn-instance vpn-instance-name: Specifies debugging for the specified VPN instance. The
vpn-instance-name argument is the MPLS L3VPN name, a case-sensitive string of 1 to 31 characters.
fec mask: Specifies debugging for a particular FEC. The fec argument is an FEC destination address. The
mask argument is the mask length of the FEC destination address, in the range of 0 to 32.
hsb: Specifies debugging for hot standby (HSB) events.
remote-peer remote-peer-name: Specifies debugging for the specified remote peer. The
remote-peer-name argument is the remote peer name, a case-insensitive string of 1 to 32 characters.

Usage guidelines

This command is supported only on devices with single-core CPUs.

Table 75 describes output fields and messages for the debugging mpls ldp command.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>US Peer</td>
<td>Upstream peer.</td>
</tr>
<tr>
<td>USCB’s state</td>
<td>Status of the upstream control block.</td>
</tr>
<tr>
<td>DSCB’s state</td>
<td>Status of the downstream control block.</td>
</tr>
<tr>
<td>Delete USCB(X)</td>
<td>LDP deleted the upstream control block of the specified type:</td>
</tr>
<tr>
<td></td>
<td>• 1—Ingress, the ingress node control block.</td>
</tr>
<tr>
<td></td>
<td>• 2—Aggregated, the aggregation control block.</td>
</tr>
<tr>
<td></td>
<td>• 3—Common, the common control block.</td>
</tr>
<tr>
<td>OutIf</td>
<td>Outgoing interface.</td>
</tr>
<tr>
<td>Add Info Out_if</td>
<td>LDP added information about the outgoing interface.</td>
</tr>
</tbody>
</table>
Examples

The output in the following examples was created under the following conditions:

- Two routers are connected directly.
- MPLS and MPLS LDP are configured on both routers. An LDP session has been successfully established between the routers.

# Enable debugging for MPLS LDP advertisement on one router (PE 1, in this example). Restart LDP on the other router. Output similar to the following example is generated:

```
<PE1> debugging mpls ldp advertisement
<PE1>
*0.16830023 PE1 LDP/8/Advertisement:
DU Re-advertise for FEC: 140.2.2.2/0xffffffff.
*0.16830023 PE1 LDP/8/Advertisement:
0 - FEC: 140.2.2.2/0xffffffff, Successful Next Hop Query

// LDP was advertising FEC information regularly based on the DU Re-advertise timer.
```

Dec 22 14:55:50:388 2005 PE1 LDP/5/SessionDown: Session(140.2.2.2:0. public Instance)'s state change to Down

// The session went down.

```
*0.16838403 PE1 LDP/8/Advertisement: Serial0/2/0
  FEC: 140.2.2.2/0xffffffff, DU DS state machine process, DS Peer 140.2.2.2:0, DSCB's state: Established, Event: DownStream Lost

// The peer restarted LDP, and the upstream state machine processed upstream events.
```

```
*0.16838403 PE1 LDP/8/Advertisement:
  FEC: 140.2.2.2/0xffffffff, DU Ingress US state machine process, USCB's state: Established Event: Internal DownStream Withdraw

// The downstream state machine processed downstream events.
```

```
*0.16838403 PE1 LDP/8/Advertisement:
  LDP send LSP Message (Event: 3) to LSPM for LSP:
    0     140.2.2.2/32       -----/3     140.12.1.2      -------      Ser0/2/0
    The token is: 0. The MTU is: 1500.

// LDP sent an LSP message to the LSP management (LSPM) module.
```

```
*0.16838403 PE1 LDP/8/Advertisement:
  Delete USCB, FEC: 140.2.2.2/0xffffffff.
*0.16838403 PE1 LDP/8/Advertisement:
  Delete DSCB, FEC: 140.2.2.2/0xffffffff.
*0.16838419 PE1 LDP/8/Advertisement:
  Delete FEC Radix Tree Node, Destination: 140.2.2.2, Mask: 0xffffffff
*0.16838419 PE1 LDP/8/Advertisement:
  Downstream Lost, Cleanup Successful

// LDP deleted the downstream control block, upstream control block, and FEC node. The downstream was lost and the cleanup was successful.
```

```
*0.16838538 PE1 LDP/8/Advertisement:
  Start to process LSP Message(Type: 13) from LSPM - FEC: 140.2.2.2/0xffffffff, Nexthop: 140.12.1.2

// LDP was processing LSPM messages to re-establish the session.
```

```
*0.16838538 PE1 LDP/8/Advertisement:
```
Start to process Route add (FEC: 140.2.2.2/0xffffffff, OutIf: Serial0/2/0 Nexthop: 140.12.1.2)

// LDP was processing route addition event.
*0.16868001 PE1 LDP/8/Advertisement: Serial0/2/0
Receive Label Mapping Message from Peer(140.2.2.2:0) for FEC: 140.2.2.2/0xffffffff

// LDP received a label mapping message.
*0.16868001 PE1 LDP/8/Advertisement:
Create FEC Radix Tree Node, Destination: 140.2.2.2, Mask: 0xffffffff
*0.16868001 PE1 LDP/8/Advertisement:
Create cross node for FEC: 140.2.2.2, Mask: 0xffffffff.
*0.16868001 PE1 LDP/8/Advertisement:
Create DSCB, FEC: 140.2.2.2/0xffffffff.

// LDP created the FEC node, cross node, and downstream control block.
*0.16868001 PE1 LDP/8/Advertisement:
0 - FEC: 140.2.2.2/0xffffffff, Successful Next Hop Query
*0.16868001 PE1 LDP/8/Advertisement:
Update OutSeg Info, FEC: 140.2.2.2/0xffffffff, NHCount: 1.

// LDP updated the outgoing label information.
*0.16868001 PE1 LDP/8/Advertisement:
Add Info Out_If: 3145744, NHAddr: 140.12.1.2.
*0.16868001 PE1 LDP/8/Advertisement:
FEC: 140.2.2.2/0xffffffff, DU DS state machine process, DS Peer 140.2.2.2:0, DSCB's state: Idle, Event: LDP Mapping

// The downstream state machine was processing the LDP mapping event.
*0.16868001 PE1 LDP/8/Advertisement:
0 - FEC: 140.2.2.2/0xffffffff, Successful Next Hop Query
*0.16868017 PE1 LDP/8/Advertisement:
Create USCB, FEC: 140.2.2.2/0xffffffff.

// LDP created the upstream control block.
*0.16868032 PE1 LDP/8/Advertisement:
FEC: 140.2.2.2/0xffffffff, DU Ingress US state machine process, USCB's state: Idle, Event: Internal DownStream Mapping

// The upstream state machine was processing the downstream mapping event.
*0.16868032 PE1 LDP/8/Advertisement:
LDP send LSP Message (Event: 1) to LSPM for LSP:
0 140.2.2.2/32 140.12.1.2 Ser0/2/0
The token is: 557235. The MTU is: 1500.

// LDP sent an LSP creation success message to the LSPM.

# Enable debugging for MPLS LDP session on one router (PE 1, in this example). Restart LDP on the other router. Output similar to the following example is generated:
<PE1> debugging mpls ldp session
<PE1>
*0.19200353 PE1 LDP/8/Session: Serial0/2/0
Link Hello message sent on interface: Serial0/2/0

// LDP sent a hello message.
*0.19201475 PE1 LDP/8/Session:
  Session(140.3.3.3) received keep alive message on Operational state.

  // LDP received a keepalive message.
*0.19202353 PE1 LDP/8/Session: Serial0/2/0 Link Hello message received on interface: Serial0/2/0

  // LDP received a hello message.
*0.19202695 PE1 LDP/8/Session:
  Sent keep alive message to LSR: 140.3.3.3.

  // LDP sent a keepalive message.
#Dec 22 15:32:40:498 2005 PE1 LDP/5/SessionDown: Session(140.2.2.2:0. public Instance)'s state change to Down

  // The peer restarted and the session went down.
*0.19212498 PE1 LDP/8/Session: Serial0/2/0 Session(140.2.2.2) received address withdraw message on Operational state.

  // LDP received an address withdraw message.
*0.19212498 PE1 LDP/8/Session: Serial0/2/0 Session(140.2.2.2) received notification in Operational state.

  // LDP received a notification.
*0.19212498 PE1 LDP/8/Session:
  Send shut down notify,Tcp connid is 3!

  // LDP sent a shutdown notification.
*0.19212498 PE1 LDP/8/Session:
  Informed Session Down event to L2VPN

  // LDP notified L2VPN of the session down event.
*0.19215395 PE1 LDP/8/Session:
  Session counter is 0

  // The session counter had a value of 0.
*0.19215395 PE1 LDP/8/Session: Serial0/2/0 Del adjacency from adjacency list if peer exists

  // LDP deleted adjacency from the adjacency list if the peer existed.
*0.19215395 PE1 LDP/8/Session: Serial0/2/0 Link Hello message sent on interface: Serial0/2/0

  // LDP sent a hello message.
*0.19215395 PE1 LDP/8/Session: Serial0/2/0 Link Hello message received on interface: Serial0/2/0

  // LDP received a hello message.
*0.19215395 PE1 LDP/8/Session:
  Create session with LSR: 140.2.2.2

  // LDP created a session.
*0.19215395 PE1 LDP/8/Session: Serial0/2/0 Link Hello message sent on interface: Serial0/2/0

*0.19215421 PE1 LDP/8/Session: Serial0/2/0 Session(140.2.2.2) received init msg in Initialized state.
// LDP received an initialization message.
*0.19215421 PE1 LDP/8/Session:
    Session Init message sent to LSR: 140.2.2.2

// LDP sent an initialization message.
*0.19215421 PE1 LDP/8/Session: Serial0/2/0
    Sent keep alive message to LSR: 140.2.2.2.

// LDP sent a keepalive message.
*0.19215421 PE1 LDP/8/Session: Serial0/2/0
    Session(140.2.2.2)'s state changed from Initialized to Open received.

// The session’s status changed to Open received.
# Dec 22 15:32:43:437 2005 PE1 LDP/5/SessionUp: Session(140.2.2.2:0. public Instance)'s state change to Up

// The session came up.
*0.19215437 PE1 LDP/8/Session: Serial0/2/0
    Session(140.2.2.2) received keep alive message on Open Received state.

// LDP received a keepalive message.
*0.19215437 PE1 LDP/8/Session: Serial0/2/0
    Session(140.2.2.2)'s state changed from Open received to operational.

// The session’s status changed to operational.
*0.19215437 PE1 LDP/8/Session:
    Address message sent to LSR: 140.2.2.2

// LDP sent an address message to the peer.
*0.19215437 PE1 LDP/8/Session:
    Informed Session Up event to L2VPN

// LDP notified L2VPN of the session up event.
*0.19215437 PE1 LDP/8/Session: Serial0/2/0
    Session(140.2.2.2) received address message on Operational state.

// LDP received an address message.
*0.19216438 PE1 LDP/8/PDU:
    Session(140.3.3.3) received keep alive message on Operational state.

// LDP received a keepalive message.
# Enable debugging for MPLS LDP PDU on one router (PE 1, in this example). Output similar to the following example is generated:
<PE1> debugging mpls ldp pdu
<PE1>
*0.19804957 PE1 LDP/8/PDU:
    Receive UDP Message from Peer: 140.3.3.3. Message content:
    01 00 00 1c 00 00 17 bf 04 00 00 04 00 2d c0 00
    04 01 00 04 8c 03 03 03 04 02 00 04 00 00 00 00
    The message type: Hello Message.

// LDP received a hello message from 140.3.3.3.
*0.19806530 PE1 LDP/8/PDU: Serial0/2/0
    Send UDP Link Hello Message on interface: Serial0/2/0. Message content:
    01 00 00 1c 00 00 17 ab 04 00 00 04 00 0f 00 00

139
The message type: Hello Message.

// LDP sent a hello message.
*0.19807692 PE1 LDP/8/PDU:
Receive UDP Message from Peer: 140.2.2.2. Message content:
01 00 00 1c 00 00 2a 05 04 00 00 04 00 0f 00 00
04 01 00 04 8c 02 02 02 04 02 04 02 04 00 00 00 00
The message type: Hello Message.

// LDP received a hello message from 140.2.2.2.
*0.19810159 PE1 LDP/8/PDU:
Received an LDP PDU from LSR: 140.2.2.2
*0.19810159 PE1 LDP/8/PDU: Serial0/2/0
Receive TCP Message from Peer: 140.2.2.2. Message content:
02 01 00 04 00 00 2a 08
The message type: Keepalive Message.

// LDP received a keepalive message from 140.2.2.2.
*0.19811519 PE1 LDP/8/PDU:
Received an LDP PDU from LSR: 140.3.3.3
*0.19811519 PE1 LDP/8/PDU:
Receive TCP Message from Peer: 140.3.3.3. Message content:
02 01 00 04 00 00 17 c2
The message type: Keepalive Message.
*0.19811550 PE1 LDP/8/PDU: Serial0/2/0
Send UDP Link Hello Message on interface: Serial0/2/0. Message content:
01 00 00 1c 00 00 17 ac 04 00 00 04 00 00 0f 00 00
04 01 00 04 8c 01 01 01 04 02 00 04 00 00 00 00
The message type: Hello Message.
*0.19812630 PE1 LDP/8/PDU:
Receive UDP Message from Peer: 140.2.2.2. Message content:
01 00 00 1c 00 00 2a 09 04 00 00 04 00 0f 00 00
04 01 00 04 8c 02 02 02 04 02 04 02 00 04 00 00 00 00
The message type: Hello Message.
*0.19815620 PE1 LDP/8/PDU: Serial0/2/0
Send TCP Message to Peer: 140.2.2.2. Message content:
02 01 00 04 00 00 17 ad
The message type: Keepalive Message.

// LDP sent a keepalive message to 140.2.2.2.

debugging mpls lfib

Use debugging mpls lfib to enable debugging for MPLS entry synchronization or encoding and decoding.

Use undo debugging mpls lfib to disable debugging for MPLS entry synchronization or encoding and decoding.

Syntax

debugging mpls lfib [ code | sync ]
undo debugging mpls lfilb [ code | sync ]

Default
Debugging for MPLS entry synchronization and encoding/decoding is disabled.

Views
User view

Default command level
1: Monitor level

Parameters
- code: Specifies debugging for MPLS entry encoding and decoding.
- sync: Specifies debugging for MPLS entry synchronization.

Usage guidelines

⚠️ CAUTION:
Do not enable debugging for MPLS entry encoding and decoding when a large number of entries are to be sent. Otherwise, the large amounts of command output might result in system abnormality.

The output debugging information for encoding is from the main MPU, and that for decoding is from the interface card and the backup MPU.

This command is supported only on devices with single-core CPUs.

Table 76 describes output fields and messages for the debugging mpls lfilb code command.

### Table 76 Output from the debugging mpls lfilb code command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SourceId</td>
<td>Entry ID.</td>
</tr>
<tr>
<td>OperType</td>
<td>Operation type.</td>
</tr>
<tr>
<td>Size</td>
<td>Number of valid bytes in the entry.</td>
</tr>
<tr>
<td>CmdBlk Count</td>
<td>Number of entries in the compressed packet.</td>
</tr>
<tr>
<td>CmdBlk Size</td>
<td>Total number of entry bytes in the compressed packet.</td>
</tr>
</tbody>
</table>

Table 77 describes output fields and messages for the debugging mpls lfilb sync command.

### Table 77 Output from the debugging mpls lfilb sync command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>time cost</td>
<td>Time cost, in milliseconds.</td>
</tr>
<tr>
<td>total size</td>
<td>Total number of bytes.</td>
</tr>
<tr>
<td>CmdQ length</td>
<td>Number of entries in the queue.</td>
</tr>
<tr>
<td>CmdQ size</td>
<td>Total number of entry bytes in the queue.</td>
</tr>
</tbody>
</table>

Table 78 describes output fields and messages for the debugging mpls lfilb command.
Table 78 Output from the debugging mpls lfib command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MFW-BFD</td>
<td>Interaction between the MFW module and the BFD module.</td>
</tr>
<tr>
<td>Begin Process</td>
<td>Begun creating/canceling BFD sessions.</td>
</tr>
<tr>
<td>End Process</td>
<td>Stopped creating/canceling BFD sessions.</td>
</tr>
<tr>
<td>BFD MSG</td>
<td>BFD message type.</td>
</tr>
<tr>
<td>SESS</td>
<td>BFD session information: source address, destination address, and egress interface.</td>
</tr>
</tbody>
</table>

Examples

The output in the following examples was created when the maximum reservable bandwidth on VLAN-interface 23 is reconfigured under the following conditions:

- The router is operating in standalone mode.
- The router is enabled with MPLS and MPLS TE.

# On the router, enable debugging for MPLS entry coding and decoding.

```
<PE1> debugging mpls lfib code

[PE1-Vlan-interface23]
```

```
/*Oct  9 09:44:51:64 2006 PE1 MFW/7/MPLSFW PACKET:
Begin encoding..., CmdBlk Count: 0, CmdBlk Size: 0

// MPLS begun encoding. The number of entries in the compressed packet was 0, and the total number of entry bytes in the compressed packet was 0.
/*Oct  9 09:44:51:64 2006 PE1 MFW/7/MPLSFW PACKET:
Encode entry, SourceId: 12, OperType: 25, Size: 12

// MPLS was encoding an entry: the ID is 12, operation type is 25, and number of valid bytes is 12.
/*Oct  9 09:44:51:64 2006 PE1 MFW/7/MPLSFW PACKET:
Encode entry, SourceId: 12, OperType: 25, Size: 12

// MPLS encoded the entry: the ID is 12, operation type is 25, and number of valid bytes is 12.
/*Oct  9 09:44:51:64 2006 PE1 MFW/7/MPLSFW PACKET:
End encoding..., CmdBlk Count: 2, CmdBlk Size: 32

// Encoding ended. The number of entries in the compressed packet is 2, and the total number of entry bytes in the compressed packet is 32.
```

```
/*Oct  9 09:44:54:368 2006 PE1 MFW/7/MPLSFW PACKET:Slot=1;
Begin decoding..., CmdBlk Count: 2, CmdBlk Size:32

// MPLS began decoding. The number of entries in the compressed packet is 2, and the total number of entry bytes in the compressed packet is 32.
/*Oct  9 09:44:54:368 2006 PE1 MFW/7/MPLSFW PACKET:Slot=1;
Decode entry, SourceId: 12, OperType: 25, Size: 12

// MPLS was decoding an entry: the ID is 12, operation type is 25, and number of valid bytes is 12.
/*Oct  9 09:44:54:384 2006 PE1 MFW/7/MPLSFW PACKET:Slot=1;
Decode entry, SourceId: 12, OperType: 25, Size: 12
```

142
// MPLS decoded the entry: the ID is 12, operation type is 25, and number of valid bytes is 12.
*Oct 9 09:44:54:384 2006 PE1 MFW/7/MPLSFW PACKET:Slot=1;
End decoding...

// Decoding ended.

# On the router, enable debugging for MPLS entry synchronization.
<PE1>debugging mpls lfib sync
...
[PE1-Vlan-interface23]
*Oct 9 09:44:29:194 2006 PE1 MFW/7/MPLSFW PACKET:
Send entry, count: 2, time cost: 33640

// MPLS sent two entries in 33640 milliseconds.
*Oct 9 09:44:29:194 2006 PE1 MFW/7/MPLSFW PACKET:
Send entry, count: 2, total size: 32, CmdQ length: 0, CmdQ size: 0
// MPLS sent two entries. The total number of bytes is 32, the number of entries in the queue is 0, and the total number of entry bytes in the queue is 0.
*Oct 9 09:44:29:194 2006 PE1 MFW/7/MPLSFW PACKET:Slot=1;
Receive entry, count: 2, total size: 32

// MPLS received two entries. The total number of bytes is 32.

debugging mpls lfib slot drv

Use debugging mpls lfib slot drv to enable debugging for MFW entry distribution to driver.

Use the undo debugging mpls lfib slot drv command to disable debugging for MFW entry distribution to driver.

Syntax

Distributed devices/centralized IRF devices:

debugging mpls lfib slot slot-number drv
undo debugging mpls lfib slot slot-number drv

Distributed IRF devices:

debugging mpls lfib chassis chassis-number slot slot-number drv
undo debugging mpls lfib chassis chassis-number slot slot-number drv

Default

All types of debugging for MFW entry distribution to driver are disabled.

Views

User view

Default command level

1: Monitor level

Parameters

lfib: Specifies debugging for MFW.

slot slot-number: Specifies debugging for a card. The slot-number argument specifies the number of the slot that holds the card. (Distributed devices.)
slot slot-number: Specifies debugging for an IRF member device. The slot-number argument specifies the ID of an IRF member device. (Centralized IRF devices.)

drv: Specifies debugging for distribution to driver.

chassis chassis-number slot slot-number: Specifies debugging for a card on an IRF member device. The chassis-number argument specifies the ID of the IRF member device. The slot-number argument specifies the number of the slot that holds the card. (Distributed IRF devices.)

Usage guidelines

This command is supported only on devices with single-core CPUs.

Table 79 describes output fields and messages for the debugging mpls lfib slot drv command.

Table 79 Output from the debugging mpls lfib slot drv command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MFW</td>
<td>Forwarding module.</td>
</tr>
<tr>
<td>DRV</td>
<td>Driver module.</td>
</tr>
<tr>
<td>VC</td>
<td>Virtual circuit.</td>
</tr>
<tr>
<td>ILM</td>
<td>Incoming label mapping.</td>
</tr>
<tr>
<td>PW</td>
<td>Pseudo wire.</td>
</tr>
<tr>
<td>del</td>
<td>Delete.</td>
</tr>
<tr>
<td>FRR</td>
<td>Fast reroute.</td>
</tr>
<tr>
<td>MFW-DRV</td>
<td>Interaction between the MFW module and the DRV module.</td>
</tr>
<tr>
<td>NHLFE</td>
<td>Next hop label forwarding entry.</td>
</tr>
<tr>
<td>DrvContext</td>
<td>Information returned to the upper layer by the drive.</td>
</tr>
</tbody>
</table>

result is  
- 0—Success. 
- 1—Failure.
Examples

As shown in Figure 1, PE 1 and PE 2 are directly connected. PE 2 is connected to PE 3 through P 1 and P 2. Two equal-cost Martini VCs are established between PE 2 and PE 3. One VC is through P 1 and the other through P 2.

Figure 1 Network diagram for equal cost LSPs

# On PE 2, enable debugging for MFW entry distribution to the driver. When PE 2 and P 1 are disconnected, output similar to the following example is generated:

```
<PE2> debugging mpls ifib slot 3 drv
*Apr 19 15:59:58:853 2008 PE2 MFW/7/MPLSFW PACKET:Slot=3;
INFO : The content of send entry toDrv is:
AccessType is 0
Transtype is 4
CtrlWord is 0
Mtu is 1500
InIfIndex is 0x190006d
InVcLabel is 98350
OutVCLabel is 98331
RmtVTag is 4
LspTnlId is 11206659
CType is 0
VlanID is 4
OutIfIndex is 0x1900063
OutAtIndex is 0
SlotNumber is 3
LabelStackDepth is 1, label is 1082
TnlNextHop is 20.20.20.2
The outest tunnel is TE tunnel, if name Tunnel3/0/1 (0x1af0000)
The type of bandwidth is 0
The bandwidth is 0
DrvContext is 50266404, 4294901764
```

// A VC entry was added for the path from PE 2 through P 2 to PE 3.

```
*Apr 19 15:59:59:578 2008 PE2 MFW/7/MPLSFW PACKET:Slot=3;
MFW-VLL: The content of VLL DelInfo to Drv is:
  In IfIndex is 0x190006d
```
In VCLabel is 98350
TunnelId is 0xab0003
EncapType is 0
VlanID is 4
DrvContext is 50266404, 4294901764

// The first equal-cost VC entry was deleted.
*Apr 19 15:59:59:888 2008 PE2 MFW/7/MPLSFW PACKET:Slot=3;
MFW-VLL: The content of VLL DelInfo to Drv is:
In IfIndex is 0x190006d
In VCLabel is 98350
TunnelId is 0xab0004
EncapType is 0
VlanID is 4
DrvContext is 0, 0

// The second equal-cost VC entry was deleted.
*Apr 19 16:00:00:178 2008 PE2 MFW/7/MPLSFW PACKET:Slot=3;
MFW-DVR: Modify VLL VC equal entry. TunnelID is 11206660, New Entry count is 1, Old Entry count is 2, result is 0.

// MPLS informed the driver of the update of equal cost entries. The tunnel ID is 11206660, number of new entries is 1, and number of old entries is 2. The result (success) that the drive returned is success (0 indicates success, and 1 indicates failure).

# When the link between PE 2 and P 1 is resumed, output similar to the following example is generated:

<PE2> debugging mpls lfib slot 3 drv
*Apr 19 16:00:28:853 2008 PE2 MFW/7/MPLSFW PACKET:Slot=3;
INFO : The content of send entry to Drv is:
AccessType is 0
Transtype is 4
CtrlWord is 0
Mtu is 1500
InIfIndex is 0x190006d
InVcLabel is 98350
OutVCLabel is 98331
RmtVTag is 4
LspTnlId is 11206659
CEType is 0
VlanID is 4
OutIfIndex is 0x1900063
OutAtIndex is 0
SlotNumber is 3
LabelStackDepth is : 1 , label is :1082
TnlNextHop is 20.20.20.2
The outest tunnel is TE tunnel, if name Tunnel3/0/1 (0x1af0000)
The type of bandwidth is 0
The bandwidth is 0
DrvContext is 50266404, 4294901764

// The first equal-cost VC entry was added.
*Apr 19 16:00:29:579 2008 PE2 MFW/7/MPLSFW PACKET:Slot=3;
INFO : The content of send entry to Drv is:
AccessType is 0
Transtype is 4
CtrlWord is 0
Mtu is 1500
InIfIndex is 0x190006d
InVcLabel is 98350
OutVcLabel is 98331
RmtVTag is 4
LspTnlId is 11206660
CETYPE is 0
VlanID is 4
OutIfIndex is 0x1900063
OutAtIndex is 0
SlotNumber is 3
LabelStackDepth is 1, label is 1083
TnlNextHop is 20.20.20.2
The outest tunnel is TE tunnel, if name Tunnel3/0/2 (0x1af0001)
The type of bandwidth is 0
The bandwidth is 0
DrvContext is 0, 0

// The second equal-cost VC entry was added.
*Apr 19 16:00:30:288 2008 PE2 MFW/7/MPLSFW PACKET:Slot=3;
MFW-VLL: The content of VLL DelInfo to Drv is:
    In IfIndex is 0x190006d
    In VcLabel is 98350
    TunnelId is 0xab0003
    EncapType is 0
    VlanID is 4
    DrvContext is 50266404, 4294901764

// The VC entry was deleted for the path from PE 2 through P 2 to PE 3.
*Apr 19 16:00:30:598 2008 PE2 MFW/7/MPLSFW PACKET:Slot=3;
MFW-DRV: Modify VLL VC equal entry. TunnelID is 11206660, New Entry count is 2, Old Entry count is 1, result is 0.

// MPLS informed the driver of the update of the VC entries. The tunnel ID is 11206660, number of new entries is 2, and number of old entries is 1. The result (success) that the drive returned is success (0 indicates success, and 1 indicates failure).

delmsg mpls lspc

Use **debugging mpls lspc** to enable debugging for the LSP control module.

Use **undo debugging mls lspc** to disable debugging for the LSP control module.

**Syntax**

```plaintext
debugging mpls lspc { all | error | event | packet }
undo debugging mpls lspc { all | error | event | packet }
```
Default

Debugging for the LSP control module is disabled.

Views

User view

Default command level

1: Monitor level

Parameters

- **all**: Specifies all types of debugging.
- **error**: Specifies error debugging.
- **event**: Specifies event debugging.
- **packet**: Specifies packet debugging.

Usage guidelines

Table 80 describes the output fields and messages for the `debugging mpls lspc` command.

Table 80 Output from the debugging mpls lspc command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DS TLV is invalid.</td>
<td>The downstream mapping TLV was invalid.</td>
</tr>
<tr>
<td>Socket send error!</td>
<td>MPLS failed to send packets by socket.</td>
</tr>
</tbody>
</table>

Examples

The output in the following example was created under the following conditions:

- Three routers are directly connected to each other.
- MPLS and MPLS LDP are configured on all the routers. An LDP session has been established between any two routers.

# Enable all types of debugging for the LSP control module on one router. When the ping lsp command is executed on another router, output similar to the following example is generated:

```
<Sysname> terminal debugging
<Sysname> debugging mpls lspc all
<Sysname>
*Nov 29 17:01:54:548 2006 Sysname LSPC/7/Event:
  Receive echo request message on egress.

// LSP control received an echo request message on the egress.
*Nov 29 17:01:54:548 2006 Sysname LSPC/7/Packet:
  Receive echo request message on egress. Lenth: 100
  00 01 00 00 01 02 00 00 00 00 00 00 01 00 00 00 01
  00 4c c6 00 00 02 a5 00 00 00 00 00 00 00 00 00
  00 01 00 00 00 01 00 05 30 30 30 30 20 00 00 00
  00 03 00 30 02 41 42 43 44 45 46 47 48 49 4a 4b
  4c 4d 4e 4f 50 51 52 53 54 55 56 57 58 59 5a 41
  42 43 44 45 46 47 48 49 4a 4b 4c 4d 4e 4f 50 51
  52 53 54 55

// LSP control received an echo request message on the egress. The packet was 100-byte long.
```
debugging mpls management

Use **debugging mpls management** to enable debugging for the MPLS LSPM module.

Use **undo debugging mpls management** to disable MPLS LSPM debugging.

**Syntax**

```
debugging mpls management { all | event | interface | policy | process | tunnel }
undo debugging mpls management { all | event | interface | policy | process | tunnel }
```

**Default**

All types of MPLS LSPM debugging are disabled.

**Views**

User view

**Default command level**

1: Monitor level

**Parameters**

- **all**: Specifies all types of debugging for MPLS LSP management (LSPM).
- **event**: Specifies debugging for MPLS LSPM events.
- **interface**: Specifies debugging for MPLS LSPM interfaces sending and receiving messages.
- **policy**: Specifies debugging for policies that LDP uses to create LSPs.
- **process**: Specifies debugging for MPLS LSPM processing.
- **tunnel**: Specifies debugging for MPLS LSPM tunnels.

**Usage guidelines**

When the LSPM module is not operating correctly, you can use the **debugging mpls management** command to locate the problem. This command might decrease system performance. Use this command only when required.

---

**NOTE:**

This command is supported only on devices with single-core CPUs.
Table 81 describes output fields and messages for the **debugging mpls management interface** command.

### Table 81 Output from the debugging mpls management interface command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Received Signal XX message</td>
<td>LSPM received a message from the signaling module, such as a signaling DELETE or signaling CREATE message.</td>
</tr>
<tr>
<td>OutIfIndex</td>
<td>Outgoing interface index.</td>
</tr>
<tr>
<td>InIfIndex</td>
<td>Incoming interface index.</td>
</tr>
<tr>
<td>Received Route XX message</td>
<td>LSPM received a message from the routing module, such as a CREATE or DELETE message.</td>
</tr>
</tbody>
</table>

Table 82 describes output fields and messages for the **debugging mpls management policy** command.

### Table 82 Output from the debugging mpls management policy command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Received Route XX message</td>
<td>LSPM received a message from Route module, such as a route ADD or a route DELETE message.</td>
</tr>
<tr>
<td>LSP XX</td>
<td>Performed a specific operation of the LSP, such as DELETE and ADD.</td>
</tr>
<tr>
<td>IfIndex</td>
<td>Outgoing interface index.</td>
</tr>
<tr>
<td>LSPIndex</td>
<td>LSP index.</td>
</tr>
<tr>
<td>Route Does not Satisfy Policy!. Sig Prot is informed.</td>
<td>LSPM notified the signaling module because the route did not satisfy the policy.</td>
</tr>
</tbody>
</table>

Table 83 describes output fields and messages for the **debugging mpls management process** command.

### Table 83 Output from the debugging mpls management process command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OutIfIndex</td>
<td>Outgoing interface index.</td>
</tr>
<tr>
<td>InIfIndex</td>
<td>Incoming interface index.</td>
</tr>
<tr>
<td>evt/lsptype X/X</td>
<td>Event type and LSP type.</td>
</tr>
<tr>
<td>token</td>
<td>LSP token.</td>
</tr>
</tbody>
</table>

Table 84 describes output fields and messages for the **debugging mpls management tunnel** command.

### Table 84 Output from the debugging mpls management tunnel command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Begin physic check</td>
<td>LSPM has begun to check the physical interface of the tunnel.</td>
</tr>
<tr>
<td>ulIfIndex: X</td>
<td>Physical interface index.</td>
</tr>
<tr>
<td>ulTunnelIfIndex</td>
<td>Tunnel interface index.</td>
</tr>
</tbody>
</table>
Examples

The output in the following examples was created under the following conditions:

- Two routers are connected directly.
- MPLS and MPLS LDP are configured on both routers. An LDP session has been successfully established between the routers.

# Enable debugging for MPLS management interface on one router (PE 1, in this example). When MPLS LDP is reset on the other router, output similar to the following example is generated:

```bash
<PE1> debugging mpls management interface
*0.22430627 PE1 LSPM/8/LSPM INTERFACE:
   Received Signal DELETE message, VRF Index: 0 destination: 0x8c010101 destination mask: 0xffffffff Nexthop: 0x7f000001 OutIfIndex: 0x150000 InIfIndex: 0x300010

   // LSPM received a signaling DELETE message.
*0.22430627 PE1 LSPM/8/LSPM INTERFACE:
   Received Signal DELETE message, VRF Index: 0 destination: 0x8c030303 destination mask: 0xffffffff Nexthop: 0x8c0c0102 OutIfIndex: 0x300010 InIfIndex: 0x0

   // LSPM received a signaling DELETE message.
```

# Dec 22 16:22:17:972 2005 PE1 LDP/5/SessionUp: Session(140.2.2.2:0. public Instance)'s state change to Up

```
// The session came up again.
*0.22435678 PE1 LSPM/8/LSPM INTERFACE:
   Received Route CREATE message, destination: 0x8c010101 destination mask: 0xffffffff.

   // LSPM received a route CREATE message.
*0.22435901 PE1 LSPM/8/LSPM INTERFACE:
   Received Signal CREATE message, VRF Index: 0 destination: 0x8c010101 destination mask: 0xffffffff Nexthop: 0x7f000001 OutIfIndex: 0x150000 InIfIndex: 0x300010

   // LSPM received a signaling CREATE message.
```

# Enable debugging for MPLS management policy on one router (PE 1, in this example). When MPLS LDP is reset on the other router, output similar to the following example is generated:

```bash
<PE1> debugging mpls management policy
<PE1>
```

# Dec 23 10:12:18:552 2005 PE1 LDP/5/SessionDown: Session(140.2.2.2:0. public Instance)'s state change to Down

```
// The peer restarted and the session went down.
*0.92036640 PE1 LSPM/8/LSPM POLICY:
   LSP deleted, VRF:0 Destination: 0x8c010101 Destination mask: 0xffffffff Nexthop: 0x7f000001 IfIndex:0x150000 LSPIndex:0x2443.

   // LSPM deleted an LSP.
*0.92036640 PE1 LSPM/8/LSPM POLICY:
   LSP deleted, VRF:0 Destination: 0x8c030303 Destination mask: 0xffffffff Nexthop: 0x8c0c0102 IfIndex:0x300010 LSPIndex:0x2444.

   // LSPM deleted an LSP.
*0.92036640 PE1 LSPM/8/LSPM POLICY:
   LSP deleted, VRF:0 Destination: 0x8c020202 Destination mask: 0xffffffff Nexthop: 0x8c0c0102 IfIndex:0x300010 LSPIndex:0x2445.

   // LSPM deleted an LSP.
```
// LSPM deleted an LSP.
*0.92036640 PE1 LSPM/8/LSPM POLICY:
Route Doesnot Satisfy Policy!. Sig Prot is informed.

// The route did not satisfy the policy. LSPM notified the signaling module.
*0.92036640 PE1 LSPM/8/LSPM POLICY:
Calling Signalling protocol for VRF:0 Dst Addr:0x8c020202 Dst Mask:0xffffffff NextHop:
0x8c0c0102 If Index: 0x300010 Event: 13

// LSPM called the signaling protocol.
*0.92036640 PE1 LSPM/8/LSPM POLICY:
Route Doesnot Satisfy Policy!. Sig Prot is informed.

// The route did not satisfy the policy. LSPM notified the signaling module.
*0.92036640 PE1 LSPM/8/LSPM POLICY:
Calling Signalling protocol for VRF:0 Dst Addr:0x8c030303 Dst Mask:0xffffffff NextHop:
0x8c0c0102 If Index: 0x300010 Event: 13

// LSPM called the signaling protocol.
*0.92041801 PE1 LSPM/8/LSPM POLICY:
Received Route Add message, VRF:0 Destination: 0x8c010101 Destination mask: 0xffffffff
Nexthop: 0x7f000001 IfIndex:1376256.

// LSPM received a route ADD message.
*0.92041801 PE1 LSPM/8/LSPM POLICY:
Calling Signalling protocol for VRF:0 Dst Addr:0x8c010101 Dst Mask:0xffffffff NextHop:
0x7f000001 If Index: 0x150000 Event: 1

// LSPM called the signaling protocol for processing.
*0.92041801 PE1 LSPM/8/LSPM POLICY:
Route added successfully

// LSPM added the route successfully.
*0.92042024 PE1 LSPM/8/LSPM POLICY:
LSP added, VRF:0 Destination: 0x8c010101 Destination mask: 0xffffffff Nexthop:0x7f000001
IfIndex:0x150000 LSPIndex:0x2446.

// LSPM added an LSP.
*0.92066067 PE1 LSPM/8/LSPM POLICY:
LSP added, VRF:0 Destination: 0x8c020202 Destination mask: 0xffffffff Nexthop:0x8c0c0102
IfIndex:0x300010 LSPIndex:0x2447.

// LSPM added an LSP.
*0.92066083 PE1 LSPM/8/LSPM POLICY:
LSP added, VRF:0 Destination: 0x8c030303 Destination mask: 0xffffffff Nexthop:0x8c0c0102
IfIndex:0x300010 LSPIndex:0x2448.

// LSPM added an LSP.

# Enable debugging for MPLS management process on one router (PE 1, in this example). When MPLS LDP is reset on the other router, output similar to the following example is generated:
<PE1> debugging mpls management process
<PE1>
The peer restarted and the session went down.

LSPM received a delete LSP message from the signaling protocol.

LSPM received a signaling message.

Received Signal DELETE message, VRF Index: 0 destination: 0x8c010101 destination mask: 0xffffffff Nexthop: 0x7f000001 OutIfIndex: 0x150000 InIfIndex: 0x300010

LSPM set the MTU.

Succeeded in sending forwarding message to MFW.

LSPM notified the TNLM.

Free Token success: 557247

LSPM freed the token successfully.
*0.25271860 PE1 LSPM/8/LSPM PROCESS:
  Setting MTU = LSP MTU
*0.25271860 PE1 LSPM/8/LSPM PROCESS:
  Succeeded in sending forwarding message to MFW.
*0.25271860 PE1 LSPM/8/LSPM PROCESS:
  Notify to TNLM SUCCESS. Event: 2, LspIndex: 9267, 0x8c030303/0x20/token557248.
*0.25271860 PE1 LSPM/8/LSPM PROCESS:
  Free Token success: 557248
  // LSPM freed the token successfully.

Dec 22 17:05:59:280 2005 PE1 LDP/5/SessionUp: Session(140.2.2.2:0. public Instance)'s state change to Up
  // The session came up again.
*0.25274675 PE1 LSPM/8/LSPM PROCESS:
  Alloc Token success: 557249/0x300010
  // LSPM allocated a token successfully.
*0.25274675 PE1 LSPM/8/LSPM PROCESS:
  Rcv create lsp msg from signaling protocol 0x8c020202/0xffffffff In If - 0x0
  // LSPM received a create LSP message from the signaling protocol.
*0.25274690 PE1 LSPM/8/LSPM PROCESS:
  Received Signaling message.
*0.25274690 PE1 LSPM/8/LSPM PROCESS:
  Received Signal CREATE message, VRF Index: 0 destination: 0x8c020202 destination mask: 0xffffffff Nexthop: 0x8c0c0102 OutIfIndex: 0x300010 InIfIndex: 0x0
*0.25274690 PE1 LSPM/8/LSPM PROCESS:
  Setting MTU = LSP MTU
*0.25274690 PE1 LSPM/8/LSPM PROCESS:
  Add route success, VRF:0 Destination: 0x8c020202 Destination mask: 0xffffffff Nexthop: 0x8c0c0102 IfIndex:0x300010 .
  // LSPM added a route successfully.
*0.25274690 PE1 LSPM/8/LSPM PROCESS:
  Added LSP in LM Successfully
  // LSPM added an LSP successfully.
*0.25274690 PE1 LSPM/8/LSPM PROCESS:
  Succeeded in sending forwarding message to MFW.
  // LSPM sent the forwarding message to MFW successfully.
*0.25274690 PE1 LSPM/8/LSPM PROCESS:
  Notify to TNLM SUCCESS. Event: 1, LspIndex: 9268, 0x8c020202/0x20/token557249.
  // LSPM notified the TNLM.
Received Signaling message.
*0.25275812 PE1 LSPM/8/LSPM PROCESS:
    Received Signal CREATE message, VRF Index: 0 destination: 0x8c010101 destination mask: 0xffffffff Nexthop: 0x7f000001 OutIfIndex: 0x150000 InIfIndex: 0x300010
*0.25275812 PE1 LSPM/8/LSPM PROCESS:
    Setting MTU = LSP MTU
*0.25275812 PE1 LSPM/8/LSPM PROCESS:
    Error: No LSP found for the route.
*0.25275812 PE1 LSPM/8/LSPM PROCESS:
    Added LSP in LM Successfully
*0.25275812 PE1 LSPM/8/LSPM PROCESS:
    Succeeded in sending forwarding message to MFW.

// LSPM sent the forwarding message MFW successfully.

# Enable debugging for MPLS management tunnel on one router (PE 1, in this example). Output similar to the following example is generated when these tasks are performed on the other router:

- Shuts down the interface connected to PE 1.
- Brings up the interface again.

<PE1> debugging mpls management tunnel
<PE1>
*0.94534296 PE1 LSPM/8/LSPM TUNNEL:
    Begin physic check,ulIfIndex:3407872
*0.94534296 PE1 LSPM/8/LSPM TUNNEL:
    TE tunnel get mode data ok when check physic,ulTunnelIfIndex:3407872
*0.94534296 PE1 LSPM/8/LSPM TUNNEL:
    End physic check,the tunnel can up

// The interface was up. Physical interface checking succeeded, and the tunnel can come up.
%Dec 23 10:51:12:310 2005 PE1 IFNET/5/UPDOWN:
    Line protocol on the interface Tunnel0 is DOWN

// Because the peer interface was shut down, the tunnel went down.
*0.94564310 PE1 LSPM/8/LSPM TUNNEL:
    Begin physic check,ulIfIndex:3407872
*0.94564310 PE1 LSPM/8/LSPM TUNNEL:
    TE tunnel get mode data ok when check physic,ulTunnelIfIndex:3407872
*0.94564310 PE1 LSPM/8/LSPM TUNNEL:
    End physic check,the tunnel can't up

// Physical interface checking failed, and the tunnel cannot come up.
%Dec 23 10:52:12:591 2005 PE1 IFNET/5/UPDOWN:
    Line protocol on the interface Tunnel0 is UP

// After the undo shutdown operation was performed on the peer, the tunnel came up.
*0.94629360 PE1 LSPM/8/LSPM TUNNEL:
    Begin physic check,ulIfIndex:3407872
*0.94629360 PE1 LSPM/8/LSPM TUNNEL:
    TE tunnel get mode data ok when check physic,ulTunnelIfIndex:3407872
*0.94629360 PE1 LSPM/8/LSPM TUNNEL:
    End physic check,the tunnel can up

// Physical interface checking succeeded. The tunnel can come up.
debugging mpls packet

Use `debugging mpls packet` to enable debugging for MPLS packet forwarding.

Use `undo debugging mpls packet` to disable debugging for MPLS packet forwarding.

**Syntax**

```text
debugging mpls packet [ acl acl-number ] [ error ] [ inlabel outer-in-label [ inner-in-label ] ] [ l2vpn-in-interface interface-type interface-number ]

undo debugging mpls packet
```

**Default**

Debugging for MPLS packet forwarding is disabled.

**Views**

User view

**Default command level**

1: Monitor level

**Parameters**

- `acl`: Specifies debugging for MPLS packets that match the specified ACL.
- `acl-number`: Specifies an advanced ACL by its number in the range of 3000 to 3999.
- `error`: Specifies debugging for MPLS error packets.
- `inlabel`: Specifies debugging for MPLS packets with the specified incoming label.
- `outer-in-label`: Specifies the outer incoming label in the range of 0 to 1048576.
- `inner-in-label`: Specifies the inner incoming label in the range of 16 to 1048576.
- `l2vpn-in-interface`: Specifies debugging for MPLS packets that are bound to an L2VPN interface.
- `interface-type interface-number`: Specifies an interface by its type and number.

**Usage guidelines**

Table 85 describes output fields and messages for the `debugging mpls packet` command on single-core devices.

**Table 85** Output from the `debugging mpls packet` command (on single-core devices)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MPLSFW_Input</td>
<td>MFW received MPLS packets.</td>
</tr>
<tr>
<td>L2VPNFoward</td>
<td>Function for processing.</td>
</tr>
<tr>
<td>PKTTYPE_MPLS</td>
<td>The packet type is MPLS.</td>
</tr>
<tr>
<td>Label(s)</td>
<td>Labels, including the inner label (private network) and outer label (public network).</td>
</tr>
<tr>
<td>PktLen</td>
<td>Packet length.</td>
</tr>
</tbody>
</table>

Table 86 describes output fields and messages for the `debugging mpls packet` command on multi-core devices.
Table 86 Output from the debugging mpls packet command (on multi-core devices)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>L2VPN Input</td>
<td>L2VPN packet forwarding from PE to PE.</td>
</tr>
<tr>
<td>L2VPN Output</td>
<td>Sent L2VPN packets from PE to CE.</td>
</tr>
<tr>
<td>CCC</td>
<td>L2VPN local CCC packets.</td>
</tr>
<tr>
<td>REMOTE</td>
<td>L2VPN remote CCC packets.</td>
</tr>
<tr>
<td>VCLabel</td>
<td>L2VPN VC label.</td>
</tr>
<tr>
<td>PktLen</td>
<td>Packet length.</td>
</tr>
<tr>
<td>Label(s)</td>
<td>Labels, including the inner label (private network) and outer label (public network).</td>
</tr>
</tbody>
</table>

Examples

An MPLS L3VPN is established on five routers—two PE routers, a P router, and two CE routers.

# Execute the debugging mpls packet command on a PE. When the PE is a single-core router, output similar to the following example is generated:

```
<PE1> debugging mpls packet
<PE1>
*0.3815709 F-RT1 MFW/8/MPLSFW PACKET:
PUSH Label=3
PKTTYPE_IP :Sending to Vlan3, Dest=15.33.33.33, Nexthop=15.20.1.1
  // MPLS added label 3 into a packet and sent the packet.
*0.3824701 F-RT1 MFW/8/MPLSFW PACKET:
PUSH Label=1024, EXP=0, TTL=255
PUSH Label=1024, EXP=0, TTL=255
PKTTYPE_MPLS :Sending to Vlan3, PktLen=92, Label(s)=1024,1024, EXP=0, TTL=255
  // MPLS added inner in-label 1024 and outer in-label 1024 into a packet and sent the packet to VLAN 3.
*0.3824779 F-RT1 MFW/8/MPLSFW PACKET:Slot=2;
MPLSFW_Input: Receiving from Vlan3, PktLen=88, Label(s)=1024, EXP=0, TTL=254
POP Label=1024, EXP=0, TTL=254
Delivering up to IP!
  // MPLS received a packet with the length 88 and label 1024 from VLAN 3, popped label 1024, and forwarded the packet as an IP packet.
*0.3824701 F-RT1 MFW/8/MPLSFW PACKET:Slot=2;
PKTTYPE_IP :Sending to Slot 6, Dest=15.40.1.1, InnerLabel=1024, Vpn=1, Token=9371650
  // MPLS sent the interface a packet with the destination address 15.40.1.1 and inner label 1024.
*0.3824779 F-RT1 MFW/8/MPLSFW PACKET:
PUSH Label=1024, EXP=0, TTL=255
PUSH Label=1024, EXP=0, TTL=255
PKTTYPE_MPLS :Sending to Vlan3, PktLen=92, Label(s)=1024,1024, EXP=0, TTL=255
  // MPLS added inner in-label 1024 and outer in-label 1024 into a packet and sent the packet to VLAN 3.
*0.3824779 F-RT1 MFW/8/GPLSFW PACKET:Slot=2;
```

157
MPLSFW_Input: Receiving from Vlan3, PktLen=88, Label(s)=1024, EXP=0, TTL=254
POP Label=1024, EXP=0, TTL=254
Delivering up to IP!

// MPLS received a packet with the length 88 and label 1024 from VLAN3, popped label 1024, and forwarded the packet as an IP packet.
*0.3824779 F-RT1 MFW/8/MPLSFW PACKET:Slot=2;
PKTTYPE_IP :Sending to Slot 6, Dest=15.40.1.1, InnerLabel=1024, Vpn=1, Token=9371650

// MPLS sent the interface a packet with the destination address 15.40.1.1 and inner label 1024.
*0.3824851 F-RT1 MFW/8/GPLSFW PACKET:
PUSH Label=1024, EXP=0, TTL=255
PUSH Label=1024, EXP=0, TTL=255
PKTTYPE_MPLS :Sending to Vlan3, PktLen=92, Label(s)=1024,1024, EXP=0, TTL=255

// MPLS added inner in-label 1024 and outer in-label 1024 into a packet and sent the packet to VLAN 3.
*0.3824851 F-RT1 MFW/8/GPLSFW PACKET:Slot=2;
MPLSFW_Input: Receiving from Vlan3, PktLen=88, Label(s)=1024, EXP=0, TTL=254
POP Label=1024, EXP=0, TTL=254
Delivering up to IP!

// MPLS received a packet with the length 88 and label 1024 from VLAN3, popped label 1024, and forwarded the packet as an IP packet.

# Execute the debugging mpls packet command on a PE. When the PE is a multi-core router, output similar to the following example is generated:

<PE> debugging mpls packet

*Jun 5 11:30:07:589 2007 PE DPMPLS/7/MPLSFW PACKET:
MPLS Input: Receiving from GE0/1, PktLen = 63, Label(s) = 1026, EXP = 6, TTL = 254!

// MPLS received a packet with the packet length 63 and label 1026 from interface GigabitEthernet 0/1.
*Jun 5 11:30:07:589 2007 PE DPMPLS/7/GPLSFW PACKET:
MPLS Forward: Pop all labels and perform IP forwarding!PktLen = 59, TTL = 253, SrcAddr = 66.8.1.1, DstAddr = 66.2.1.1!

// MPLS popped the outer label 1026 and forwarded the packet as an IP packet.
*Jun 5 11:30:01:698 2006 PE DPMPLS/7/GPLSFW PACKET:
MPLS Input: Receiving from GE0/1, PktLen = 92, Label(s) = 1026,1024, EXP = 0, TTL = 254!

// MPLS received a packet with the packet length 63, outer label 1026, and inner label 1024 from interface GigabitEthernet 0/1.
*Jun 5 11:30:01:698 2006 PE DPMPLS/7/GPLSFW PACKET:
MPLS Forward: Pop all labels and perform IP forwarding!PktLen = 84, TTL = 253, SrcAddr = 65.1.1.1, DstAddr = 45.1.1.2!

// MPLS popped the outer label 1026 and inter label 1024 and forwarded the packet as an IP packet.
*Jun 5 11:30:26:335 2006 PE DPMPLS/7/GPLSFW PACKET:
Receiving IP Packet!PktLen = 84, TTL = 254, SrcAddr = 45.1.1.2, DstAddr = 45.1.1.1!
Begin MPLS forwarding!InLabel = 1024, TunnelId = 6, VrfIndex = 1

// MPLS received a packet with the length 84, TTL 254, source IP address 45.1.1.2, and destination IP address 65.1.1.1. MPLS forwarded the packet as an MPLS packet, with the label 1024, tunnel token 6, and private network VRF index 1.
*Jun 5 11:30:26:335 2006 PE DPMPLS/7/GPLSFW PACKET:
debugging tnlm

Use `debugging tnlm` to enable debugging for tunnel management.
Use `undo debugging tnlm` to disable debugging for tunnel management.

**Syntax**
```
debugging tnlm { all | error | event | hsb }
undo debugging tnlm { all | error | event | hsb }
```

**Default**
Debugging for tunnel management is disabled.

**Views**
User view

**Default command level**
1: Monitor level

**Parameters**
- **all**: Specifies all types of debugging for tunnel management.
- **event**: Specifies event debugging for tunnel management.
- **error**: Specifies error debugging for tunnel management.
- **hsb**: Specifies hot standby debugging for tunnel management.

**Examples**
```
# Enable all types of debugging for tunnel management.
<Sysname> debugging tnlm all
```
BFD debugging commands

The output description tables in this document only contain fields and messages that require an explanation.

debugging bfd all

Use `debugging bfd all` to enable all BFD debugging.

Use `undo debugging bfd all` to disable all BFD debugging.

Syntax

```
debugging bfd all
undo debugging bfd all
```

Default

All BFD debugging is disabled.

Views

User view

Default command level

1: Monitor level

Usage guidelines

BFD debugging information displayed includes system running time, device name, BFD, debugging information level, module name, and event.

Table 87 describes output fields and messages for the `debugging bfd all` command.

**Table 87 Output from the debugging bfd all command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PKT_RCV</td>
<td>Name of the receiving module.</td>
</tr>
<tr>
<td>MBuf</td>
<td>Data blocks of encapsulated packets.</td>
</tr>
<tr>
<td>FSM</td>
<td>State machine name.</td>
</tr>
<tr>
<td>PKT_SND</td>
<td>Name of the sending module.</td>
</tr>
<tr>
<td>TMR</td>
<td>Timer name.</td>
</tr>
<tr>
<td>SCM</td>
<td>Session control module.</td>
</tr>
<tr>
<td>IPC</td>
<td>Inter-processor communication module.</td>
</tr>
<tr>
<td>LD</td>
<td>Local session authentication code.</td>
</tr>
<tr>
<td>RD</td>
<td>Remote session authentication code.</td>
</tr>
</tbody>
</table>

Examples

```
# Enable all BFD debugging. When two ends attempt to establish a BFD session, output similar to the following example is generated:
```

160
The interface card (Slot 1) received the Init message from the peer and sent the message to the active MPU.

The session status changed to Init.

BFD sent an Init message to the peer.

BFD reset the message send timer.

BFD created the detect timer.

The interface card 1 received an up message from the peer and sent the message to the active MPU.

The session status changed to up.

BFD sent an up message to the peer.

BFD switched session maintenance from the active MPU (No. 6) to the interface card (No. 1).

BFD removed timers from the active MPU.

BFD created the IPC message of the session.
debugging bfd drv

Use `debugging bfd drv` to enable BFD driver debugging.

Use `undo debugging bfd drv` to disable BFD driver debugging.

Syntax

```plaintext
debugging bfd drv
undo debugging bfd drv
```

Default

BFD driver debugging is disabled.

Views

User view

Default command level

1: Monitor level

Usage guidelines

Table 88 describes output fields and messages for the `debugging bfd drv` command.

### Table 88 Output from the debugging bfd drv command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gen DPMBuf failed for BFD VCPU, sess (session index)</td>
<td>The control VCPU failed to generate MBuf packets for the BFD virtual CPU (VCPU).</td>
</tr>
<tr>
<td>Switch MsgType failed for BFD VCPU, sess (session index), opertype (opertype), chgattr (chgattr)</td>
<td>The control VCPU failed to switch message type for the BFD VCPU.</td>
</tr>
<tr>
<td>Send Msg to BFD VCPU, msgtype (msgtype), sess (session index)</td>
<td>Sending messages to the BFD VCPU.</td>
</tr>
</tbody>
</table>
### Field

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Send Msg failed to BFD VCPU, ret (return value), msgtype (msgtype), sess (session index)</td>
<td>The control VCPU failed to send messages to the BFD VCPU.</td>
</tr>
<tr>
<td>Receive Msg on Ctrl VCPU, msgtype (msgtype), para1 (para1)</td>
<td>The control VCPU received messages on the control VCPU.</td>
</tr>
<tr>
<td>Cannot find session on BFD VCPU, msgtype (msgtype), sessindex (session index)</td>
<td>No corresponding session exists on the BFD VCPU.</td>
</tr>
<tr>
<td>Notify BFD Modul failed on Ctrl VCPU, ret (return value), event (event), sessindex (session index)</td>
<td>The control VCPU notified the BFD module of the failure.</td>
</tr>
<tr>
<td>Receive Msg on BFD VCPU, msgtype (msgtype), para1 (para1)</td>
<td>The BFD VCPU received messages from other VCPUs.</td>
</tr>
<tr>
<td>Modify Protocol: Sess not exist on BFD VCPU, msgtype (msgtype), sessindex (session index)</td>
<td>When modifying the protocol, no corresponding session exists on the BFD VCPU.</td>
</tr>
<tr>
<td>Clear Statistic Information: Sess not exist on BFD VCPU, msgtype (msgtype), sessindex (session index)</td>
<td>When clearing the statistics, no corresponding session exists on the BFD VCPU.</td>
</tr>
<tr>
<td>Sess already exist on BFD VCPU, sessindex (session index)</td>
<td>The session already exists on the BFD VCPU.</td>
</tr>
<tr>
<td>Total Sess MaxNum reached, can’t creat sess on BFD VCPU, sessindex (session index)</td>
<td>The number of sessions reached the maximum, so no further sessions can be created on the BFD VCPU.</td>
</tr>
<tr>
<td>Sessindex (session index) send pkt (ret: return value) when create session on BFD VCPU.</td>
<td>The first packet was sent after a session was created on the BFD VCPU.</td>
</tr>
<tr>
<td>Sessindex (session index) send pkt (ret: return value) when modify session on BFD VCPU.</td>
<td>The first packet was sent after a session was modified on the BFD VCPU.</td>
</tr>
<tr>
<td>Modify Packet: Sess not exist on BFD VCPU, sessindex (session index)</td>
<td>When modifying the packet, no corresponding session exists on the BFD VCPU.</td>
</tr>
<tr>
<td>Delete Session Entry: Sess not exist on BFD VCPU, sessindex (session index)</td>
<td>When deleting the session, no corresponding session exists on the BFD VCPU.</td>
</tr>
<tr>
<td>Notify First Packet: Sess not exist on BFD VCPU, sessindex (session index)</td>
<td>The packet receiving test succeeded, but no corresponding session exists on the BFD VCPU.</td>
</tr>
<tr>
<td>Pkt deliverup Msg send failed on BFD VCPU, ret (return value)</td>
<td>The BFD VCPU failed to deliver the message.</td>
</tr>
</tbody>
</table>

### Examples

- On a single-core device:

```
# Enable BFD driver debugging on a BFD-capable device. The output in this example was created when the following conditions exist:

- The IP address of VLAN-interface 2 is 100.1.2.2.
- The IP address of the peer interface connected to VLAN-interface 2 is 100.1.2.1.
- A BFD-enabled static route with a next hop of 100.1.2.2 has been configured on the peer device.

<Sysname> debugging bfd drv
<Sysname> system-view
[Sysname] ip route-static 2.2.3.1 255.255.255.255 vlan-interface 2 100.1.2.1 bfd control-packet
*Mar 22 18:38:20:154 2007 Sysname BFD/7/DRV: Chassis=1-Slot=5;Notify driver to add bfd session [100.1.2.2/100.1.2.1,Vlan2,Ctrl], Discr: 103
```

163
// BFD notified the driver on interface card No. 5 to add a BFD session with the following session information: source address 100.1.2.2, destination address 100.1.2.1, VLAN-interface 2, control packet mode, and local discriminator 103.

Mar 22 18:38:20:305 2007 Sysname BFD/7/DRV: Chassis=1-Slot=3; Notify driver to add bfd session [100.1.2.2/100.1.2.1,Vlan2,Ctrl], Discr: 103

// BFD notified the driver on interface card No. 3 to add a BFD session with the following session information: source address 100.1.2.2, destination address 100.1.2.1, VLAN-interface 2, control packet mode, and local discriminator 103.

Mar 22 18:38:20:465 2007 Sysname BFD/7/DRV: Chassis=1-Slot=3; Received BFD session first-detect-start message from driver, sess[100.1.2.2/100.1.2.1,Vlan2,Ctrl]

// BFD received a notification from the driver: Detection of session’s first packet starts, using session source address 100.1.2.2, destination address 100.1.2.1, VLAN-interface 2, and control packet mode.

Mar 22 18:38:20:645 2007 Sysname BFD/7/DRV: Chassis=1-Slot=3; Notify driver session [100.1.2.2/100.1.2.1,Vlan2,Ctrl] start, Discr: 103

// BFD notified the driver of the session detection start, using session source address 100.1.2.2, destination address 100.1.2.1, VLAN-interface 2, and control packet mode.

• On a multi-core device:

# Enable BFD driver debugging on a BFD-capable device. The output in this example was created when the following conditions exist:

• The IP address of GigabitEthernet 1/2 is 12.1.1.61.
• The IP address of the peer interface connected to GigabitEthernet 1/2 is 12.1.1.88.
• The device has established OSPF neighboring relationship with the peer device.
• The peer device has BFD enabled on its OSPF interface.

<Sysname> debugging bfd drv
<Sysname> system-view
[Sysname] interface gigabitethernet 1/2
[Sysname-GigabitEthernet1/2] ospf bfd enable

*Apr 28 06:34:08:939 2008 Sysname DPBFD/7/debug: Send Msg to BFD VCPU, msgtype 0xcf20005, sess 120

// Sending a message to the BFD VCPU. The message type is 0xcf20005 (indicating to create a BFD session), and the session index is 120.

*Apr 28 06:34:08:940 2008 Sysname DPBFD/7/debug: Receive Msg on BFD VCPU, msgtype 0xcf20005, paral 0x14ea9ab0

// BFD received the message on the BFD VCPU. The message type is 0xcf20005 (indicating to create a BFD session).

*Apr 28 06:34:08:940 2008 Sysname DPBFD/7/debug: Sessindex 120 send pkt(ret:0) when create session on BFD VCPU.

// The first packet was sent after a session with index 120 was created on the BFD VCPU. The return value is 0, indicating that the packet was sent.

*Apr 28 06:34:08:940 2008 Sysname DPBFD/7/debug: Receive Msg on Ctrl VCPU, msgtype 0xcf20003, paral 0x79

// The control VCPU received a message. The message type is 0xcf20003 (indicating that the first packet was received).
debugging bfd error

Use `debugging bfd error` to enable BFD error debugging.

Use `undo debugging bfd error` to disable BFD error debugging.

Syntax

```
debbuging bfd error
undo debugging bfd error
```

Default

BFD error debugging is disabled.

Views

User view

Default command level

1: Monitor level

Examples

# Enable BFD error debugging. The output in this example was created when the following conditions exist:

- BFD is enabled, and the device memory runs low.
- BFD parameters are configured on the interface that is not configured with BFD.

```
<Sysname> debugging bfd error
<Sysname> system-view
<Sysname> interface ethernet1/1
```
debugging bfd event

Use **debugging bfd event** to enable BFD event debugging.

Use **undo debugging bfd event** to disable BFD event debugging.

**Syntax**

```plaintext
debugging bfd event
undo debugging bfd event
```

**Default**

BFD event debugging is disabled.

**Views**

User view

**Default command level**

1: Monitor level

**Usage guidelines**

Table 89 describes output fields and messages for the **debugging bfd event** command.

**Table 89 Output from the debugging bfd event command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sync if: Failed to encode IPC message. Len(message length)</td>
<td>The active MPU failed to encapsulate the IPC message before sending interface structure.</td>
</tr>
<tr>
<td>Send interface message to (chassis chassis-id, slot slot-id)</td>
<td>The active MPU sent the IPC message containing interface structure to the newly inserted interface card (interface card number).</td>
</tr>
<tr>
<td>Sync if: Failed to encode last IPC message. Len(message length)</td>
<td>During interface information synchronization, the active MPU failed to encapsulate the last IPC message.</td>
</tr>
<tr>
<td>Send last interface message to (chassis chassis-id, slot slot-id)</td>
<td>The active MPU sent the last IPC message containing interface structure to the newly inserted interface card (interface card number).</td>
</tr>
<tr>
<td>Sync sess: Failed to encode last IPC message. Len(message length)</td>
<td>During session information synchronization, the active MPU failed to encapsulate the last IPC message.</td>
</tr>
<tr>
<td>Process sync if: Failed to create interface (Interface Index)</td>
<td>The newly inserted interface card failed to create the interface after receiving the IPC message with interface structure from the active MPU.</td>
</tr>
<tr>
<td>Process sync if: Error, interface(Interface Index) exist</td>
<td>When the newly inserted interface card used the IPC message with interface structure received from the active MPU to create the interface, an error occurred due to the existence of the interface.</td>
</tr>
</tbody>
</table>
Field                                                                 | Description                                                                 |
---                                                                 | ---                                                                 |
Process sync sess: Failed to create session (session index)          | The newly inserted interface card failed to create the session after receiving the IPC message with session structure from the active MPU. |
Process sync sess: Error, session already exist (session index)      | When the newly inserted interface card used the IPC message with session structure received from the active MPU to create the session, an error occurred due to the existence of the session. |
Receive (application message)                                         | BFD received an application message from the routing module. |
Unknown message type                                                  | Unknown message type. |

**Examples**

# Enable event debugging on a BFD-capable device. When you remove an interface configured with BFD parameters, output similar to the following example is generated:

```
<Sysname> debugging bfd event
<Sysname> system-view
[Sysname] undo interface dialer 0
*Dec 31 15:08:35:54 2006 Sysname BFD/7/EVENT:Receive a message of deleting Dialer0

// BFD received a notification that Dialer 0 was removed.
*Jan 4 09:39:02:598 2007 Sysname BFD/7/EVENT:Receive Create-sess, [Src:6.6.66.6, Dst:6.6.66.8, Dialer0, Echo], Direct, Instance:0x0, Proto: STATIC
*Jan 4 09:39:02:598 2007 Sysname BFD/7/EVENT:Receive Create-sess, [Src:10.1.1.1,Dst:10.1.1.2,Ethernet0/1/0,Ctrl], Direct, Instance:0x0, Proto: STATIC

// BFD received a session establishment request.
*Jan 4 09:40:41:914 2007 Sysname BFD/7/EVENT:Receive Delete-sess, [Src:6.6.66.6,Dst:6.6.66.8,Dialer0,Echo], Direct, Instance:0x0, Proto: STATIC
*Jan 4 09:40:41:914 2007 Sysname BFD/7/EVENT:Receive Delete-sess, [Src:10.1.1.1,Dst:10.1.1.2,Ethernet0/1/0,Ctrl], Direct, Instance:0x0, Proto: STATIC

// BFD received a session removal message.
```

disabling bfd fsm

Use `debugging bfd fsm` to enable BFD state machine debugging.

Use `undo debugging bfd fsm` to disable BFD state machine debugging.

**Syntax**

- `debugging bfd fsm`
- `undo debugging bfd fsm`

**Default**

BFD state machine debugging is disabled.

**Views**

- User view

**Default command level**

- Monitor level
## Usage guidelines

Table 90 describes output fields and messages for the `debugging bfd fsm` command.

### Table 90 Output from the debugging bfd fsm command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pkt Sta: AdminDown, Sess Sta: AdminDown, Oper: none</td>
<td>The incoming AdminDown packet was not processed when the session state was AdminDown.</td>
</tr>
<tr>
<td>Pkt Sta: Down, Sess Sta: AdminDown, Oper: None</td>
<td>The incoming down packet was not processed when the session state was AdminDown.</td>
</tr>
<tr>
<td>Pkt Sta: Init, Sess Sta: AdminDown, Oper: None</td>
<td>The incoming Init packet was not processed when the session state was AdminDown.</td>
</tr>
<tr>
<td>Pkt Sta: Up, Sess Sta: AdminDown, Oper: None</td>
<td>The incoming up packet was not processed when the session state was AdminDown.</td>
</tr>
<tr>
<td>Sess:(session information), Sta: INIT -&gt; DOWN, Diag: (diagnostic code of session)</td>
<td>After the AdminDown packet was received when the session status was Init, the session status changed to down.</td>
</tr>
<tr>
<td>Pkt Sta: Down, Sess Sta: Init, Oper: Keep session state and discard the packet</td>
<td>After the down packet was received when the session status was Init, the session kept its state and discarded the packet.</td>
</tr>
<tr>
<td>Sess:(session information), Sta: INIT -&gt; UP, Diag: (diagnostic code of session)</td>
<td>After the Init packet was received when the session status was Init, the session state changed to up.</td>
</tr>
<tr>
<td>Sess:(session information), Sta: UP -&gt; DOWN, Diag: (diagnostic code of session)</td>
<td>After the AdminDown packet was received when the session state was up, the session state changed to down.</td>
</tr>
<tr>
<td>Delete session</td>
<td>After the up session on the interface card received the AdminDown (or down) packet, the session was removed.</td>
</tr>
<tr>
<td>Failed to delete session</td>
<td>BFD failed to remove the up session after it received the AdminDown (or down) packet.</td>
</tr>
<tr>
<td>Reset session</td>
<td>After the up (or Init) session on the primary card of the centralized or distributed device received the AdminDown (or down) packet, the session was reset.</td>
</tr>
<tr>
<td>Failed to reset session</td>
<td>After the up (or Init) session on the primary card of the centralized or distributed device received the AdminDown (or down) packet, the session was reset but the operation failed.</td>
</tr>
</tbody>
</table>

### Examples

```bash
# Enable BFD state machine debugging.
<Sysname> debugging bfd fsm
*Dec 31 15:13:11:868 2006 Sysname BFD/7/FSM: Sess[10.1.1.1/10.1.1.2, Eth0/1/0, Ctrl], Sta: UP->DOWN, Diag: 1
// Session timed out. The debugging information diagnostic code of the output state change is 1, which indicates that the cause is session timeout.
```

168
debugging bfd ha

Use `debugging bfd ha` to enable BFD HA event debugging.
Use `undo debugging bfd ha` to disable BFD HA event debugging.

Syntax

debugging bfd ha
undo debugging bfd ha

Default

BFD HA event debugging is disabled.

Views

User view

Default command level

1: Monitor level

Usage guidelines

This command applies only to distributed devices.

Examples

# Enable BFD HA event debugging on a BFD-capable device.
<Sysname> debugging bfd ha

%Dec 31 16:20:33:772 2006 Sysname HA/4/LOG: Slave board in chassis 1 slot 7 changes to master
*Dec 31 16:20:35:240 2006 Sysname BFD/7/HA:Receive HA Smoothness Message
*Dec 31 16:20:35:255 2006 Sysname BFD/7/HA:Receive a message of starting smoothness. Smoothness begin...
*Dec 31 16:20:35:255 2006 Sysname BFD/7/HA:Delete aging timer for timing synchronization
*Dec 31 16:20:35:336 2006 Sysname BFD/7/HA:Receive application protocol OSPF/ISIS_BR_L1/ISIS_BR_L2/ISIS_P2P/STATIC GR over message
*Dec 31 16:20:36:522 2006 Sysname BFD/7/HA:Receive HA Smoothness Over Message
*Dec 31 16:20:36:522 2006 Sysname BFD/7/HA:Receive a message of smoothness aging. Start aging...
*Dec 31 16:20:36:522 2006 Sysname BFD/7/HA:Create HA aging timer
#Dec 31 16:20:36:586 2006 Sysname MIX/1/SLAVE SWITCH OVER:

// After the standby MPU (No.7) became the active MPU, it received an HA smoothness message. Then BFD started HA smoothness and deleted the synchronization aging timer. After the GR end message and HA smoothness message were received, a new HA aging timer was created.

debugging bfd ipc

Use `debugging bfd ipc` to enable BFD IPC event debugging.
Use `undo debugging bfd ipc` to disable BFD IPC event debugging.

Syntax

debugging bfd ipc
undo debugging bfd ipc
Default

BFD IPC event debugging is disabled.

Views

User view

Default command level

1: Monitor level

Usage guidelines

This command applies only to distributed devices.

Table 91 describes output fields and messages for the `debugging bfd ipc` command.

**Table 91 Output from the debugging bfd ipc command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[SND] Sess (session information), MAIN-&gt;IO, Mode: Unicast. Oper: Append Session, Dst chassis/slot: chassis-id/slot-id</td>
<td>BFD sent the IPC message (append session) to the interface card (or standby card) in unicast.</td>
</tr>
<tr>
<td>[SND] Sess (session information), MAIN-&gt;IO, Mode: Multicast, Oper: Append Session</td>
<td>BFD sent the IPC message (append session) to all the interface cards (or standby cards) in multicast.</td>
</tr>
<tr>
<td>[SND] Sess (session information), MAIN-&gt;IO, Mode: Unicast, Oper: Delete Session, Dst chassis/slot: chassis-id/slot-id</td>
<td>BFD sent the IPC message (delete session) to the interface card (or standby card) in unicast.</td>
</tr>
<tr>
<td>[SND] Sess (session information), MAIN-&gt;IO, Mode: Multicast, Oper: Delete Session</td>
<td>BFD sent the IPC message (delete session) to all the interface cards (or standby cards) in multicast.</td>
</tr>
<tr>
<td>[SND] Sess (session information), MAIN-&gt;IO, Mode: Unicast, Oper: Append App, Dst chassis/slot: chassis-id/slot-id</td>
<td>BFD sent the IPC message (append application) to the interface card (or standby card) in unicast.</td>
</tr>
<tr>
<td>[SND] Sess (session information), MAIN-&gt;IO, Mode: Multicast, Oper: Append App</td>
<td>BFD sent the IPC message (append application) to all the interface cards (or standby cards) in multicast.</td>
</tr>
<tr>
<td>[SND] Sess (session information), MAIN-&gt;IO, Mode: Unicast, Oper: Delete App, Dst chassis/slot: chassis-id/slot-id</td>
<td>BFD sent the IPC message (delete application) to the interface card (or standby card) in unicast.</td>
</tr>
<tr>
<td>[SND] Sess (session information), MAIN-&gt;IO, Mode: Multicast, Oper: Delete App</td>
<td>BFD sent the IPC message (delete application) to all the interface cards (or standby cards) in multicast.</td>
</tr>
<tr>
<td>[SND] Interface [interface name] MAIN-&gt;IO, Mode: Multicast, Oper: Delete Interface</td>
<td>BFD sent the IPC message (delete interface) to all the interface cards (or standby cards) in multicast.</td>
</tr>
<tr>
<td>[SND] MAIN-&gt;IO, Mode: Multicast, Oper: Is Sess Exist, IsExist: (flag of session exist)</td>
<td>BFD sent the session exist flag message to all the interface cards (or standby cards) in multicast.</td>
</tr>
<tr>
<td>[SND] MAIN-&gt;IO, Mode: Unicast, Oper: Is Sess Exist, IsExist: (flag of session exist)</td>
<td>BFD sent the session exist flag message to the newly inserted interface card (or standby card) in unicast.</td>
</tr>
<tr>
<td>[SND] MAIN-&gt;IO, Mode: Multicast, Oper: HA Notify</td>
<td>BFD notified the HA session message to all the cards.</td>
</tr>
<tr>
<td>[SND] Interface[interface name], MAIN-&gt;IO, Mode: Multicast, Oper: Synchronize TX</td>
<td>BFD synchronized the configured minimum transmit interval to all the cards.</td>
</tr>
<tr>
<td>[SND] Interface[interface name], MAIN-&gt;IO, Mode: Multicast, Oper: Synchronize RX</td>
<td>BFD synchronized the configured minimum transmit interval information to all the cards.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td>[SND] Interface[interface name], MAIN-&gt;IO, Mode: Multicast, Oper: Synchronize Detecting Multiplier</td>
<td>BFD synchronized the configured detecting multiplier to all the cards.</td>
</tr>
<tr>
<td>[SND] Interface[interface name], MAIN-&gt;IO, Mode: Multicast, Oper: Synchronize Auth Info</td>
<td>BFD synchronized the configured authentication information to all the cards.</td>
</tr>
<tr>
<td>[SND] MAIN-&gt;IO, Mode: Multicast, Oper: Synchronize Debugging Switch(debugging flag)</td>
<td>BFD synchronized the debugging switch information to all the cards.</td>
</tr>
<tr>
<td>[SND] MAIN-&gt;IO, Mode: Unicast, Oper: Synchronize Debugging Switch (debugging flag), Dst chassis/slot: chassis-id/slot-id</td>
<td>BFD synchronized the debugging switch information to the newly inserted card.</td>
</tr>
<tr>
<td>[SND] MAIN-&gt;IO, Mode: Multicast, Oper: Synchronize Trap Flag, Flag: (trap switch flag)</td>
<td>BFD synchronized the trap flag information to all the cards.</td>
</tr>
<tr>
<td>[SND] MAIN-&gt;IO, Mode: Unicast, Oper: Synchronize Trap Flag, Flag: (trap switch flag), Dst chassis/slot: chassis-id/slot-id</td>
<td>BFD synchronized the trap flag information to the newly inserted cards.</td>
</tr>
<tr>
<td>[SND] MAIN-&gt;IO, Mode: Multicast, Oper: Reset All Stats Info</td>
<td>BFD reset session statistics information on all the cards.</td>
</tr>
<tr>
<td>[SND] MAIN-&gt;IO, Mode: Multicast, Oper: Download Interface</td>
<td>BFD synchronized interface structure to all cards periodically.</td>
</tr>
<tr>
<td>[SND] MAIN-&gt;IO, Mode: Unicast, Oper: Download Session, Dst chassis/slot: chassis-id/slot-id</td>
<td>BFD synchronized session structure to the specified card periodically.</td>
</tr>
<tr>
<td>[SND] MAIN-&gt;IO, Mode: Multicast, Oper: Download Session</td>
<td>BFD synchronized session structure to all cards periodically.</td>
</tr>
<tr>
<td>[SND] MAIN-&gt;IO, Mode: Multicast, Oper: Synchronize IO Aging Timer Interval</td>
<td>BFD synchronized the interface card aging timer interval to all cards, including standby cards.</td>
</tr>
<tr>
<td>[SND] Sess (session information), IO-&gt;MAIN, Mode: Unicast, Oper: Session Update, Dst chassis/slot: chassis-id/slot-id</td>
<td>BFD sent the update message to the active MPU.</td>
</tr>
<tr>
<td>[RCV] Sess (session information), Oper: Append Session, Src chassis/slot: chassis-id/slot-id</td>
<td>BFD received an Append Session message.</td>
</tr>
</tbody>
</table>

**Examples**

# Enable IPC event debugging on a BFD-capable distributed device. When you configure BFD parameters on an interface, output similar to the following example is generated:

```
<Sysname> debugging bfd ipc
<Sysname> system-view
[Sysname] interface ethernet 7/1/0
[Sysname-Ethernet7/1/0] bfd min-receive-interval 500
*Dec 31 16:32:58:272 2006 Sysname BFD/7/IPC:[SND] Interface[Ethernet7/1/0], MAIN->IO, Mode: Multicast, Oper: Synchronize RX
*Dec 31 16:32:58:272 2006 Sysname BFD/7/IPC: Chassis=1-Slot=0;[RCV] Oper: Cfg change min-rx, Src chassis/slot: 1/7
// The parameter configuration was synchronized to other cards.
```

%Dec 31 16:31:02:304 2006 Sysname DEV/4/BOARD STATE NORMAL:
Board state changes to NORMAL in Frame 0 Chassis 1 Slot 1, type is Wvrp board.
System is busy with VIU configuration recovery, please wait a moment...
*Dec 31 16:31:05:336 2006 Sysname BFD/7/IPC:[SND] MAIN->IO, Mode: Unicast, Oper:
Synchronize Debugging Switch 32, Dst chassis/slot: 1/1
*Dec 31 16:31:05:360 2006 Sysname BFD/7/IPC: Chassis=1-Slot=0;[RCV] Oper: Interface
Download

//After a new interface card was inserted, BFD information was synchronized to the interface card
through IPC.

debugging bfd packet

Use debugging bfd packet to enable BFD packet debugging.
Use undo debugging bfd packet to disable BFD packet debugging.

Syntax

debugging bfd packet [ { receive | send } [ acl acl-number ] [ acl6 acl6-number ] ]
undo debugging bfd packet [ receive | send ]

Default

BFD packet debugging is disabled.

Views

User view

Default command level

1: Monitor level

Parameters

receive: Specifies debugging for received packets.
send: Specifies debugging for sent packets.
acl acl-number: Specifies an ACL by its number in the range of 2000 to 2999.
acl6 acl6-number: Specifies an IPv6 ACL by its number in the range of 2000 to 2999.

Examples

# Enable debugging for outgoing packets on a BFD-capable device, and reference ACL 2000 in the
command.
<Sysname> debugging bfd packet send acl 2000
<Sysname> terminal debugging
<Sysname> terminal monitor
*Jan  4 09:26:07:674 2007 Sysname BFD/7/PKT_SND:Echo packet,IP Head[Src: 5.5.55.5,
Dst:6.6.66.6], Ver:1, Sta:1, Reserved:0, Len:16, LD:2, Src:6.6.66.6, Dst:6.6.66.8
  // BFD sent an Echo packet.
*Jan  4 09:26:07:946 2007 Sysname BFD/7/PKT_RCV:Ctrl packet, Src:10.1.1.2, Dst:10.1.1.1,
Ver:1, Diag:0, Sta:3, P/F/C/A/D/R:0/0/1/0/0/0, multi:5, LD/RD:1/1, Tx:400ms, Rx:400ms
  // BFD received a control packet.

# Enable debugging for received packets on a BFD-capable device, and reference IPv6 ACL 2000 in the
command.
<Sysname> debugging bfd packet receive acl6 2000
debugging bfd scm

Use **debugging bfd scm** to enable BFD session control management debugging.

Use **undo debugging bfd scm** to disable BFD session control management debugging.

**Syntax**

```
debugging bfd scm
undo debugging bfd scm
```

**Default**

BFD session control management debugging is disabled.

**Views**

User view

**Default command level**

1: Monitor level

**Usage guidelines**

Table 92 describes output fields and messages for the **debugging bfd scm** command.

**Table 92** Output from the debugging bfd scm command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Remove from cache: {(application message)}</td>
<td>BFD removed the session messages not processed from the cache.</td>
</tr>
<tr>
<td>Bfd-if(interface name), Oper: create</td>
<td>BFD created an interface entry.</td>
</tr>
<tr>
<td>Bfd-if(interface name), Oper: delete</td>
<td>BFD deleted an interface entry.</td>
</tr>
<tr>
<td>Bfd-if(interface name), Oper: delete(Cleanup sessions)</td>
<td>BFD deleted an interface.</td>
</tr>
<tr>
<td>Clean up sessions in bfd-if(interface name)</td>
<td>BFD cleared session information on the interface.</td>
</tr>
<tr>
<td>Failed to update session control right</td>
<td>BFD failed to update session control right. This is normal in the SMOOTH phase.</td>
</tr>
</tbody>
</table>

**Examples**

# Enable BFD session control management debugging on a BFD-capable device.

<Sysname> debugging bfd scm
<Sysname> terminal debugging
<Sysname> terminal monitor

*Jan 4 09:30:09:642 2007 Sysname BFD/7/SCM:Sess[6.6.66.6/6.6.66.8,Dia0,Echo], Oper: Create, Application(STATIC)

// A BFD session was created: source address is 6.6.66.6, destination address is 6.6.66.8, interface is Dialer 0, working mode is Echo, and protocol type is static.
debugging bfd timer

Use **debugging bfd timer** to enable BFD timer event debugging.

Use **undo debugging bfd timer** to disable BFD timer event debugging.

**Syntax**

```
debugging bfd timer
undo debugging bfd timer
```

**Default**

BFD timer event debugging is disabled.

**Views**

User view

**Default command level**

1: Monitor level

**Examples**

# Enable BFD timer event debugging on a BFD-capable device.

```
<Sysname> debugging bfd timer
```

*Jan 4 09:34:24:54 2007 Sysname BFD/7/TMR:Create send-echo-packet timer

*Jan 4 09:34:24:54 2007 Sysname BFD/7/TMR:Send-echo-packet timer[1000ms] of session[6.6.66.6/6.6.66.8,Dia0,Echo] is created
*Jan 4 09:34:24:54 2007 Sysname BFD/7/TMR:Create send-packet timer
*Jan 4 09:34:24:54 2007 Sysname BFD/7/TMR:Send-packet timer[1000ms] of
  session[10.1.1.1/10.1.1.2,Eth1/1,Ctrl] is created
%Jan 4 09:34:24:66 2007 Sysname BFD/4/LOG: Sess [10.1.1.1/10.1.1.2, Eth1/1, Ctrl], Sta:
  DOWN->UP, Diag: 0
*Jan 4 09:34:24:66 2007 Sysname BFD/7/TMR:Reset send-packet timer
*Jan 4 09:34:24:66 2007 Sysname BFD/7/TMR:Send-packet timer[400ms] of
  session[10.1.1.1/10.1.1.2, Eth1/1,Ctrl] is reseted
*Jan 4 09:34:24:66 2007 Sysname BFD/7/TMR:Create detect-timer
  [10.1.1.1/10.1.1.2, Eth0/1/0,Ctrl] is created
*Jan 4 09:34:25:00 2007 Sysname LDP/4/SessionDown: Session(20.1.1.2:0. public Instance)'s
  state change to Down
*Jan 4 09:34:25:336 2007 Sysname LDP/4/SessionUp: Session(20.1.1.2:0. public Instance)'s
  state change to Up

//The Echo packet send timer and control packet send timer were created with a timeout of 1000 ms.
The session [10.1.1/10.1.1.2, Eth 1/1, Ctrl] state changed from down to up, and diagnostic code is 0 (no diagnostic information).
**BGP debugging commands**

The output description tables in this document only contain fields and messages that require an explanation.

**debugging bgp**

Use `debugging bgp` to enable debugging for specific BGP message types.

Use `undo debugging bgp` to disable debugging for specific BGP message types.

**Syntax**

```plaintext
debugging bgp [ ipv4-address | ipv6-address ] { keepalive | open | packet | raw-packet | route-refresh } [ receive | send ] [ verbose ]
undo debugging bgp [ ipv4-address | ipv6-address ] { keepalive | open | packet | raw-packet | route-refresh } [ receive | send ] [ verbose ]
```

**Default**

Debugging for specific BGP message types is disabled.

**Views**

User view

**Default command level**

1: Monitor level

**Parameters**

- `ipv4-address`: Specifies the IPv4 address of a peer.
- `ipv6-address`: Specifies the IPv6 address of a peer.
- `keepalive`: Enables BGP Keepalive message debugging.
- `open`: Enables BGP OPEN message debugging.
- `packet`: Specifies to enable BGP packet debugging.
- `route-refresh`: Enables BGP Route-Refresh message debugging.
- `receive`: Enables the debugging for received BGP packets.
- `send`: Enables the debugging for sent BGP packets.
- `verbose`: Displays detailed debugging information.

**Usage guidelines**

Enabling debugging might decrease system performance. Therefore, use this command only when necessary, and disable debugging after the debugging operation is complete.

Table 93 describes output fields and messages for the `debugging bgp` command.
Table 93 Output from the debugging bgp command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Recv</td>
<td>Received packets.</td>
</tr>
<tr>
<td>Send</td>
<td>Sent packets.</td>
</tr>
<tr>
<td>Length: LengthNumber</td>
<td>Packet length.</td>
</tr>
</tbody>
</table>

Table 94 describes output fields and messages for the **debugging bgp open** command.

Table 94 Output from the debugging bgp open command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version: X</td>
<td>BGP protocol version.</td>
</tr>
<tr>
<td>Local AS: X</td>
<td>Local AS number.</td>
</tr>
<tr>
<td>OPT Type: 2 (Capability)</td>
<td>Capability negotiation.</td>
</tr>
<tr>
<td>CAP Type: 1 (Multiprotocol)</td>
<td>Multiprotocol capability: afi: 1; safi: 1.</td>
</tr>
<tr>
<td>IPv4-UNC (1/1)</td>
<td>Capable of route-refresh.</td>
</tr>
<tr>
<td>CAP Type: 2 (RouteRefresh)</td>
<td>Capable of filtering outbound routes. The capability field length is 7.</td>
</tr>
<tr>
<td>CAP Type: 3 (RouteFilter)</td>
<td>Type: 64 refers to the type value of the ORF capability; both refers to the ORF sending (send) and receiving (receive) capabilities.</td>
</tr>
<tr>
<td>Total CAPB Len: X</td>
<td>Total capability length.</td>
</tr>
<tr>
<td>Total OPT Len: X</td>
<td>Total optional parameter length.</td>
</tr>
</tbody>
</table>

Table 95 describes output fields and messages for the **debugging bgp refresh** command.

Table 95 Output from the debugging bgp refresh command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AFI: 1; SAFI: 1</td>
<td>• Address family: 1.</td>
</tr>
<tr>
<td></td>
<td>• Sub-address family: 1.</td>
</tr>
<tr>
<td>AFI/ SAFI</td>
<td>Address family/sub-address family.</td>
</tr>
<tr>
<td>WTR</td>
<td>(When to Refresh) Wait delay for sending refresh messages.</td>
</tr>
<tr>
<td>ORF Type</td>
<td>Type value of the ORF capability.</td>
</tr>
<tr>
<td>Length of ORFs</td>
<td>Total length of the ORF entries.</td>
</tr>
<tr>
<td>Action</td>
<td>Action field of the ORF entry.</td>
</tr>
<tr>
<td>Match</td>
<td>Match field of the ORF entry.</td>
</tr>
</tbody>
</table>

For more information about Update message debugging information, see the following section.
Examples

# Enable debugging for all BGP messages. When a BGP peer relationship is being established, output similar to the following example is generated:

<Sysname> debugging bgp
<Sysname> system-view
[Sysname] bgp 100
[Sysname-bgp] undo synchronization
[Sysname-bgp] peer 11.1.1.2 group 200

*0.2683484 4945 RM/6/RMDEBUG:
   BGP.: 11.1.1.2 Send OPEN, Version: 4
   Local AS: 100, HoldTime: 180, Router ID: 192.168.74.1

   OPT Type:   2 (Capability)  
   CAP Type:   1 (Multiprotocol)  CAP Len: 4
   IPv4-UNC (1/1)
   CAP Type:   2 (RouteRefresh)  CAP Len: 0

   Total CAPB Len : 8
   Total OPT Len  : 10
   Total Message Len : 39

// Information about the sent BGP Open message was displayed.

*0.2683500 4945 RM/6/RMDEBUG:
   BGP.: 11.1.1.2 Recv OPEN Length: 39
   Version: 4, Local AS: 200, HoldTime : 180,
   BGP ID: 192.168.74.2, TotOptLen: 10

   OPT Type:   2 (Capability)  OPT Len: 8
   CAP Type:   1 (Multiprotocol)  CAP Len: 4
   IPv4-UNC (1/1)
   CAP Type:   2 (RouteRefresh)  CAP Len: 0

// Information about the received BGP Open message was displayed.

*0.2683500 4945 RM/6/RMDEBUG:
   BGP.: Send KEEPALIVE
   Length 19

// Information about the sent BGP Keepalive message was displayed.

*0.2683609 4945 RM/6/RMDEBUG:
   BGP.: 11.1.1.2 Recv KEEPALIVE
   Length: 19

// Information about the received BGP Keepalive message was displayed.

[Sysname-bgp] import-route static
*0.2710031 4945 RM/6/RMDEBUG:
   BGP.: Send UPDATE to 11.1.1.2 for following destinations :
   Origin       : Incomplete
   AS Path      : 100
   Next Hop     : 11.1.1.1
   MED          : 0
111.1.1.1/32,

// Information about the sent BGP update of a redistributed static route was displayed.
*0.2728250 4945 RM/6/RMDEBUG:
  BGP.: Recv UPDATE from 11.1.1.2 with following destinations :
  Update message length : 53
  MED          : 0
  Origin       : Incomplete
  AS Path      : 200
  Next Hop     : 11.1.1.2
222.1.1.1/32

// Information about the received BGP Update of a redistributed static route was displayed.
The output in the following examples was created after the BGP connection between Device A and
Device B reaches Established state.

# On Device B, enable debugging for the sent BGP route-refresh messages. Execute the refresh bgp all
import command.
<void>
</void>

// Device B sent a BGP Route-refresh message, with a length of 23, address family of 1, and sub-address family of 1.

# On device A, enable debugging for received BGP route-refresh messages.
<void>
</void>

// Device A received the BGP Route-refresh message, with a length of 23, address family of 1, sub-address family of 1, and When to Refresh of 0.

debugging bgp all

Use debugging bgp all to enable all BGP debugging.

Use undo debugging bgp all to disable all BGP debugging.

Syntax

d debugging bgp [ ipv4-address | ipv6-address ] all
undo debugging bgp [ ipv4-address | ipv6-address] all

Default
All BGP debugging is disabled.

Views
User view

Default command level
1: Monitor level
Parameters

- **ipv4-address**: Specifies the IPv4 address of a peer.
- **ipv6-address**: Specifies the IPv6 address of a peer.

Usage guidelines

This command enables all BGP debugging, which might cause heavy traffic and decrease system performance. Therefore, use this command only when necessary.

Disable debugging after the debugging operation is complete.

Examples

```
# Display BGP configuration on Device A and Device B.

[Sysname-bgp] display this
#
bgp 100
undo synchronization
peer 10.1.1.2 as-number 100
#

[Sysname-bgp] display this
#
bgp 100
undo synchronization
peer 10.1.1.1 as-number 100
#

# Enable all BGP debugging on Device A. While Device A and Device B are establishing an IBGP peer relationship with each other, output similar to the following example is generated:

<Sysname> debugging bgp all
  BGP_TIMER: CR Timer Expired for Peer 10.1.1.2
  BGP.: 10.1.1.2 Current event is CRTimerExpired.
  BGP.: 10.1.1.2 Current event is Start.
*Aug 24 15:32:02:630 2006 Sysname RM/6/RMDEBUG:
  BGP.: 10.1.1.2 State is changed from IDLE to CONNECT.

// Upon expiration of the connect timer, BGP initiated a connection with the peer.

*Aug 24 15:31:55:331 2006 Sysname RM/6/RMDEBUG:
  BGP.: 10.1.1.2 Current event is TransConnOpenFailed.
*Aug 24 15:31:55:331 2006 Sysname RM/6/RMDEBUG:
  BGP.: 10.1.1.2 State is changed from CONNECT to ACTIVE.
*Aug 24 15:32:02:630 2006 Sysname RM/6/RMDEBUG:
  BGP.: 10.1.1.2 Send OPEN, Version: 4
    Local AS: 100, HoldTime: 180, Router ID: 10.1.1.1
    OPT Type: 2 (Capability)
    CAP Type: 1 (Multiprotocol) CAP Len: 4
                  IPv4-UNC (1/1)
    CAP Type: 2 (RouteRefresh) CAP Len: 0
```

180
After the connection is established, BGP sent an OPEN message to the peer.

BGP.: 10.1.1.2 State is changed from ACTIVE to OPENSENT.

After the connection is established, BGP sent an OPEN message to the peer.

BGP.: 10.1.1.2 Recv OPEN Length: 39
  Version: 4, Local AS: 100, HoldTime: 180,
  BGP ID: 10.1.1.2, TotOptLen: 10

  OPT Type: 2 (Capability) OPT Len: 8
  CAP Type: 1 (Multiprotocol) CAP Len: 4
    IPv4-UNC (1/1)
  CAP Type: 2 (RouteRefresh) CAP Len: 0

BGP received an OPEN message from the peer.

BGP.: 10.1.1.2 Current event is ReceiveOpenMessage.

BGP has established the peer relationship successfully.

NOTE:
If two devices cannot establish a BGP peer relationship, you can compare the example debugging information with the actual debugging information to locate the fault. For detailed descriptions of the specific messages, see the following commands.

debugging bgp bfd
Use debugging bgp bfd to enable BGP BFD debugging.
Use undo debugging bgp bfd to disable BGP BFD debugging.

Syntax

debugging bgp bfd [ [ vpn-instance vpn-instance-name ] { ipv4-address | ipv6-address } ]
undo debugging bgp bfd [ [ vpn-instance vpn-instance-name ] [ ipv4-address | ipv6-address ] ]

Default

BGP BFD debugging is disabled.

Views

User view

Default command level

1: Monitor level

Parameters

vpn-instance vpn-instance-name: Specifies an MPLS L3VPN instance. The vpn-instance-name argument is a case-sensitive string of 1 to 31 characters. If no VPN instance is specified, BGP BFD debugging is enabled for the public network. Support for this option depends on the device model.

vpn-instance vpn-instance-name: Specifies a VPN instance name, a string of 1 to 31 characters.

ipv4-address: Specifies the peer IPv4 address.

ipv6-address: Specifies the peer IPv6 address.

Usage guidelines

If no VPN instance name or IP address is specified, all BGP BFD debugging information is displayed.

Table 96 describes output fields and messages for the debugging bgp bfd command for IPv4 environment.

Table 96 Output from the debugging bgp bfd command (for IPv4 environment)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| BGP_BFD: Send msg-type msg to BFD, Connection type con-type, Src IP src-ip, Dst IP dst-ip, Instance ID instance-id. | A BFD message was sent to BFD. The message contains message type, link type, source address and destination address of the session, and BGP instance ID (used to identify unicast BGP, VPN BGP, or VF BGP).  
  - msg-type—Message type: CREATE, DELETE or DISABLE.  
  - con-type—Link type: DIRECT or INDIRECT.  
  - src-ip—Source address of the session.  
  - dst-ip—Destination address of the session.  
  - instance-id—BGP instance ID. |
| BGP_BFD: Recv BFD DOWN msg, Src IP src-ip, Dst IP dst-ip, Instance ID instance-id. | A DOWN BFD session message was received, with source address src-ip, destination address dst-ip, and BGP instance ID instance-id. |
| BGP_BFD: Recv BFD NOT DOWN msg, Src IP src-ip, Dst IP dst-ip, Instance ID instance-id. | A NOT DOWN BFD session message was received, with source address src-ip, destination address dst-ip, and BGP instance ID instance-id. |
| BGP_BFD: BGP session dst-ip not establish when recv BFD DOWN msg. | A DOWN BFD session message was received, but the BGP session with destination address dst-ip had not been established. |
| BGP_BFD: BGP session dst-ip not exist when recv BFD DOWN msg. | A DOWN BFD session message was received, but the BGP session with destination address dst-ip was not in existence. |

182
Table 97 describes output fields and messages for the **debugging bgp bfd** command for IPv6 environment.

### Table 97 Output from the debugging bgp bfd command (for IPv6 environment)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Send XXX msg to BFD</td>
<td>The XXX can be one of the following:</td>
</tr>
<tr>
<td></td>
<td>• CREATE.</td>
</tr>
<tr>
<td></td>
<td>• DELETE.</td>
</tr>
<tr>
<td></td>
<td>• Disable.</td>
</tr>
<tr>
<td>Connection type XXX</td>
<td>Connection type: <strong>DIRECT</strong> or <strong>INDIRECT</strong>.</td>
</tr>
<tr>
<td>Src IP</td>
<td>Source IP address.</td>
</tr>
<tr>
<td>Dst IP</td>
<td>Destination IP address.</td>
</tr>
</tbody>
</table>

**Examples**

# Enable BGP BFD debugging on device 10.1.1.2. Output similar to the following example is generated when the following tasks are performed:

- Establishes a BGP peer relationship between device 10.1.1.2 and device 10.1.1.
- Enables BFD on both devices.

```plaintext
<Sysname> debugging bgp bfd
// (A BGP session was created.) A CREATE BFD session message was sent to BFD, with link type DIRECT, source address 10.1.1.2, destination address 10.1.1.1, and BGP instance ID 0 (indicating unicast BGP).
```

# Enable BGP BFD debugging on device 15::2. Output similar to the following example is generated when the following tasks are performed:

- Establishes an IPv6 BGP peer relationship between device 15::2 and device 15::1.
- Enables BFD on both devices.

```plaintext
<Sysname> debugging bgp bfd 15::1 [Sysname-bgp-af-ipv6] peer 15::1 bfd
*Aug 12 21:10:00:297 2009 Sysname RM/6/RMDEBUG: BGP_BFD: Send CREATE msg to BFD, Connection type DIRECT, Src IP 15::2, Dst IP 15::1, Instance ID 0.
// BGP sent a CREATE message to BFD, with connection type DIRECT, source address 15::2, destination address 15::1, and instance ID 0.
```

**debugging bgp detail**

Use **debugging bgp detail** to enable detailed debugging for BGP.

Use **undo debugging bgp detail** to disable detailed debugging for BGP.
Syntax

debugging bgp detail
undo debugging bgp detail

Default

Detailed debugging for BGP is disabled.

Views

User view

Default command level

1: Monitor level

Usage guidelines

Enabling debugging might decrease system performance. Therefore, use this command only when necessary. Disable debugging after the debugging operation is complete.

Table 98 describes output fields and messages for the debugging bgp detail command.

Table 98 Output from the debugging bgp detail command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BGP.xxx</td>
<td>BGP instance name.</td>
</tr>
<tr>
<td>BGP can’t get physical interface index for destination X.X.X.X</td>
<td>BGP cannot get the physical interface index for destination X.X.X.X.</td>
</tr>
<tr>
<td>Fail to search Token for destination X.X.X.X</td>
<td>Token search for destination X.X.X.X failed.</td>
</tr>
<tr>
<td>Allocate token failed. destination= X.X.X.X LspmErr=X</td>
<td>Failed to allocate Token: destination X.X.X.X, LSP error code X.</td>
</tr>
<tr>
<td>There are XX routes allocate token failed at this time before</td>
<td>BGP failed to allocate Tokens for XX routes.</td>
</tr>
<tr>
<td>Recv UPDATE with following RD XXX</td>
<td>BGP received an Update message with RD XXX.</td>
</tr>
<tr>
<td>Received New VPNv4 Route with Ext-Community NULL Ignoring Route</td>
<td>BGP received a new VPNv4 route with Ext-Community as NULL and therefore ignored the route.</td>
</tr>
<tr>
<td>Received VPNv4 Packet But No Such Route In Loc Rib</td>
<td>BGP received a VPNv4 packet, and there is no such a route in the local routing table.</td>
</tr>
<tr>
<td>operation XX</td>
<td>Label operation code.</td>
</tr>
<tr>
<td>Receive SMB UP backup requirement</td>
<td>BGP received a standby board startup event requesting an active/standby switchover.</td>
</tr>
<tr>
<td>Receive routine backup requirement</td>
<td>BGP received a routine request for an active/standby switchover.</td>
</tr>
<tr>
<td>Receive SMB down backup requirement</td>
<td>BGP received a standby board down event requesting an active/standby switchover.</td>
</tr>
</tbody>
</table>

Examples

# Enable detailed debugging for BGP. When a BGP peer relationship is being established, output similar to the following example is generated:

<Sysname> debugging bgp detail
debugging bgp event

Use `debugging bgp event` to enable event debugging for BGP.

Use `undo debugging bgp event` to disable event debugging for BGP.

**Syntax**

```
debugging bgp [ ipv4-address | ipv6-address ] event
undo debugging bgp [ ipv4-address | ipv6-address ] event
```

**Default**

BGP event debugging is disabled.

**Views**

User view

**Default command level**

1: Monitor level

**Parameters**

- `ipv4-address`: Specifies the IPv4 address of a peer.
- `ipv6-address`: Specifies the IPv6 address of a peer.

**Usage guidelines**

After this debugging is enabled, information about all the BGP state machine transitions and the events triggering these transitions will be displayed. If no BGP peer relationship can be established, you can enable this debugging to locate the fault. Enabling debugging might decrease system performance. Therefore, use this command only when necessary, and disable debugging after the debugging operation is complete.

Table 99 describes output fields and messages for the `debugging bgp event` command.

**Table 99 Output from the debugging bgp event command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Current event is Eventname</td>
<td>Name of the current event.</td>
</tr>
</tbody>
</table>

**Examples**

```
# Enable BGP event debugging. When a BGP peer relationship is being established, output similar to the following example is generated:
<Sysname> debugging bgp event
*Aug 24 14:12:13:674 2006 Sysname RM/6/RMDEBUG:
  BGP.: 10.1.1.2 Current event is CRTimerExpired.
```
debugging bgp graceful-restart

Use debugging bgp graceful-restart to enable GR event debugging for BGP.
Use undo debugging bgp graceful-restart to disable GR event debugging for BGP.

Syntax

documentation bgp [ ipv4-address | ipv6-address ] graceful-restart
undo debugging bgp [ ipv4-address | ipv6-address ] graceful-restart

Default
GR event debugging for BGP is disabled.

Views
User view

Default command level
1: Monitor level

Parameters
ipv4-address: Specifies the IPv4 address of a peer.
ipv6-address: Specifies the IPv6 address of a peer.

Usage guidelines
After this debugging is enabled, the debugging information for the BGP GR operation will be displayed, including the currently triggered event, state transitions, receiving/sending of EOR messages, and peer
IP address. If a fault occurs to the GR operation of BGP, you can display the debugging information for fault location.

Enabling debugging might decrease system performance. Therefore, use this command only when necessary, and disable debugging after the debugging operation is complete.

**Examples**

# Enable BGP GR event debugging on the BGP GR helper. When the neighbor performs an active/standby switchover, output similar to the following example is generated:

```
<Sysname> debugging bgp graceful-restart
%Aug 24 17:05:08:770 2006 Sysname RM/3/RMLOG:
  BGP.: 10.1.1.1 State is changed from ESTABLISHED to IDLE.
*Aug 24 17:05:08:786 2006 Sysname RM/6/RMDEBUG:
  BGP_GR: Peer 10.1.1.1 Gracefully Restarting
  // The BGP neighbor 10.1.1.1 was performing GR.
%Aug 24 17:05:40:851 2006 Sysname RM/3/RMLOG:
  BGP.: 10.1.1.1 State is changed from OPENCONFIRM to ESTABLISHED.
*Aug 24 17:05:40:867 2006 Sysname RM/6/RMDEBUG:
  BGP_GR: Sent EOR to Peer 10.1.1.1 (IPv4-UNC)
  // The BGP GR helper finished sending initial routes and then sent an EOR message to peer 10.1.1.1.
*Aug 24 17:05:41:96 2006 Sysname RM/6/RMDEBUG:
  BGP_GR: Received EOR from Peer 10.1.1.1 (IPv4-UNC)
  // The BGP GR helper received an EOR message from peer 10.1.1.1.
```

debugging bgp non-stop-routing

Use **debugging bgp non-stop-routing** to enable BGP NSR debugging.

Use **undo debugging bgp non-stop-routing** to disable BGP NSR debugging.

**Syntax**

```
debugging bgp non-stop-routing
undo debugging bgp non-stop-routing
```

**Default**

BGP NSR debugging is disabled.

**Views**

User view

**Default command level**

1: Monitor level

**Examples**

# Enable BGP NSR debugging. When BGP NSR is enabled, output similar to the following example is generated:

```
<Sysname> debugging bgp non-stop-routing
*0.92311078 Sysname RM/6/RMDEBUG:
  BGP NSR: The instance isn't found in the synchronization message.
```
BGP NSR: Prefix 1.1.1.1 Mask 32 is found in the synchronization message.

BGP NSR: Batch backup of NSR data starts.

BGP NSR: Batch backup of Neighbor routes received from neighbors starts.

BGP NSR: Batch backup of Neighbor routes is complete.

BGP NSR: Packet with length 50 is expected from socket 1, but the received packet length is 50.

BGP NSR: Timestamp is wrong in the ACK packet.

BGP NSR: Session isn't found in ACK packet.

BGP NSR: NSR smooth starts.

debbuging bgp timer

Use `debugging bgp timer` to enable BGP timer debugging.

Use `undo debugging bgp timer` to disable BGP timer debugging.

**Syntax**

```
debugging bgp [ ipv4-address | ipv6-address ] timer
undo debugging bgp [ ipv4-address | ipv6-address ] timer
```

**Default**

BGP timer debugging is disabled.

**Views**

User view

**Default command level**

1: Monitor level

**Parameters**

- `ipv4-address`: Specifies the IPv4 address of a peer.
- `ipv6-address`: Specifies the IPv6 address of a peer.
Usage guidelines

Enabling debugging might decrease system performance. Therefore, use this command only when necessary, and disable debugging after the debugging operation is complete.

Table 100 describes output fields and messages for the `debugging bgp timer` command.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CR Timer</td>
<td>Connection retry timer.</td>
</tr>
<tr>
<td>HOLD Timer</td>
<td>Hold timer.</td>
</tr>
</tbody>
</table>

Examples

# Enable BGP timer debugging on the device. Output similar to the following example is generated when the following conditions exist:

- The device specifies the neighbor device 2.2.2.2 as a BGP peer.
- The neighbor device does not specify the device as the BGP peer.

```
<Sysname> debugging bgp timer
*0.92311078 Sysname RM/6/RMDEBUG:
  BGP_TIMER: CR Timer Expired for Peer 2.2.2.2
```

// The ConnectRetry timer for BGP peer 2.2.2.2 expired.

# The device has successfully established a BGP peer relationship with a neighbor device. Enable BGP timer debugging on the device. When the device cannot receive keepalive messages from the neighbor device, output similar to the following example is generated:

```
<Sysname> debugging bgp timer
*0.92311078 Sysname RM/6/RMDEBUG:
  BGP_TIMER: HOLD Timer Expired for Peer 2.2.2.2
```

// The hold timer expired on BGP peer 2.2.2.2.

debugging bgp update

Use `debugging bgp update` to enable debugging for BGP Update messages.

Use `undo debugging bgp update` to disable debugging for BGP Update messages.

Syntax

```
debugging bgp update [ acl acl-number | ip-prefix ip-prefix-name ] [ receive | send ] [ verbose ]
undo debugging bgp update [ acl acl-number | ip-prefix ip-prefix-name ] [ receive | send ] [ verbose ]
```

Default

Debugging for BGP Update messages is disabled.

Views

User view

Default command level

1: Monitor level
Parameters

acl acl-number: Specifies an ACL in the range of 2000 to 3999 for filtering packet debugging information.

ip-prefix ip-prefix-name: Specifies an IP prefix list for filtering the packet debugging information. ip-prefix-name is a string of 1 to 19 characters.

receive: Enables the debugging for received BGP packets.

send: Enables the debugging for sent BGP packets.

verbose: Displays detailed debugging information.

Usage guidelines

Enabling debugging might decrease system performance. Therefore, use this command only when necessary, and disable debugging after the debugging operation is complete.

Table 101 describes output fields and messages for the debugging bgp update command.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BGP.xxx</td>
<td>BGP instance name.</td>
</tr>
<tr>
<td>Err/SubEr</td>
<td>Error code/error subcode.</td>
</tr>
<tr>
<td>Errdata:</td>
<td>Error data.</td>
</tr>
<tr>
<td>x.x.x.x/xx</td>
<td>Destination address/mask.</td>
</tr>
<tr>
<td>RPM policy failed for policy: name:xx</td>
<td>Failed to pass policy xx.</td>
</tr>
<tr>
<td>INBOUND</td>
<td>Inbound policy.</td>
</tr>
<tr>
<td>tnl id</td>
<td>Tunnel ID.</td>
</tr>
<tr>
<td>afi = 196(l2vpn) safi = 128(l2vpn)</td>
<td>Address family is 196. Sub-address family is 128 (L2VPN).</td>
</tr>
<tr>
<td>afi = 155(vpls) safi = 128(vpls)</td>
<td>Address family is 155. Sub-address family is 128 (VPLS).</td>
</tr>
</tbody>
</table>

Examples

See the Update message debugging information for specific parameters.

debugging bgp update ipv4

Use debugging bgp update ipv4 to enable debugging for IPv4 BGP Update messages.

Use undo debugging bgp update ipv4 to disable debugging for IPv4 BGP Update messages.

Syntax

debugging bgp update ipv4 [ peer { ipv4-address | ipv4-group-name } | ip-prefix ip-prefix-name ] [ receive | send ] [ verbose ]

undo debugging bgp update ipv4 [ peer { ipv4-address | ipv4-group-name } | ip-prefix ip-prefix-name ] [ receive | send ] [ verbose ]

Default

Debugging for IPv4 BGP Update messages is disabled.
Views

User view

Default command level

1: Monitor level

Parameters

peer { ipv4-address | ipv4-group-name }: Specifies BGP IPv4 update debugging for the specified IPv4 peer/peer group.

ip-prefix ip-prefix-name: Filters the output message debugging information with the specified IP prefix list.

receive: Specifies debugging for received BGP updates.

send: Specifies debugging for sent BGP updates.

verbose: Displays detailed debugging information.

Usage guidelines

Enabling debugging might decrease system performance. Therefore, use this command only when necessary, and disable debugging after the debugging operation is complete.

Examples

# Device A and Device B have established an IPv4 peer relationship. Enable debugging for IPv4 BGP Update messages on either device. Output similar to the following example is generated:

<Sysname> debugging bgp update ipv4

*Sep 4 16:08:20:458 2006 Sysname RM/6/RMDEBUG:
BGP.: Send UPDATE to 11.1.1.2 for following destinations :
Origin : Incomplete
AS Path :
Next Hop : 11.1.1.1
Local Pref : 100
MED : 0
1.1.1.1/32,

*Sep 4 16:08:20:473 2006 Sysname RM/6/RMDEBUG:
BGP.: Send UPDATE to 11.1.1.2 for following destinations :
Origin : Incomplete
AS Path :
Next Hop : 11.1.1.1
Local Pref : 100
MED : 0
11.1.1.0/24,

*Sep 4 16:08:20:473 2006 Sysname RM/6/RMDEBUG:
BGP.: Send UPDATE to 11.1.1.2 for following destinations :
Origin : Incomplete
AS Path :
Next Hop : 11.1.1.1
Local Pref : 100
MED : 0
11.1.1.2/32,
debugging bgp update ipv6

Use debugging bgp update ipv6 to enable debugging for IPv6 BGP Update messages.

Use undo debugging bgp update ipv6 to disable debugging for IPv6 BGP Update messages.

Syntax

d debugging bgp update ipv6 [ peer { ipv6-address | ipv6-group-name } | ipv6-prefix ipv6-prefix-name ]
[ receive | send ] [ verbose ]

undo debugging bgp update ipv6 [ peer { ipv6-address | ipv6-group-name } | ipv6-prefix ipv6-prefix-name ]
[ receive | send ] [ verbose ]

Default

Debugging for IPv6 BGP Update messages is disabled.

Views

User view
Default command level

1: Monitor level

Parameters

peer { ipv6-address | ipv6-group-name }: Specifies BGP IPv6 update debugging for the specified IPv6 peer/peer group.

ipv6-prefix ipv6-prefix-name: Filters the output message debugging information with the specified IP prefix list.

receive: Specifies debugging for received BGP packets.

send: Specifies debugging for sent BGP packets.

verbose: Displays detailed debugging information.

Usage guidelines

Enabling debugging might decrease system performance. Therefore, use this command only when necessary, and disable debugging after the debugging operation is complete.

Examples

# Device A and Device B have established an IPv6 BGP peer relationship. Enable debugging for IPv6 BGP Update messages on either device. Output similar to the following example is generated:

```bash
<Sysname> debugging bgp update ipv6

%Sep 4 16:17:20:642 2006 Sysname RM/3/RMLOG:
  BGP.: 100::2 State is changed from OPENCONFIRM to ESTABLISHED.

*Sep 4 16:17:20:642 2006 Sysname RM/6/RMDEBUG:
  BGP.: Send UPDATE to 100::2 for following destinations :
  Origin : Incomplete
  AS Path :
  Next Hop : 100::1
  Local Pref : 100
  MED : 0
  11::11/128,

*Sep 4 16:17:20:642 2006 Sysname RM/6/RMDEBUG:
  BGP.: Send UPDATE to 100::2 for following destinations :
  Origin : Incomplete
  AS Path :
  Next Hop : 100::1
  Local Pref : 100
  MED : 0
  100::/64,

*Sep 4 16:17:20:708 2006 Sysname RM/6/RMDEBUG:
  BGP_IPV6.: Recv UPDATE from peer 100::2 with following destinations :

  Update message length : 86
  Local Pref : 100
  MED : 0
```

193
debugging bgp update ipv6 vpn-instance

Use **debugging bgp update ipv6 vpn-instance** to enable debugging for IPv6 BGP Update messages of the specified VPN instance.

Use the **undo debugging bgp update ipv6 instance** command to disable debugging for IPv6 BGP Update messages of the specified VPN instance.

**Syntax**

```
debugging bgp update ipv6 vpn-instance vpn-instance-name [ ipv6-prefix ipv6-prefix-name | peer ipv6-address ] [ receive | send ] [ verbose ]
undo debugging bgp ipv6 update vpn-instance vpn-instance-name [ ipv6-prefix ipv6-prefix-name | peer ipv6-address ] [ receive | send ] [ verbose ]
```

**Default**

Debugging for IPv6 BGP Update messages of the specified VPN instance is disabled.

**Views**

User view

**Default command level**

1: Monitor level

**Parameters**

- `vpn-instance-name`: Specifies an MPLS L3VPN instance. The `vpn-instance-name` argument is a case-sensitive string of 1 to 31 characters.
- `ipv6-prefix ipv6-prefix-name`: Filters the output message debugging information with the specified IP prefix list.
- `peer ipv6-address`: Specifies BGP IPv6 update debugging for the specified IPv6 peer/peer group.
- `receive`: Specifies debugging for received BGP packets.
- `send`: Specifies debugging for sent BGP packets.
- `verbose`: Displays detailed debugging information.
Usage guidelines

See the previous command for the description of the Update message information.

debugging bgp update l2vpn

Use debugging bgp update l2vpn to enable debugging for BGP L2VPN Update messages.
Use undo debugging bgp update l2vpn to disable debugging for BGP L2VPN Update messages.

Syntax

debugging bgp update l2vpn [ peer { ipv4-address | group-name } ] [ receive | send ] [ verbose ]
undo debugging bgp update l2vpn [ peer { ipv4-address | group-name } ] [ receive | send ] [ verbose ]

Default

Debugging for BGP L2VPN Update messages is disabled.

Views

User view

Default command level

1: Monitor level

Parameters

peer { ipv4-address | group-name }: Specifies BGP L2VPN Update message debugging for the specified peer/peer group.
receive: Specifies the debugging for received BGP packets.
send: Specifies the debugging for sent BGP packets.
verbose: Displays detailed debugging information.

Usage guidelines

Enabling debugging might decrease system performance. Therefore, use this command only when necessary, and disable debugging after the debugging operation is complete.

Table 102 describes output fields and messages for the debugging bgp update l2vpn command.

Table 102 Output from the debugging bgp update l2vpn command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>afi = 196(l2vpn) safi = 128(l2vpn)</td>
<td>Address family is 196. Sub-address family is 128(L2VPN).</td>
</tr>
</tbody>
</table>

Examples

# In an L2VPN, Device A and Device B have established a BGP peer relationship. Enable debugging for BGP L2VPN Update messages on either device. Output similar to the following example is generated:
<Sysname> debugging bgp update l2vpn
*0.84372 Sysname RM/7/RMDEBUG:
BGP.L2VPN: Send UPDATE to 1.1.1.1 for following destinations:
Origin: IGP
AS Path: 200
Next Hop: 5.5.5.5
afi = 196(l2vpn) safi = 128(l2vpn)
debugging bgp update label-route

Use **debugging bgp update label-route** to enable debugging for BGP labeled route Update messages.

Use **undo debugging bgp update label-route** to disable debugging for BGP labeled route Update messages.

**Syntax**

```
debugging bgp update label-route [ peer { ipv4-address | group-name } ] [ acl acl-number | ip-prefix ip-prefix-name ] [ receive | send ] [ verbose ]
```

**Default**

Debugging for BGP labeled route Update messages is disabled.

**Views**

User view

**Default command level**

1: Monitor level

**Parameters**

- **peer { ipv4-address | group-name }**: Specifies BGP labeled route Update message debugging for the specified peer/peer group.
- **acl acl-number**: Filters the output message debugging information with the specified ACL.
- **ip-prefix ip-prefix-name**: Filters the output message debugging information with the specified address prefix.
- **receive**: Specifies the debugging for received BGP labeled route update packets.
- **send**: Specifies the debugging for sent BGP labeled route update packets.
verbose: Displays detailed debugging information.

Usage guidelines

Enabling debugging might decrease system performance. Therefore, use this command only when necessary, and disable debugging after the debugging operation is complete.

Examples

The output in the following example is created when the following conditions exist:

- Device A and Device B have established a BGP peer relationship.
- MPLS has been enabled on the connected interfaces.

# Enable debugging for BGP labeled route Update messages on either device. Output similar to the following example is generated:

```bash
<Sysname> debugging bgp update route-label
```

```
*Sep 4 16:14:32:16 2006 Sysname RM/6/RMDEBUG:
BGP.: Send UPDATE to peer 2.2.2.2 for following destinations:
```

```
Origin       : Incomplete
AS Path      : 1.1.1.1
Next Hop     : 1.1.1.1
Local Pref   : 100
MED          : 0
111.1.1.1/32 (1024)
```

```
*Sep 4 16:14:32:22 2006 Sysname RM/6/RMDEBUG:
BGP_L3VPN.: Recv UPDATE from 2.2.2.2 with following destinations:
```

```
Update message length : 76
Local Pref   : 100
MED          : 0
Origin       : Incomplete
AS Path      :
Next Hop     : 2.2.2.2
111.2.2.2/32 (1025)
```

```
// Update message information for L3VPN labeled routes was displayed.
```

```
*Sep 4 16:17:20:642 2006 Sysname RM/6/RMDEBUG:
BGP.: Send UPDATE to 100::2 for following destinations:
```

```
Origin       : Incomplete
AS Path      :
Next Hop     : 100::1
Local Pref   : 100
MED          : 0
100::/64(1025),
```

```
*Sep 4 16:17:20:708 2006 Sysname RM/6/RMDEBUG:
BGP_IPV6.: Recv UPDATE from peer 100::2 with following destinations:
```

```
Update message length : 86
```
Local Pref : 100
MED          : 0
Origin       : Incomplete
AS Path      :
Next Hop     : 100::2
22::22/128(1027)

// Update message information for IPv6 labeled routes were displayed.

deeding bgp update mdt

Use `debugging bgp update mdt` to enable debugging for BGP MDT Update messages.

Use `undo debugging bgp update mdt` to disable debugging for BGP MDT Update messages.

Syntax

```
debugging bgp update mdt [ peer { ipv4-address | group-name } ] [ receive | send ] [ verbose ]
undo debugging bgp update mdt [ peer { ipv4-address | group-name } ] [ receive | send ] [ verbose ]
```

Default

Debugging for BGP MDT Update messages is disabled.

Views

User view

Default command level

1: Monitor level

Parameters

```
peer { ipv4-address | group-name }: Specifies debugging for BGP messages of the specified peer/peer group.
receive: Specifies the debugging for received BGP MDT messages.
send: Specifies the debugging for sent BGP MDT messages.
verbose: Displays detailed BGP MDT debugging information.
```

Usage guidelines

Disable the debugging in time because it decreases the system performance.

Table 103 describes output fields and messages for the `debugging bgp update mdt` command.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BGP_MDT</td>
<td>BGP MDT instance.</td>
</tr>
<tr>
<td>Recv UPDATE from x.x.x.x</td>
<td>A route Update message was received from BGP neighbor x.x.x.x.</td>
</tr>
<tr>
<td>Recv UPDATE(Withdraw) from x.x.x.x</td>
<td>A route update (withdraw) message was received from BGP neighbor x.x.x.x.</td>
</tr>
<tr>
<td>Send UPDATE to x.x.x.x</td>
<td>A route Update message was sent to BGP neighbor x.x.x.x.</td>
</tr>
</tbody>
</table>
### Field Description

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Send UPDATE(Withdraw) to peer x.x.x.x</td>
<td>A route update (withdraw) message was sent to BGP neighbor x.x.x.x.</td>
</tr>
<tr>
<td>Source x.x.x.x</td>
<td>Source address of multicast VPN in the Update message.</td>
</tr>
<tr>
<td>ShareGrp x.x.x.x</td>
<td>Share-Group of multicast VPN in the Update message.</td>
</tr>
</tbody>
</table>

#### Examples

# Device A and Device B have established a BGP peer relationship in MDT instance. Enable debugging for BGP MDT messages on either device. Output similar to the following example is generated:

```plaintext
<Sysname> debugging bgp update mdt

*Apr 29 14:15:28:15 2008 Sysname RM/6/RMDEBUG:
  BGP_MDT: Send UPDATE to peer 1.1.1.1 for following destinations:
  Origin: Incomplete
  Next Hop: 2.2.2.2
  Local Pref: 100
  RD 100:1, Source 2.2.2.2, ShareGrp 224.1.1.1,

// A route Update message was sent to BGP peer 1.1.1.1: Origin is Incomplete, next hop is 2.2.2.2, local preference is 100, RD is 100:1, source address of multicast VPN is 2.2.2.2, and Share-Group is 224.1.1.1.

*Apr 29 14:15:28:31 2008 Sysname RM/6/RMDEBUG:
  BGP_MDT: Recv UPDATE from peer 1.1.1.1 with following destinations:
  Update message length: 67
  Local Pref: 100
  Origin: Incomplete
  Next Hop: 1.1.1.1
  RD 100:2, Source 1.1.1.1, ShareGrp 224.2.2.2,

// A route Update message was received from BGP peer 1.1.1.1: Update message length is 67, local preference is 100, Origin is Incomplete, next hop is 1.1.1.1, RD is 100:2, source address of multicast VPN is 1.1.1.1, and Share-Group is 224.2.2.2. 
```

**debugging bgp update peer**

Use **debugging bgp update peer** to enable BGP Update message debugging for the specified peer/peer group.

Use **undo debugging bgp update peer** to disable BGP Update message debugging for the specified peer/peer group.

**Syntax**

```
debugging bgp update peer { ipv4-address | group-name } [ acl acl-number | ip-prefix ip-prefix-name ] [ receive | send ] [ verbose ]
```
undo debugging bgp update peer \{ ipv4-address | group-name \} \[ receive | send \] \[ verbose \]

Default
BGP Update message debugging for the specified peer/peer group is disabled.

Views
User view

Default command level
1: Monitor level

Parameters
peer \{ ipv4-address | group-name \}: Specifies BGP VPLS Update message debugging for the specified peer/peer group.

default debugging bgp update vpls

Use debugging bgp update vpls to enable debugging for BGP VPLS Update messages.
Use undo debugging bgp update vpls to disable debugging for BGP VPLS Update messages.

Syntax
default debugging bgp update vpls \[ peer \{ ipv4-address | group-name \} \] \[ receive | send \] \[ verbose \]

Default
Debugging for BGP VPLS Update messages is disabled.

Views
User view

Default command level
1: Monitor level

Parameters
peer \{ ipv4-address | group-name \}: Specifies BGP VPLS Update message debugging for the specified peer/peer group.
receive: Specifies the debugging for the received BGP packets.
send: Specifies the debugging for the sent BGP packets.
verbose: Displays detailed debugging information.

Usage guidelines

Enabling debugging might decrease system performance. Therefore, use this command only when necessary, and disable debugging after the debugging operation is complete.

Table 104 describes output fields and messages for the `debugging bgp update vpls` command.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>afi = 196(vpls) safi = 128(vpls)</code></td>
<td>Address family is 196. Sub-address family is 128 (VPLS).</td>
</tr>
</tbody>
</table>

Examples

# In a VPLS, Device A and Device B have established a BGP peer relationship. Enable debugging for BGP VPLS Update messages on either device. Output similar to the following example is generated:

```plaintext
<Sysname> debugging bgp update vpls
*0.84372 Sysname RM/7/RMDEBUG:
  BGP.VPLS: Send UPDATE to 1.1.1.1 for following destinations :
  Origin : IGP
  AS Path : 200
  Next Hop : 5.5.5.5
  afi = 155(vpls) safi = 128(vpls)
  Route Distinguisher:100:1
  CE-ID:1
  label offset:0
  label base:20480

  // Information about the sent VPLS Update message was displayed.

*0.92566 Sysname RM/7/RMDEBUG:
  BGP.VPLS: Recv UPDATE from 1.1.1.1 with following destinations :
  Update message length : 88
  Origin : IGP
  AS Path : 100
  Next Hop : 1.1.1.1
  afi = 155(vpls) safi = 128(vpls)
  Route Distinguisher:100:1
  CE-ID:0
  label offset:0
  label base:8192

  // Information about the received VPLS Update message was displayed.
```

debugging bgp update vpn-instance

Use `debugging bgp update vpn-instance` to enable BGP Update message debugging for a VPN instance.
Use the `undo debugging bgp update instance` command to disable BGP Update message debugging for a VPN instance.

**Syntax**

```
debugging bgp update vpn-instance vpn-instance-name [ ip-prefix ip-prefix-name | peer { ipv4-address | group-name } ] [ receive | send ] [ verbose ]
undo debugging bgp update vpn-instance vpn-instance-name [ ip-prefix ip-prefix-name | peer { ipv4-address | group-name } ] [ receive | send ] [ verbose ]
```

**Default**

BGP Update message debugging for a VPN instance is disabled.

**Views**

User view

**Default command level**

1: Monitor level

**Parameters**

- `vpn-instance-name`: Specifies the BGP Update message debugging for the specified VPN instance. `vpn-instance-name` is a case-sensitive string of 1 to 31 characters.
- `ip-prefix ip-prefix-name`: Filters the output message debugging information with the specified IP prefix list.
- `peer { ipv4-address | group-name }`: Specifies BGP Update message debugging for the specified peer/peer group.
- `receive`: Specifies the debugging for received BGP packets.
- `send`: Specifies the debugging for the sent BGP packets.
- `verbose`: Displays detailed debugging information.

**Usage guidelines**

Enabling debugging might decrease system performance. Therefore, use this command only when necessary, and disable debugging after the debugging operation is complete.

See the following command for the description of the Update message information.

**debugging bgp update vpnv4**

Use `debugging bgp update vpnv4` to enable debugging for BGP VPNv4 Update messages.

Use `undo debugging bgp update vpnv4` to disable debugging for BGP VPNv4 Update messages.

**Syntax**

```
deploying bgp update vpnv4 [ peer { ipv4-address | group-name } ] [ receive | send ] [ verbose ]
undo deploying bgp update vpnv4 [ peer { ipv4-address | group-name } ] [ receive | send ] [ verbose ]
```

**Default**

Debugging for BGP VPNv4 Update messages is disabled.

**Views**

User view
Default command level

1: Monitor level

Parameters

peer {ipv4-address | group-name}: Specifies BGP Update message debugging for the specified peer/peer group.

receive: Specifies the debugging for the received BGP VPNv4 packets.

send: Specifies the debugging for the sent BGP VPNv4 packets.

verbose: Displays detailed debugging information.

Usage guidelines

Enabling debugging might decrease system performance. Therefore, use this command only when necessary, and disable debugging after the debugging operation is complete.

Examples

# Device A and Device B have established a BGP peer relationship under the VPNv4 instance. Enable debugging for BGP VPNv4 Update messages on either device. Output similar to the following example is generated:

<Sysname> debugging bgp update vpnv4

%Sep 4 16:14:32:00 2006 Sysname RM/3/RMLOG:
BGPP.: 2.2.2.2 State is changed from OPENCONFIRM to ESTABLISHED.

*Sep 4 16:14:32:16 2006 Sysname RM/6/RMDEBUG:
BGPL3VPN.: Send UPDATE to peer 2.2.2.2 for following destinations:

*Sep 4 16:14:32:16 2006 Sysname RM/6/RMDEBUG:
Origin: Incomplete
AS Path: 
Next Hop: 1.1.1.1
Local Pref: 100
MED: 0
Ext-Community: <1 : 1>
111.1.1.1/32 (RD 100:1, Label 1024),

*Sep 4 16:14:32:22 2006 Sysname RM/6/RMDEBUG:
BGPL3VPN.: Recv UPDATE from 2.2.2.2 with following destinations:

Update message length: 92
Local Pref: 100
MED: 0
Ext-Community: <1 : 1>
Origin: Incomplete
AS Path: 
Next Hop: 2.2.2.2
111.2.2.2/32 (RD 100:2, Label 1024),

*Sep 4 16:14:33:164 2006 Sysname RM/6/RMDEBUG:
debugging bgp update vpnv6

Use `debugging bgp update vpnv6` to enable debugging for BGP VPNv6 Update messages.

Use `undo debugging bgp update vpnv6` to disable debugging for BGP VPNv6 Update messages.

**Syntax**

```
debugging bgp update vpnv6 [ peer ipv4-address ] [ receive | send ] [ verbose ]
undo debugging bgp update vpnv6 [ peer ipv4-address ] [ receive | send ] [ verbose ]
```

**Default**

Debugging for BGP VPNv6 Update messages is disabled.

**Views**

User view

**Default command level**

1: Monitor level

**Parameters**

- **peer ipv4-address**: Specifies BGP Update message debugging for the specified peer/peer group.
- **receive**: Specifies the debugging for the received BGP VPNv6 packets.
- **send**: Specifies the debugging for the sent BGP VPNv6 packets.
- **verbose**: Displays detailed debugging information.

**Usage guidelines**

See the previous command for the description of the Update message information.
Bridging debugging commands

The bridging module name is identified as "BRIDGE" in debugging messages.
The output description tables in this document only contain fields and messages that require an explanation.

depending bridge error

Use `debugging bridge error` to enable debugging for transparent bridging errors.
Use `undo debugging bridge error` to disable debugging for transparent bridging errors.

Syntax

```
debugging bridge error
undo debugging bridge error
```

Default

Debugging for bridging errors is disabled.

Views

User view

Default command level

1: Monitor level

Usage guidelines

Table 105 describes output fields and messages for the `debugging bridge error` command.

Table 105: Output from the `debugging bridge error` command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Failed to reference-copy packet from BVI</td>
<td>The system determined that the outgoing interface for a packet is the bridge virtual interface (BVI) by searching the routing table/Layer 3 forwarding table, but failed to copy the packet from the BVI when delivering it in the bridge set.</td>
</tr>
<tr>
<td>Failed to rawcopy mbuf when deliver to BVI</td>
<td>BRIDGE failed to copy the packet when delivering it to the BVI.</td>
</tr>
<tr>
<td>Failed to Set mac-borrow flag</td>
<td>BRIDGE failed to set the MAC borrowed flag bit for the Ethernet port during BVI initialization.</td>
</tr>
<tr>
<td>Specified interface does not exist</td>
<td>BRIDGE is unable to find the physical or virtual interface during packet forwarding or command execution.</td>
</tr>
<tr>
<td>Process vlan id failed</td>
<td>BRIDGE failed to get the VLAN ID when delivering a packet to an Ethernet sub-interface.</td>
</tr>
<tr>
<td>Mac-addresses of TB has reached the maximum</td>
<td>The number of MAC entries had reached the upper limit, and no more MAC entries can be configured or learned.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td>Static mac-addresses of TB has reached the maximum.</td>
<td>The number of static MAC entries had reached the upper limit, and no more static MAC entries can be configured.</td>
</tr>
<tr>
<td>TB failed to learn address: the source mac address is invalid</td>
<td>BRIDGE failed to learn a MAC address, because the MAC address is invalid. Broadcast, multicast, and all-0 MAC addresses are all invalid MAC addresses.</td>
</tr>
</tbody>
</table>
| TB output process error: error description | An error occurred during packet delivery to the outgoing interface based on the destination address of the packet. The error is of one of the following types:  
  - Null packet.  
  - Outgoing interface does not exist.  
  - Unable to get the link-layer protocol type of the outgoing interface.  
  - Incorrect link-layer protocol type of the outgoing interface. |
| TB module process error: error description | An error occurred before bridging a packet received on an Ethernet interface. The error is of one of the following types:  
  - Null packet.  
  - No valid data in the packet.  
  - Packet incoming interface does not exist (possibly an Ethernet interface or sub-interface).  
  - Error in packet type analysis.  
  - Failed to get the Ethernet link-layer control block.  
  - Failed to copy packet. |
| TB l3intercept from eth: error description | The system determined that the outgoing interface for a packet is BVI by searching the routing table/Layer 3 forwarding table, but failed to deliver the packet. The error is of one of the following types:  
  - Null packet.  
  - Failed to get the outgoing interface type.  
  - Transparent bridging/transparent bridge routing is not enabled on the device.  
  - Failed to get private data block.  
  - Invalid transparent bridge set ID of the BVI.  
  - Transparent bridge routing is not enabled on the transparent bridge set of the BVI. |
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Send DLSw packet on set failed, **error description** | An error occurred during packet delivery within the specified transparent bridge set. The error is of one of the following types:  
  - Null packet.  
  - The specified bridge set ID is invalid.  
  - The specified bridge set is not enabled.  
  - The bridge set contains no DLSw virtual interface.  
  - Some interfaces in the bridge set are invalid.  
  - Some interfaces in the bridge set are not Ethernet interfaces.  
  - VLAN processing failed when the outgoing interface was an Ethernet interface. |
| Send DLSw packet on interface failed, **error description** | An error occurred during packet delivery on the specified outgoing interface. The error is of one of the following types:  
  - Null packet.  
  - The specified outgoing interface does not exist.  
  - The specified outgoing interface is not an Ethernet interface.  
  - Transparent bridging is not enabled on the device.  
  - Invalid transparent bridge set data block associated with the interface.  
  - The specified outgoing interface is not in the bridge set.  
  - The bridge set contains no DLSw virtual interface.  
  - VLAN processing failed when the outgoing interface was an Ethernet interface. |
| Clear DLSw circuit with set failed, **error description** | An error occurred during deletion of all DLSw virtual interfaces associated with the specified transparent bridge set. The error is of one of the following types:  
  - The specified bridge set ID is invalid.  
  - The specified bridge set is not enabled. |

**debugging bridge eth-forwarding**

Use **debugging bridge eth-forwarding** to enable debugging for transparent bridging forwarding.  
Use **undo debugging bridge eth-forwarding** to disable debugging for transparent bridging forwarding.

**Syntax**

```
debugging bridge eth-forwarding [ dlsw | interface interface-type interface-number ]
undo debugging bridge eth-forwarding [ dlsw | interface interface-type interface-number ]
```

**Default**

Debugging for transparent bridging forwarding is disabled.

**Views**

User view
Default command level

1: Monitor level

Parameters

dlsw: Debugging for DLSw virtual interfaces.
interface: Debugging for the specified interface.

interface-type interface-number: Specifies an interface by its type and number.

Usage guidelines

Table 106 describes output fields and messages for the debugging bridge eth-forwarding command.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Forward a frame</td>
<td>Action for the packet.</td>
</tr>
<tr>
<td>received from interface interface-type interface-number</td>
<td>Incoming interface of the packet.</td>
</tr>
<tr>
<td>send to interface interface-type interface-number</td>
<td>Outgoing interface of the packet (this field is not printed if no outgoing interface exists).</td>
</tr>
<tr>
<td>frame head</td>
<td>Packet header content.</td>
</tr>
</tbody>
</table>

Examples

**Figure 2 Network diagram**

# Enable debugging for transparent bridging forwarding on Router B. Output similar to the following example is generated when you ping Router C from Router A under the following conditions:

- Transparent bridging is enabled on Router B.
- Ethernet 1/2 and Ethernet 1/3 on Router B are assigned to the same bridge set.

```
<Sysname> debugging bridge eth-forwarding
*0.63325961 Sysname BRIDGE/7/DEBUGGING:
Broadcast a frame; and try to send to interface Ethernet1/3, with following frame head:
ff ff ff ff ff 00 0f e2 00 00 21 08 06
// BRIDGE received an ARP request on Ethernet 1/2 and forwarded the ARP request to all interfaces but Ethernet 1/2 in the bridge set.
*0.63325961 Sysname BRIDGE/7/DEBUGGING:
Forward a frame; received from interface Ethernet1/3; and try to send to interface Ethernet1/2, with following frame head:
00 0f e2 00 00 21 00 e0 fc 3a 5f 07 08 06
// BRIDGE received an ARP reply on Ethernet 1/3 and forwarded the ARP reply to Ethernet 1/2.
*0.63325962 Sysname BRIDGE/7/DEBUGGING:
Forward a frame; received from interface Ethernet1/2; and try to send to interface Ethernet1/3, with following frame head:
00 e0 fc 3a 5f 07 00 0f e2 00 00 21 08 00
```
// BRIDGE received a ping packet on Ethernet 1/2 and forwarded the ping packet to Ethernet 1/3.
*0.63325963 Sysname BRIDGE/7/DEBUGGING:
Forward a frame; received from interface Ethernet1/3; and try to send to interface
Ethernet1/2, with following frame head:
00 0f e2 00 00 21 00 0e 0 fc 3a 5f 07 08 00
// BRIDGE received a ping reply packet on Ethernet 1/3 and forwarded the ping replay packet to
Ethernet 1/2.

debugging bridge event

Use `debugging bridge event` to enable debugging for transparent bridging events.

Use `undo debugging bridge event` to disable debugging for transparent bridging events.

Syntax

```
debugging bridge event
```

```
undo debugging bridge event
```

Default

Debugging for transparent bridging events is disabled.

Views

User view

Default command level

1: Monitor level

Usage guidelines

Table 107 describes output fields and messages for the `debugging bridge event` command.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TB module inactive, process PASS</td>
<td>Transparent bridging is not enabled on the device. Packets received on Ethernet interfaces cannot be forwarded through transparent bridging. They undergo other types of forwarding or are passed to upper application protocols.</td>
</tr>
<tr>
<td>Packet is being handled by TB interceptor</td>
<td>Transparent bridging is enabled on the device. A further decision will be made about whether packets received on Ethernet interfaces need to be forwarded through transparent bridging.</td>
</tr>
<tr>
<td>00.e0.fc.3a.5f.07.00.0f.e2.00.00.21.08.00.45.00.00.54.00.2b.00.00.ff.01.b7</td>
<td>The first 25 bytes of the packet to be processed.</td>
</tr>
<tr>
<td>TB_Receive process, from %s</td>
<td>The incoming interface of the packet to be processed.</td>
</tr>
<tr>
<td>Input packet tag:%d</td>
<td>VLAN tag of the packet, if any.</td>
</tr>
<tr>
<td>Search sub-interface finished</td>
<td>Failed to find the Ethernet sub-interface corresponding to the VLAN tag carried in the packet.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Interface doesn’t belong to any bridge-set. PASS</td>
<td>The incoming interface of a packet is not in any bridge set, so the packet does not need to be forwarded through transparent bridging. The packet undergoes other types of forwarding or is passed to the upper application protocols.</td>
</tr>
<tr>
<td>Broadcast or multicast packet received, intercept and proc PASS</td>
<td>The packet to be processed is a broadcast or multicast packet. A copy of the packet is forwarded through transparent bridging, and the original packet undergoes other types of forwarding or is passed to the upper application protocols.</td>
</tr>
<tr>
<td>Unicast packet received, intercept, proc END</td>
<td>The destination MAC address of the packet to be processed is not the MAC address of the incoming interface, so it needs to be forwarded through transparent forwarding.</td>
</tr>
<tr>
<td>Unicast packet received, bridge-routing enabled, proc END</td>
<td>The destination MAC address of the packet to be processed is the MAC address of the incoming interface, but the transparent bridge routing function is enabled on the device (the bridge routing-enable and bridge bridge-set routing { ip</td>
</tr>
<tr>
<td>Unicast packet received, proc PASS</td>
<td>The destination MAC address of the packet to be processed is the MAC address of the incoming interface, and the transparent bridge routing function is not enabled on the device. As a result, the packet needs to go through other types of forwarding or be passed to the upper application protocols.</td>
</tr>
<tr>
<td>Interface doesn’t belong to any bridge-set</td>
<td>The incoming interface of a transparently bridged packet received from a remote endpoint is a non-Ethernet interface (an ATM interface, or PPP/FR/X.25/HDLC encapsulated synchronous serial interface) that belongs to no bridge set, so this even is output and the packet is dropped.</td>
</tr>
<tr>
<td>frame type:%d,protocol type:%d</td>
<td>The incoming interface belongs to a bridge set. The Ethernet frame header in the packet is analyzed, and the packet type and protocol type are output.</td>
</tr>
<tr>
<td>TB_Receive process,analyse error</td>
<td>Error in Ethernet frame header analysis.</td>
</tr>
<tr>
<td>Packet is received from interface %s</td>
<td>Ethernet frame header analysis completed, and the incoming interface is given.</td>
</tr>
<tr>
<td>Packet is broadcasted in native bridge-set</td>
<td>The packet is a broadcast packet or no MAC entry for its destination MAC address is found, so that packet is broadcast in the native bridge set.</td>
</tr>
<tr>
<td>DLSw frame is passed to DLSw module</td>
<td>The packet is a DLSw packet, so it is passed to the DLSw module for further processing.</td>
</tr>
<tr>
<td>TB output process : send to interface %s, link layer protocol %d</td>
<td>Outgoing interface of the packet and its link-layer protocol type.</td>
</tr>
</tbody>
</table>
**Field** | **Description**
--- | ---
TB Filter: COS has been set | This event is printed if the Ethernet frame filter function is configured on the incoming/outgoing interface for transparent bridge forwarding and the packet carries a VLAN tag.

TB Filter: Packet has no VLAN tag, ignore COS | This event is printed if the Ethernet frame filter function is configured on the incoming/outgoing interface for transparent bridge forwarding and the packet carries no VLAN tag.

It is no need to select MAC again | This event is printed when an Ethernet interface joins a bridge set after the BVI of the bridge set borrows a MAC address and the Ethernet interface that owns the MAC address leaves the bridge set.

### Examples

#### Figure 3 Network diagram

![Network diagram](image)

# Enable debugging for transparent bridging events on Router B. Output similar to the following example is generated when you ping Router C from Router A under the following conditions:

- Transparent bridging is enabled on Router B.
- Ethernet 1/2 and Ethernet 1/3 on Router B are assigned to the same bridge set.
- An ARP entry already exists.

```
<Sysname> debugging bridge event
*0.64266100 Sysname BRIDGE/7/EVENT: Packet is being handled by TB intercepter.
*0.64266300 Sysname BRIDGE/7/EVENT: 00.e0.fc.3a.5f.07.00.0f.e2.00.00.21.08.00.45.00.00.54.00.2b.00.00.ff.01.b7
*0.64266300 Sysname BRIDGE/7/EVENT: TB_Receive process, from Ethernet1/1.
*0.64266500 Sysname BRIDGE/7/EVENT: frame type:0, protocol type:2048
*0.64266500 Sysname BRIDGE/7/EVENT: Packet is received from interface Ethernet1/1
*0.64266700 Sysname BRIDGE/7/EVENT: Unicast packet received, intercept, proc END.

// BRIDGE received a ping packet on Ethernet 1/1. BRIDGE passed the packet to the transparent bridging module for further processing because both of the following conditions exist:

- The incoming interface is already in the native bridge set.
- The packet is a unicast packet and its destination MAC address is not the MAC address of the incoming interface.
```

```
*0.64266700 Sysname BRIDGE/7/EVENT: Packet is being handled by TB module.
*0.64266910 Sysname BRIDGE/7/EVENT: TB output process: send to interface Ethernet1/2, link layer protocol 4
*0.64266910 Sysname BRIDGE/7/EVENT:
```
The transparent bridging module found the outgoing interface of the packet, which is Ethernet 1/2, and forwarded the packet. Packet is being handled by TB intercepter.

*0.64267110 Sysname BRIDGE/7/EVENT:
00.0f.e2.00.00.21.00.e0.fc.3a.5f.07.08.00.45.00.00.54.00.0a.00.00.ff.01.b7.*0.6426711
0 Sysname BRIDGE/7/EVENT:
TB_Receive process, from Ethernet1/2.
*0.64267310 Sysname BRIDGE/7/EVENT:frame type:0, protocol type:2048
*0.64267310 Sysname BRIDGE/7/EVENT: Packet is received from interface Ethernet1/2
*0.64267510 Sysname BRIDGE/7/EVENT:
Unicast packet received, intercept, proc END.

BRIDGE received a ping reply packet on Ethernet 1/2. BRIDGE passed the packet to the transparent bridging module for further processing because both of the following conditions are true:

- The incoming interface is already in the native bridge set.
- The packet is a unicast packet and its destination MAC address is not the MAC address of the incoming interface.

*0.64267510 Sysname BRIDGE/7/EVENT: Packet is being handled by TB module.
*0.64267730 Sysname BRIDGE/7/EVENT: TB output process: send to interface Ethernet1/1, link layer protocol 4

The transparent bridging module found the outgoing interface of the packet, which is Ethernet 1/1, and forwarded the packet.
Call services debugging commands

The output description tables in this document only contain fields and messages that require an explanation.

debugging voice ss cb

Use `debugging voice ss cb` to enable call backup debugging.

Use `undo debugging voice ss cb` to disable call backup debugging.

Syntax

```
  debugging voice ss cb { all | error | event | info | timer }
```

```
  undo debugging voice ss cb { all | error | event | info | timer }
```

Default

Call backup debugging is disabled.

Views

User view

Default command level

2: System level

Parameters

- `all`: Specifies all types of debugging for call backup.
- `error`: Specifies error debugging.
- `event`: Specifies event debugging.
- `info`: Specifies information debugging.
- `timer`: Specifies timer debugging.

Usage guidelines

Table 108 describes output fields and messages for the `debugging voice ss cb error` command.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>The service handle notify parameter is null in CB process.</td>
<td>The pointer to the current service block was null.</td>
</tr>
<tr>
<td>The CCB Parameter is null in CB process.</td>
<td>The pointer to the call control block (CCB) was null.</td>
</tr>
<tr>
<td>The brother leg is invalid.</td>
<td>The pointer to the brother leg was null.</td>
</tr>
<tr>
<td>Can’t fill in the new information to the in Leg.</td>
<td>Call backup failed to update the call leg information.</td>
</tr>
<tr>
<td>The in leg is not LGS type.</td>
<td>The input leg is not a circuit-switched (CS) leg.</td>
</tr>
<tr>
<td>Received invalid call leg parameter.</td>
<td>The pointer to the call leg was invalid.</td>
</tr>
<tr>
<td>Null pointer argument.</td>
<td>The input pointer was null.</td>
</tr>
</tbody>
</table>
Table 109 describes output fields and messages for the **debugging voice ss cb event** command.

**Table 109 Output from the debugging voice ss cb event command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Receive CMC_RELEASE message from SA module.</td>
<td>A CMC_RELEASE message was received from the SA module.</td>
</tr>
</tbody>
</table>

Table 110 describes output fields and messages for the **debugging voice ss cb info** command.

**Table 110 Output from the debugging voice ss cb info command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Send setup message to SPL module</td>
<td>A SETUP message was sent to the SPL module.</td>
</tr>
<tr>
<td>Receive message from in leg.</td>
<td>A message was received from the input call leg.</td>
</tr>
<tr>
<td>Can not get the new entity.</td>
<td>Call backup failed to get the index of a new entity.</td>
</tr>
<tr>
<td>Can not find the new entity.</td>
<td>Call backup failed to find an available entity.</td>
</tr>
</tbody>
</table>

Table 111 describes output fields and messages for the **debugging voice ss cb timer** command.

**Table 111 Output from the debugging voice ss cb timer command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Timer deleted, Timer ID: TIMER_INLEG_WAIT_ALERTING.</td>
<td>The INLEG_WAIT_ALERTING timer was deleted.</td>
</tr>
</tbody>
</table>

Examples

# Enable all types of debugging for call backup on the router. Output similar to the following example is generated when a telephone (1000) attempts to call the number 3000 under these conditions:

- The router (192.168.0.31) has two voice entities (3000 and 3001) to reach the telephone number 3000:
  - Entity 3000 has a priority of 1, and its destination IP address is 192.168.0.38.
  - Entity 3001 has a priority of 3, and its destination IP address is 192.168.0.32.
- The gateway at 192.168.0.38 fails.

`<RouterA> debug voice ss cb all`  
SS CB all debugging switches are on  
`<RouterA>`

// Router A found voice entity 3000 to originate a call according to the dial plan.  
// The gateway at 192.168.0.38 failed to respond, and the call was transferred to the call backup module for processing.

*Nov 14 08:18:11:37 2005 RouterA SS/7/VOICE:  
CB_EVT: [0x00030001] Receive CMC_RELEASE message from SA module.  
// A RELEASE message was received.*
CB_INFO: [0x00030001] The CMC_RELEASE message's release cause: Wait alerting message timeout

// The cause for the RELEASE message was checked.

*Nov 14 08:18:11:38 2005 RouterA SS/7/VOICE: CB_TMR: [0x00030000] Timer deleted, Timer ID: TIMER_INLEG_WAIT_ALERTING.

// A call was ready to be re-originated, and the INLEG_WAIT_ALERTING timer was restarted.

*Nov 14 08:18:11:38 2005 RouterA SS/7/VOICE: CB_INFO: [0x00030000] Send CMC_SETUP message to start a new call.

// A call was re-originated to voice entity 3001, which corresponds to the backup gateway at 192.168.0.32.

debugging voice ss cf

Use debugging voice ss cf to enable call forwarding debugging.
Use undo debugging voice ss cf to disable call forwarding debugging.

Syntax

debugging voice ss cf { all | error | event | fsm | info | timer }
undo debugging voice ss cf { all | error | event | fsm | info | timer }

Default

Call forwarding debugging is disabled.

Views

User view

Default command level

2: System level

Parameters

all: Specifies all types of debugging for call forwarding.
error: Specifies error debugging.
event: Specifies event debugging.
fsm: Specifies finite state machine debugging.
info: Specifies information debugging.
timer: Specifies timer debugging.

Usage guidelines

Table 112 describes output fields and messages for the debugging voice ss cf error command.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>The leg is null on CFR idle state</td>
<td>The input leg was null in the call forwarding recipient (CFR) idle state. This information is output when the input leg is incorrect.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td>The message is null on CFR idle state</td>
<td>The input message was null in the CFR idle state. This information is output when the input message is incorrect.</td>
</tr>
<tr>
<td>The service handle notify is null on CFR idle state</td>
<td>The input service handle notify was null in the CFR idle state. This information is output when the input service handle notify is incorrect.</td>
</tr>
<tr>
<td>The CCB is null on CFR idle state</td>
<td>The input CCB was null in the CFR idle state. This information is output when the input CCB is incorrect.</td>
</tr>
<tr>
<td>The message set type is null on CFR idle state</td>
<td>The input message set type was null in the CFR idle state. This information is output when the input message set type is incorrect.</td>
</tr>
<tr>
<td>The message type is null on CFR idle state</td>
<td>The input message type was null in the CFR idle state. This information is output when the input message type is incorrect.</td>
</tr>
<tr>
<td>Receive error message from SA module on CFR idle state</td>
<td>The CFR received an incorrect message from the SA module. This information is output when different finite state machines receive any incorrect message.</td>
</tr>
<tr>
<td>Receive error service message from SA module on CFR idle state</td>
<td>The CFR received an incorrect service message from the SA module. This information is output when different finite state machines receive any incorrect service message.</td>
</tr>
<tr>
<td>Failed to get brother leg on CFR idle state</td>
<td>The CFR in the idle state failed to get a brother leg. This information is output when any error occurs in getting a brother leg.</td>
</tr>
<tr>
<td>Input leg is wrong in CFR idle state</td>
<td>The input leg of the CFR was incorrect. This information is output in different states when the type or ID of the input leg is incorrect.</td>
</tr>
<tr>
<td>Failed to disconnect media in CFR idle</td>
<td>The media failed to be disconnected in the CFR idle state. This information is output when different finite state machines fail to disconnect the media.</td>
</tr>
<tr>
<td>Failed to delete CCB on CFR idle state</td>
<td>The CCB failed to be deleted in the CFR idle state. This information is output when different finite state machines fail to delete the CCB.</td>
</tr>
<tr>
<td>Failed to setup all new call on CFR idle state. Will release the service call</td>
<td>The CFR failed to set up a new call, and the call was cleared.</td>
</tr>
<tr>
<td>Failed to get new call number on CFR idle state</td>
<td>The CFR failed to obtain a new call number in the idle state.</td>
</tr>
<tr>
<td>Input leg is wrong or get brother leg fail on CFR pending state</td>
<td>The input leg was incorrect or getting a brother leg in the CFR pending state. This information is output in different states when the type or ID of the input leg is incorrect.</td>
</tr>
<tr>
<td>Failed to setup new call by all numbers on CFR pending state</td>
<td>The CFR in the pending state failed to set up a new call using all numbers. This information is output when the CFR in the pending state fails to set up a new call using all numbers.</td>
</tr>
<tr>
<td>The leg is invalid on CFR pending state</td>
<td>The input leg of the CFR was invalid in the idle state.</td>
</tr>
<tr>
<td>The CMC CCB is null on CFO idle state</td>
<td>The input CMC CCB was null. This information is output when the CMC CCB was null in different states.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Received brother leg is wrong on CFO Idle state</td>
<td>The type of the received brother leg was incorrect in the call forwarding originator (CFO) idle state.</td>
</tr>
<tr>
<td>Failed to find VIM index on CFO idle state</td>
<td>The CFO failed to find a VIM interface.</td>
</tr>
<tr>
<td>CF number is same to the local’s on CFO idle state</td>
<td>The CFO in the idle state found that the forwarded-to number was a local number. This information is output when the forwarded-to number is a local number in different states of the CFO and the CFR.</td>
</tr>
<tr>
<td>Failed to send service message on CFO idle state</td>
<td>The CFO failed to send a SERVICE message in the idle state. This information is output when the CFO failed to send a SERVICE message.</td>
</tr>
<tr>
<td>Failed to create timer on CFO idle state</td>
<td>The CFO failed to create a timer in the idle state.</td>
</tr>
<tr>
<td>Failed to find the receiver leg on CFO wait release state</td>
<td>The CFO failed to find a brother leg when waiting for a RELEASE message.</td>
</tr>
<tr>
<td>Input leg is wrong for CFU on CFO wait release state</td>
<td>The input leg for the CFO was incorrect when the CFO was waiting for a RELEASE message. This information is output when the CFO finds the input leg incorrect in different states.</td>
</tr>
<tr>
<td>Input leg is wrong for CFNR on CFO wait release state</td>
<td>The input leg for call forwarding no reply (CFNR) was incorrect when the CFO was waiting for a RELEASE message.</td>
</tr>
<tr>
<td>Input leg is wrong for CFNA on CFO wait release state</td>
<td>The input leg for call forwarding unavailable was incorrect when the CFO was waiting for a RELEASE message.</td>
</tr>
<tr>
<td>Receive error release message from SA module on CFO wait release state</td>
<td>The CFO received an incorrect RELEASE message from the SA module when waiting for a RELEASE message.</td>
</tr>
<tr>
<td>Failed to delete timer on CFO wait service ack state</td>
<td>The CFO failed to delete the timer when waiting for an ACK message.</td>
</tr>
<tr>
<td>CF service type is unknown on CFO wait service ack state</td>
<td>The call forwarding type was unknown when the CFO was waiting for an ACK message.</td>
</tr>
<tr>
<td>Failed to get service information in CFO wait ack stat</td>
<td>The CFO failed to get the service configuration information when waiting for an ACK message.</td>
</tr>
<tr>
<td>Receive error ack message from SA module on CFO wait service ack state</td>
<td>The CFO received an incorrect ACK message when waiting for an ACK message.</td>
</tr>
<tr>
<td>The FEATURE service failed to allocate the memory failed for CF CCB</td>
<td>Call forwarding failed to allocate memory to the CF CCB.</td>
</tr>
<tr>
<td>The CCB is null in CF Delete CCB process</td>
<td>The CF CCB to be deleted was null.</td>
</tr>
<tr>
<td>Failed to delete CFO wait service ack timer in delete CCB process</td>
<td>Call forwarding failed to delete the WAIT_ACK timer in the CF CCB deletion process.</td>
</tr>
<tr>
<td>Failed to add new address in CFR service process</td>
<td>Call forwarding failed to add a new address when the CFR received multiple addresses.</td>
</tr>
<tr>
<td>Failed to find brother leg in CFR service process</td>
<td>The CFR failed to find a brother leg when processing multiple addresses.</td>
</tr>
<tr>
<td>The address table is null in getting new number process</td>
<td>The address table was null.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td>The new user information is null in get new number process</td>
<td>The user information of a new number was null.</td>
</tr>
<tr>
<td>Current call number is zero in getting new number process</td>
<td>The new number obtained was 0 (no number was available to originate a new call).</td>
</tr>
<tr>
<td>The original user information is null in setuing new call process</td>
<td>Input parameter check for a new call origination: The information of the original forwarded-to user was null.</td>
</tr>
<tr>
<td>The redirect information is null in setuing new call process</td>
<td>Input parameter check for a new call origination: The recent call forwarding information was null.</td>
</tr>
<tr>
<td>The new called information is null in setuing new call process</td>
<td>Input parameter check for a new call origination: The information of the called number in the new call origination was null.</td>
</tr>
<tr>
<td>Called number is same to the local’s in setuing new call process</td>
<td>The called number in the new call origination was a local number.</td>
</tr>
<tr>
<td>Find the number correctly but doesn’t match the IP address</td>
<td>The called number was correct but the IP address did not match.</td>
</tr>
<tr>
<td>Failed to send setup when process the service(CF) message</td>
<td>The new call origination failed after the SERVICE (CF) message was processed.</td>
</tr>
<tr>
<td>Failed to send the ACCP_SETUP_ACK message in setup new call</td>
<td>Call forwarding failed to send an ACCP_SETUP_ACK message in new call setup.</td>
</tr>
<tr>
<td>Failed to send the ACCP_ALERTING message in setup new call</td>
<td>Call forwarding failed to send an ACCP_ALERTING message in new call setup.</td>
</tr>
<tr>
<td>The forward information is null in sending service CF process</td>
<td>The forwarding information in the sent SERVICE(CF) message was found null.</td>
</tr>
<tr>
<td>Failed to find entity type in CFO unsupported process</td>
<td>The CFO failed to find the entity type in local processing.</td>
</tr>
<tr>
<td>The changed called number is null in finding voice entity process</td>
<td>The called number after number substitution was null in finding voice entities.</td>
</tr>
<tr>
<td>The changed caller number is null in finding voice entity process</td>
<td>The calling number after number substitution was null in finding voice entities.</td>
</tr>
<tr>
<td>The FEATURE service failed to obtain call information table in finding voice entity process</td>
<td>Call forwarding failed to obtain the call information table in finding voice entities.</td>
</tr>
<tr>
<td>The FEATURE service failed to obtain entity for call in finding voice entity process</td>
<td>Call forwarding failed to obtain a voice entity in finding voice entities.</td>
</tr>
<tr>
<td>The FEATURE service failed to obtain first voice entity in finding voice entity process</td>
<td>Call forwarding failed to obtain the first voice entity in finding voice entities.</td>
</tr>
<tr>
<td>Cannot find matchable entity for call in finding voice entity process</td>
<td>No voice entity could be matched.</td>
</tr>
<tr>
<td>Dial peer is null in getting user info process</td>
<td>The dial peer was found to be null in getting user information.</td>
</tr>
<tr>
<td>Leg is null in recording setup’s information process</td>
<td>The input leg was found to be null in the special processing of the SETUP message.</td>
</tr>
<tr>
<td>Receive error message in recording setup’s information process</td>
<td>An ERROR message was received in the special processing of the SETUP message.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Leg’s father CCB is null in recording setup’s information process</td>
<td>The father CCB of the leg was found to be null in the special processing of the SETUP message.</td>
</tr>
<tr>
<td>Input parameter is null in predicting CF service process</td>
<td>The call number was found to be null in predicting the CF service.</td>
</tr>
<tr>
<td>Dial peer is null in predicting process</td>
<td>The dial peer was found to be null in predicting the CF service.</td>
</tr>
<tr>
<td>The input is null in matching voice entity process</td>
<td>The input leg or number was found to be null in processing CF information.</td>
</tr>
<tr>
<td>The CMC CCB is null in finding voice entity by forward information process</td>
<td>The CMC CCB was found to be null in processing CF information.</td>
</tr>
<tr>
<td>The FEATURE service failed to obtain call information table in finding voice entity by forward information process</td>
<td>Call forwarding failed to obtain the call information table in processing CF information.</td>
</tr>
<tr>
<td>The FEATURE service failed to obtain service configuration in creating CF register table process.</td>
<td>Call forwarding failed to obtain the service configuration in creating a CF registration table.</td>
</tr>
<tr>
<td>Redirect exceed the max limit and release the new call in interface process</td>
<td>The number of redirections exceeded the limit, and the call was released in interface processing.</td>
</tr>
<tr>
<td>Could not setup new call in interface process</td>
<td>Call forwarding failed to re-originate a call in interface processing.</td>
</tr>
<tr>
<td>Failed to create CF CCB in interface process</td>
<td>Call forwarding failed to create a CCB. This information is output when a CCB fails to be created.</td>
</tr>
<tr>
<td>Has zero addresses in message in interface process</td>
<td>No forwarded-to number was received in interface processing.</td>
</tr>
<tr>
<td>Receiver’s state is error in interface process</td>
<td>The CFR’s state was incorrect in interface processing.</td>
</tr>
<tr>
<td>Could not judge CF role in interface process</td>
<td>The CF role could not be judged in interface processing.</td>
</tr>
<tr>
<td>Receive invalid message from SA Module in interface process.</td>
<td>Invalid messages were received from the SA module in interface processing, and the message set type and the message type were displayed.</td>
</tr>
<tr>
<td>message set type: message type:</td>
<td></td>
</tr>
<tr>
<td>CF originator’s state is Error in interface process</td>
<td>The CFO’s state was incorrect in interface processing.</td>
</tr>
</tbody>
</table>

Table 113 describes output fields and messages for the **debugging voice ss cf event** command.

Table 113 Output from the debugging voice ss cf event command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Send CMC_SERVICE_ACK message to CMC module in CFR Idle state</td>
<td>The CFR in the idle state sent a CMC_SERVICE_ACK message to the CMC module.</td>
</tr>
<tr>
<td>Send CMC_SERVICE_ACK message to CMC module</td>
<td>The CFR sent a CMC_SERVICE_ACK message to the CMC module.</td>
</tr>
<tr>
<td>Send CMC_SETUP message to CMC module in CFO unsupported process state</td>
<td>The CFO in local processing state sent a CMC_SETUP message to the CMC module.</td>
</tr>
<tr>
<td>Send CMC_RELEASE message to CMC module in CFR Pending state</td>
<td>The CFR in the pending state sent a CMC_RELEASE message to the CMC module.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Send CMC_RELEASE message to CMC module in CFO Idle state</td>
<td>The CFO in the idle state sent a CMC_RELEASE message to the CMC module.</td>
</tr>
<tr>
<td>Send CMC_RELEASE message to CMC module in CFO wait service ack state</td>
<td>The CFO in the waiting-for-service-ack state sent a CMC_RELEASE message to the CMC module.</td>
</tr>
<tr>
<td>Send CMC_RELEASE message to CMC module in CF interface process state</td>
<td>A CMC_RELEASE message was sent to the CMC module in CF interface processing.</td>
</tr>
<tr>
<td>Delete CF CCB successfully.</td>
<td>The CF CCB was successfully deleted.</td>
</tr>
<tr>
<td>CMC setup was dismissed in CF CMC setup special process</td>
<td>The CMC_SETUP message disappeared after special processing by the CF CMC module.</td>
</tr>
<tr>
<td>Receive CMC_SERVICE message from SA module</td>
<td>A CMC_SERVICE message was received from the SA module, and the message included the following information:</td>
</tr>
<tr>
<td>CMC_SERVICE message include the following information:</td>
<td>Original flag:, count:, cause:</td>
</tr>
<tr>
<td></td>
<td>Redirect flag:, count:, cause:</td>
</tr>
</tbody>
</table>

Table 114 describes output fields and messages for the **debugging voice ss cf fsm** command.

**Table 114 Output from the debugging voice ss cf fsm command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>State changed from CFR idle to CFR pending</td>
<td>The state of the CFR changed from idle to pending.</td>
</tr>
<tr>
<td>CFR state changed from pending to idle</td>
<td>The state of the CFR changed from pending to idle.</td>
</tr>
<tr>
<td>CFO state changed from idle to waiting release</td>
<td>The state of the CFO changed from idle to waiting-for-release.</td>
</tr>
<tr>
<td>CFO state changed from waiting release to waiting service ack</td>
<td>The state of the CFO changed from waiting-for-release to waiting-for-service-ack.</td>
</tr>
</tbody>
</table>

Table 115 describes output fields and messages for the **debugging voice ss cf info** command.

**Table 115 Output from the debugging voice ss cf info command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Send Last number successfully on CFR idle state</td>
<td>The CFR sent the last number successfully.</td>
</tr>
<tr>
<td>Send setup message successfully on CFR pending state</td>
<td>The CFR sent a SETUP message successfully.</td>
</tr>
<tr>
<td>Receive normal onhook message on CFR pending state</td>
<td>The CFR received a normal hook message in the pending state.</td>
</tr>
<tr>
<td>Send setup message successfully on CFR pending state</td>
<td>The CFR successfully sent a SETUP message in the pending state.</td>
</tr>
<tr>
<td>Setup message information in CFO idle state: Original flag:, count:,</td>
<td>The CFO sent a SETUP message in the idle state.</td>
</tr>
<tr>
<td>Original cause:</td>
<td>Original flag, original count, and original cause:</td>
</tr>
<tr>
<td>Redirect flag:, Redirect count, Redirect cause:</td>
<td>Redirect flag, redirect count, and redirect cause:</td>
</tr>
<tr>
<td>CRI service enabled, release this call on CFO idle state</td>
<td>Incoming call barring was enabled, and the CFO released the call in the idle state.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Failed to find CFU voice entity in CFO idle state</td>
<td>The CFO failed to find a voice entity for CFU in the idle state.</td>
</tr>
<tr>
<td>Find CFU voice entity successfully in CFO idle state.</td>
<td>The CFO found a voice entity for CFU in the idle state.</td>
</tr>
<tr>
<td>Caller number: Forwarding number:</td>
<td>Calling number and called number:</td>
</tr>
<tr>
<td>Go to the unsupported process on CFO idle state</td>
<td>The CFO in the idle state went into local processing.</td>
</tr>
<tr>
<td>Send service CFU successfully, state changed from idle to waiting ack.</td>
<td>A CFU SERVICE message was successfully sent, and the state changed from idle to wait-for-ack.</td>
</tr>
<tr>
<td>Registed CF service mismatched on CFO wait release state</td>
<td>The RELEASE message received by the CFO in the waiting-for-release state mismatched the registered CF service.</td>
</tr>
<tr>
<td>Go to the unsupported process on CFO wait release state.</td>
<td>The CFO in the waiting-for-release state went into the processing of an unsupported condition.</td>
</tr>
<tr>
<td>Go to the unsupported process on CFO wait service ack state</td>
<td>The CFO in the waiting-for-ack state went into the processing of an unsupported condition.</td>
</tr>
<tr>
<td>Failed to process unsupported condition and release call</td>
<td>Local processing failed, and the call was released.</td>
</tr>
<tr>
<td>Finish the unsupported process on CFO wait ack state</td>
<td>The CFO in the waiting-for-ack state finished local processing.</td>
</tr>
<tr>
<td>No Number to setup call in CFR service process</td>
<td>The CFR had no number to continue call origination.</td>
</tr>
<tr>
<td>CFR has setup a new call in CFR service process</td>
<td>The CFR already originated a new call.</td>
</tr>
<tr>
<td>Failed to setup new call by all numbers in CFR service process</td>
<td>Call forwarding failed to originate a call after trying all numbers in multi-address processing.</td>
</tr>
<tr>
<td>No number to setup call in CFR fail message process</td>
<td>The CFR had no number to continue to originate a new call when receiving a FAIL message.</td>
</tr>
<tr>
<td>CFR has setup a new call in CFR fail message process</td>
<td>The CFR already originated a new call when receiving a FAIL message.</td>
</tr>
<tr>
<td>Failed to setup new call by all numbers in CFR fail message process</td>
<td>The CFR failed to set up a new call using all numbers after receiving a FAIL message.</td>
</tr>
<tr>
<td>Has same number in adding new address process</td>
<td>The number already existed when a new address was added.</td>
</tr>
<tr>
<td>Add the number successfully in adding new address process</td>
<td>A number was successfully added.</td>
</tr>
<tr>
<td>CF address table is full in adding new address process</td>
<td>The forwarded-to number list was already full when a new address was added.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td>Send CF service to CMC process:</td>
<td>The CFO sent a SERVICE message.</td>
</tr>
<tr>
<td>Address type:</td>
<td>Address type:</td>
</tr>
<tr>
<td>Forwarding number</td>
<td>Forwarding number:</td>
</tr>
<tr>
<td>IP addr:</td>
<td>IP address:</td>
</tr>
<tr>
<td>Port:</td>
<td>Port:</td>
</tr>
<tr>
<td>Redirection reason</td>
<td>Redirection cause:</td>
</tr>
<tr>
<td>Redirection count</td>
<td>Redirection count:</td>
</tr>
<tr>
<td>Original direction reason</td>
<td>Original redirection cause:</td>
</tr>
<tr>
<td>Original direction count</td>
<td>Original redirection count:</td>
</tr>
<tr>
<td>Forward number count</td>
<td>Forwarded-to number count:</td>
</tr>
<tr>
<td>Has disconnected the media in CFO unsupported process.</td>
<td>The CFR disconnected the media in processing an unsupported condition.</td>
</tr>
<tr>
<td>After predict in unsupported process:</td>
<td>After local processing:</td>
</tr>
<tr>
<td>Caller number:</td>
<td>Calling number and called number:</td>
</tr>
<tr>
<td>Called number:</td>
<td>Calling number and called number after number substitution:</td>
</tr>
<tr>
<td>Subst caller number:</td>
<td>Calling number and called number after number substitution:</td>
</tr>
<tr>
<td>Subst called number:</td>
<td>Calling number and called number after number substitution:</td>
</tr>
<tr>
<td>leg brother ifindex:, leg brother ip address:</td>
<td>Subscriber line index and IP address of a brother leg:</td>
</tr>
<tr>
<td>Redirect exceed the max limit in unsupported process</td>
<td>The number of redirections in local processing conditions exceeded the limit.</td>
</tr>
<tr>
<td>Get user information process:&quot;</td>
<td>The following user information was obtained:</td>
</tr>
<tr>
<td>Forward ip address is , port is ,</td>
<td>Forwarded-to IP address and port:</td>
</tr>
<tr>
<td>Has regist CF service and will process the CF service</td>
<td>The registered CF service was to be processed.</td>
</tr>
<tr>
<td>The forward information doesn’t match voice entity in setup new call.&quot;</td>
<td>The forward information mismatched the voice entity that originated the new call.</td>
</tr>
<tr>
<td>New call IP address:, IP address:,</td>
<td>IP address of a new call and IP address of a message:</td>
</tr>
<tr>
<td>Into the CFR module</td>
<td>The processing of the CFR module started.</td>
</tr>
<tr>
<td>CF CCB is null in CMC CCB</td>
<td>The CF CCB was null.</td>
</tr>
<tr>
<td>Receive service message with only one address in interface process</td>
<td>A SERVICE message with only one address was received in interface processing.</td>
</tr>
<tr>
<td>Receive address and create CFR CCB successfully in interface process</td>
<td>Multiple addresses were received, and the CF CCB was successfully created in interface processing.</td>
</tr>
<tr>
<td>Go to the receiver idle state in interface process</td>
<td>The processing of the CFR’s idle state started.</td>
</tr>
<tr>
<td>Go to the receiver pending state in interface process</td>
<td>The processing of the CFR’s pending state started.</td>
</tr>
<tr>
<td>CFR service end,return value [], service flag []</td>
<td>The processing of the CFR ended. The return value was [ ] and the return service flag was [ ].</td>
</tr>
<tr>
<td>Into the CFO module</td>
<td>The processing of the CFO module started.</td>
</tr>
<tr>
<td>The leg is wrong when receive setup message in interface process</td>
<td>The input leg was incorrect in interface processing.</td>
</tr>
</tbody>
</table>
Create CFO successfully in interface process: A CFO module was successfully created in interface processing.

Go to the originator idle state in interface process: The processing of the CFO’s idle state started.

Go to the originator wait release state in interface process: The processing of the CFR’s waiting-for-release state started.

Go to the originator wait service ack state in interface process: The processing of the CFO’s waiting-for-ack state started.

CFO service end, return value [], service flag []: The processing of the CFO ended. The return value was [] and the return service flag was [].

Table 116 describes output fields and messages for the `debugging voice ss cf timer` command.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Timer created, TimerId: TIMER_SS_CF_ORIGINATOR_WAITACK, Timer length: 5000</td>
<td>A 5-second timer was created for the CFO to wait for an ACK message.</td>
</tr>
<tr>
<td>Timer deleted, TimerId: TIMER_SS_CF_ORIGINATOR_WAITACK</td>
<td>The timer was deleted.</td>
</tr>
<tr>
<td>Timer timeout, TimerId: TIMER_SS_CF_ORIGINATOR_WAITACK</td>
<td>The timer expired.</td>
</tr>
</tbody>
</table>

Examples

# Enable debugging for call forwarding events on the router. Output similar to the following example is generated when a telephone (100) attempts to call the number 3000 under these conditions:

- Call forwarding busy is configured for the called party (3000).
- The called party (3000) is busy.

    <RouterB> debug voice ss cf event
    SS CF event debugging switch is on
    <RouterB>

    // The CF CCB was successfully deleted.

    // A RELEASE message was sent to the CMC.

The output similar to the following example was created when a telephone (100) attempts to call the number 3000 for which call forwarding unconditional is configured:

# Enable debugging for call forwarding finite state machines on the router.

    <RouterB> debug voice ss cf fsm
    SS CF FSM debugging switch is on
    <RouterB>

    *May 10 14:36:16:422 2007 RouterB SS/7/VOICE:
CF_FSM: [0x00690001] CFO state changed from wait release to wait service ack.

// The state of the CFO changed from waiting-for-release to waiting-for-service-ack.

# Enable debugging for call forwarding information on the router.
<RouterB> debug voice ss cf info
SS CF information debugging switch is on
<RouterB>

*May 10 14:54:53:796 2007 RouterB SS/7/VOICE:
  CF_INFO: [0x00770000] Has register CF service and will process the CF service.
  // The call forwarding service was processed.

*May 10 14:54:53:796 2007 RouterB SS/7/VOICE:
  CF_INFO: [0x00770000] Into the CFO module.
  // The processing of the CFO module started.

*May 10 14:54:53:796 2007 RouterB SS/7/VOICE:
  CF_INFO: [0x00770000] CFO CCB is null in CMC CCB.
  // The CF CCB was null.

*May 10 14:54:53:796 2007 RouterB SS/7/VOICE:
  CF_INFO: [0x00770000] Create CFO CCB successfully in interface process.
  // The CF CCB was successfully created.

*May 10 14:54:53:796 2007 RouterB SS/7/VOICE:
  CF_INFO: [0x00770000] Go to the originator idle state in interface process.
  // The processing of the idle state started.

*May 10 14:54:53:797 2007 RouterB SS/7/VOICE:
  CF_INFO: [0x00770000]
  Setup message information in CFO idle state:
  Original flag:0, Original count:0, Original cause:0
  Redirect flag:0, Redirect count:0, Redirect cause:0
  // The following information was read from the SETUP message: The original flag and redirect flag are invalid; the original cause and redirect cause are unknown.

*May 10 14:54:53:797 2007 RouterB SS/7/VOICE:
  CF_INFO: [0x00770000] CFO service end, return value [4], service flag [2].
  // The processing of the CFO ended.

*May 10 14:54:53:798 2007 RouterB SS/7/VOICE:
  CF_INFO: [0x00770001] Into the CFO module.
  // The processing of the CFO module started.

*May 10 14:54:53:798 2007 RouterB SS/7/VOICE:
  CF_INFO: [0x00770001] Go to the originator wait release state in interface process.
The processing of the CFO’s waiting-for-release state started.

*May 10 14:54:53:943 2007 RouterB SS/7/VOICE:
  CF_INFO: [0x00770000] Caller number: 1000, called number: 5000 in finding voice entity process.

The calling number is 1000, and the forwarded-to number is 5000.

*May 10 14:54:54:44 2007 RouterB SS/7/VOICE:
  CF_INFO: [0xffffffff]
  Get user information process:
  Forward ip address is 3.1.1.49, port is 5060

The forward IP address is 3.1.1.49, and the port number is 5060.

*May 10 14:54:54:144 2007 RouterB SS/7/VOICE:
  CF_INFO: [0x00770000]
  Send CF service to CMC module.
  Address type: 1
  Forward number: 5000
  IP address: 3.1.1.49
  Port: 5060
  Redirection reason 1
  Redirection count 1
  Original direction reason 0
  Original direction count 0
  Forward number count 1

A SERVICE message was sent.

- The forwarded-to number was 5000.
- The forward IP address was 3.1.1.49.
- The port number was 5060.
- The redirection cause was user busy.
- The original cause was unknown.
- The original count was 0.
- The count of forwarded-to numbers was 1.

*May 10 14:54:54:296 2007 RouterB SS/7/VOICE:
  CF_INFO: [0x00770001] CFO service end, return value [0], service flag [0].

The CFO service ended.

*May 10 14:54:54:396 2007 RouterB SS/7/VOICE:
  CF_INFO: [0x00770000] Go to the originator wait service ack state in interface process.

The processing of the CFO module started.

*May 10 14:54:54:648 2007 RouterB SS/7/VOICE:
  CF_INFO: [0x00770000] Go to the originator wait service ack state in interface process.

The processing of the CFO’s waiting-for-ack state started.
*May 10 14:54:54:799 2007 RouterB SS/7/VOICE:*

`CF_INFO: [0x00770000] CFO service end, return value [0], service flag [2].`

// The CF service ended.

* Enable debugging for call forwarding timers on the router.

```bash
<RouterB> debug voice ss cf timer
SS CF timer debugging switch is on
<RouterB>
```

*May 10 15:38:12:85 2007 RouterB SS/7/VOICE:*

`CF_TMR: [0x00780001] Timer created, Timer ID: TIMER_SS_CF_ORIGINATOR_WAITACK, Timer length: 5000.`

// A 5000-millisecond timer was created for the CFO to wait for an ACK message.

*May 10 15:38:12:85 2007 RouterB SS/7/VOICE:*

`CF_TMR: [0x00780000] Timer deleted, Timer ID: TIMER_SS_CF_ORIGINATOR_WAITACK.`

// The timer was deleted.

**debugging voice ss ch**

Use `debugging voice ss ch` to enable call hold debugging.

Use `undo debugging voice ss ch` to disable call hold debugging.

**Syntax**

```bash
debugging voice ss ch { all | error | event | fsm | info | timer }
```

```bash
undo debugging voice ss ch { all | error | event | fsm | info | timer }
```

**Default**

Call hold debugging is disabled.

**Views**

User view

**Default command level**

2: System level

**Parameters**

- **all**: Specifies all types of debugging for call hold.
- **error**: Specifies error debugging.
- **event**: Specifies event debugging.
- **fsm**: Specifies finite state machine debugging.
- **info**: Specifies information debugging.
- **timer**: Specifies timer debugging.

**Usage guidelines**

Table 117 describes output fields and messages for the `debugging voice ss ch error` command.
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Received service message xxx in xxx state, return fail message to SA.</td>
<td>A SERVICE message was received, and a FAIL message was returned to the SA module.</td>
</tr>
<tr>
<td>Received error service acknowledge message: XXX in state: XXX.</td>
<td>An invalid SERVICE ACK message was received in xxx state.</td>
</tr>
<tr>
<td>Received error service message XXX in XXX state</td>
<td>An invalid SERVICE message was received in xxx state.</td>
</tr>
<tr>
<td>Failed to get the held talk path for unholding in XXX state.</td>
<td>Call hold failed to resume the held talk path in xxx state.</td>
</tr>
<tr>
<td>Failed to find the new call talk path in act-proc in XXX state.</td>
<td>Call hold failed to find a new talk path during ACT processing in xxx state.</td>
</tr>
<tr>
<td>Failed to get the held talk path after fail to sent setup message.</td>
<td>Call hold failed to obtain the held talk path after it failed to send a SETUP message.</td>
</tr>
<tr>
<td>Failed to get CH config on voice-interface in XXX state.</td>
<td>Call hold failed to obtain CH configuration in xxx state.</td>
</tr>
<tr>
<td>Failed to get ct-enable config on voice-interface in XXX state.</td>
<td>Call hold failed to obtain CT configuration in xxx state.</td>
</tr>
<tr>
<td>Failed to disconnect the local media in XXX state.</td>
<td>Call hold failed to disconnect the local media in xxx state.</td>
</tr>
<tr>
<td>Failed to connect the local media in XXX state.</td>
<td>Call hold failed to connect the local media in xxx state.</td>
</tr>
<tr>
<td>Failed to restore CCB for hookflash handling in XXX state.</td>
<td>Call hold failed to update the CCB information in processing the HOOKFLASH message.</td>
</tr>
<tr>
<td>In act-proc failed to find the CCB.</td>
<td>Call hold failed to find the CCB during ACT processing.</td>
</tr>
<tr>
<td>Failed to update CCB in XXX state.</td>
<td>Call hold failed to update the CCB in xxx state.</td>
</tr>
<tr>
<td>Failed to update CCB for new call’s error handling.</td>
<td>Call hold failed to update the CCB in processing errors that occurred during new call setup.</td>
</tr>
<tr>
<td>Failed to store CCB in act-proc in XXX state.</td>
<td>Call hold failed to update the CCB during ACT processing.</td>
</tr>
<tr>
<td>Failed to restore CCB after fail to sent setup message.</td>
<td>Call hold failed to update the CCB after it failed to send a SETUP message.</td>
</tr>
<tr>
<td>Invalid CMC CCB in switch leg type.</td>
<td>The CMC CCB was null when the function of switching the leg type was invoked.</td>
</tr>
<tr>
<td>CH Service over failed to use a null service CCB.</td>
<td>The service CCB was null when the function of ending the CH service was invoked.</td>
</tr>
<tr>
<td>Failed to enter the new call service in XXX state.</td>
<td>Call hold failed to set up a new call.</td>
</tr>
<tr>
<td>Failed to start new call after receive ACCP_ALERTING message in XXX state.</td>
<td>Call hold failed to set up a new call after receiving an ACCP_ALERTING message.</td>
</tr>
<tr>
<td>Failed to start new call after receive ACCP_CONNECT message in XXX state.</td>
<td>Call hold failed to set up a new call after receiving an ACCP_CONNECT message.</td>
</tr>
<tr>
<td>Failed to unhold on local in XXX state.</td>
<td>Call hold failed to resume the held call.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Failed to create XXX timer when enter new call operation.</td>
<td>Call hold failed to create the xxx timer in the call setup process.</td>
</tr>
<tr>
<td>Failed to create XXX timer in XXX state.</td>
<td>Call hold failed to create the xxx timer in xxx state.</td>
</tr>
<tr>
<td>Failed to get call information table, when dial a new number.</td>
<td>Call hold failed to obtain the call information table after the forwarded-to number was dialed.</td>
</tr>
<tr>
<td>Failed to close DTMF, after dial the number in CH.</td>
<td>Call hold failed to disable DTMF detection after the forwarded-to number was dialed.</td>
</tr>
<tr>
<td>Failed to close DTMF, when the new call is overtime.</td>
<td>Call hold failed to disable DTMF detection after sending the forwarded-to number timed out.</td>
</tr>
<tr>
<td>Failed to close DTMF, when can not matching the number dialed.</td>
<td>Call hold failed to disable DTMF detection when no voice entity matched the dialed number.</td>
</tr>
<tr>
<td>Failed to open DTMF detect, when enter a new call.</td>
<td>Call hold failed to enable DTMF detection during the setup of a new call.</td>
</tr>
<tr>
<td>Failed to play the busy tone, when can not matching the number dialed.</td>
<td>Call hold failed to play busy tones when no voice entity matched the dialed number.</td>
</tr>
<tr>
<td>Failed to play busy tone, when the new callee on hook in XXX state.</td>
<td>Call hold failed to play busy tones after the new callee hangs up.</td>
</tr>
<tr>
<td>Failed to play busy tone for new call’s error handling.</td>
<td>Call hold failed to play busy tones in handling errors occurring to the new call.</td>
</tr>
<tr>
<td>Failed to play busy tone after fail to sent setup message.</td>
<td>Call hold failed to play busy tones after failure to send a SETUP message.</td>
</tr>
<tr>
<td>Failed to play busy tone when can not match entity for new call.</td>
<td>Call hold failed to play busy tones when no voice entity could be matched for the new call.</td>
</tr>
<tr>
<td>Failed to stop the dialing tone, when the new call is overtime.</td>
<td>Call hold failed to stop playing dial tones after the dialing timer for the new call expired.</td>
</tr>
<tr>
<td>Failed to stop the dialing tone, when dial a new number.</td>
<td>Call hold failed to stop playing dial tones after the new number was dialed.</td>
</tr>
<tr>
<td>Failed to get the service’s call role in XXX state.</td>
<td>Call hold failed to obtain the service role of the call leg.</td>
</tr>
<tr>
<td>Failed to get brother leg in state: XXX when create XXX timer.</td>
<td>Call hold failed to obtain a brother leg when the xxx timer was created.</td>
</tr>
<tr>
<td>Failed to get the new setup leg for switch leg in XXX state.</td>
<td>Call hold failed to obtain a new call leg during leg switching.</td>
</tr>
<tr>
<td>Failed to get the hookflash leg for switch call leg in XXX state.</td>
<td>Call hold failed to obtain a hookflash leg during leg switching.</td>
</tr>
<tr>
<td>Failed to get the new setup call leg, at a new call’s error handling.</td>
<td>Call hold failed to obtain a new call leg in handling errors occurring to the new call.</td>
</tr>
<tr>
<td>Failed to get the hookflash call leg for new call’s error handling.</td>
<td>Call hold failed to obtain a hookflash leg in handling errors occurring to the new call.</td>
</tr>
<tr>
<td>Failed to get the hookflash leg for unholding in XXX state.</td>
<td>Call hold failed to obtain a hookflash leg in resuming the call.</td>
</tr>
<tr>
<td>Failed to get the hookflash call leg in act-proc in XXX state.</td>
<td>Call hold failed to obtain a hookflash leg in ACT processing.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Failed to get the hookflash call leg to receive number.</td>
<td>No call leg could be found to receive the number.</td>
</tr>
<tr>
<td>Failed to get the new setup call leg for hookflash handling in XXX state.</td>
<td>Call hold failed to obtain a new call leg in handling hookflash.</td>
</tr>
<tr>
<td>Failed to get the hookflash call leg for hookflash handling in XXX state.</td>
<td>No hookflash leg was found in handling hookflash.</td>
</tr>
<tr>
<td>Failed to switch the hookflash call leg for new call's error handling.</td>
<td>Call hold failed to switch call legs in handling errors occurring to the new call.</td>
</tr>
<tr>
<td>Failed to switch the new setup leg in XXX state.</td>
<td>Call hold failed to switch the new call leg in xxx state.</td>
</tr>
<tr>
<td>Failed to switch leg type in act-proc in XXX state.</td>
<td>Call hold failed to switch the leg type in ACT processing.</td>
</tr>
<tr>
<td>Failed to create the new call leg in act-proc in XXX state.</td>
<td>Call hold failed to create a new call leg in ACT processing.</td>
</tr>
<tr>
<td>Can not match the entity, when a new call starts.</td>
<td>No voice entity was matched when the new call started.</td>
</tr>
<tr>
<td>Failed to fill the new call information in act-proc in XXX state.</td>
<td>Call hold failed to fill in new call information in ACT processing.</td>
</tr>
<tr>
<td>Failed to delete new call leg.</td>
<td>Call hold failed to delete the new call leg.</td>
</tr>
<tr>
<td>Failed to create CH Model.</td>
<td>Call hold failed to create the CH CCB.</td>
</tr>
<tr>
<td>Invalid new call info tab index in XXX state.</td>
<td>The new call information table index was invalid.</td>
</tr>
<tr>
<td>Get Voice Entity For Call Error in XXX state.</td>
<td>Call hold failed to obtain a voice entity.</td>
</tr>
<tr>
<td>Failed to fill new call information in new talk path in XXX state.</td>
<td>Call hold failed to fill the new call information in the talk path.</td>
</tr>
<tr>
<td>Find talk path error in switch leg type.</td>
<td>Call hold failed to find a talk path in switching the leg type.</td>
</tr>
<tr>
<td>CH Service Interface failed to use a null leg.</td>
<td>The leg of the CH service interface function was null.</td>
</tr>
<tr>
<td>CH Service Interface failed to find the CCB.</td>
<td>The CH service interface function failed to find the CCB.</td>
</tr>
<tr>
<td>CH service failed to start.</td>
<td>The CH service failed to start.</td>
</tr>
<tr>
<td>Wrong CH service state.</td>
<td>The internal state of the CH service was invalid.</td>
</tr>
<tr>
<td>Incorrect message to start the CH service.</td>
<td>The CH service interface function received an incorrect message.</td>
</tr>
<tr>
<td>The CH service started already can not start new one.</td>
<td>The CH CCB was not idle and no new CH operation could be started.</td>
</tr>
<tr>
<td>Wrong call state, can not start CH operation.</td>
<td>An internal error occurred to the CH service and no CH operation could be started.</td>
</tr>
<tr>
<td>CH operation has been deleted. CH operation finished exceptional.</td>
<td>The CH service aborted.</td>
</tr>
<tr>
<td>Failed to find local active leg when process release in %s state.</td>
<td>The local active leg (corresponding to the remote leg) could not be found.</td>
</tr>
<tr>
<td>Failed to find the local inactive leg in %s state.</td>
<td>The local inactive leg (corresponding to the remote held leg) could not be found.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Failed to find the remote inactive leg in %s state.</td>
<td>The remote inactive leg (corresponding to the leg held with the call hold originator) could not be found.</td>
</tr>
<tr>
<td>Failed to find the remote active leg in %s state.</td>
<td>The remote active leg (corresponding to the leg in conversation with the call hold originator) could not be found.</td>
</tr>
<tr>
<td>Failed to find the leg's talk path in %s state.</td>
<td>The talk path of the leg could not be found.</td>
</tr>
<tr>
<td>Failed to find the remote holding leg in %s state.</td>
<td>The remote held leg (corresponding to the leg held with the call hold originator) could not be found.</td>
</tr>
<tr>
<td>Failed to find remote holding path in %s state.</td>
<td>The held talk path could not be found.</td>
</tr>
<tr>
<td>Service role of input leg is invalid in %s state.</td>
<td>The service role (local active leg, local inactive leg, remote active leg, or remote inactive leg) of the input leg could not be learned.</td>
</tr>
<tr>
<td>Failed to create hold timer and unhold timer in %s state.</td>
<td>Call hold failed to create a call hold timer and a call resume timer.</td>
</tr>
<tr>
<td>Failed to create hold timer when deal with hookflash in %s state.</td>
<td>Call hold failed to create a call hold timer.</td>
</tr>
<tr>
<td>Failed to delete %s timer.</td>
<td>Call hold failed to delete the call hold timer.</td>
</tr>
<tr>
<td>Failed to create %s timer.</td>
<td>Call hold failed to create a timer.</td>
</tr>
<tr>
<td>Failed to create MCH unhold timer when process hookflash in %s state.</td>
<td>Call hold failed to create a call resume timer.</td>
</tr>
<tr>
<td>Failed to switch leg in %s state.</td>
<td>Call hold failed to switch the leg in xxx state.</td>
</tr>
<tr>
<td>Failed to disconnect media %s state.</td>
<td>Call hold failed to disconnect the media in xxx state.</td>
</tr>
<tr>
<td>Failed to connect media in %s state.</td>
<td>Call hold failed to connect the media in xxx state.</td>
</tr>
<tr>
<td>Failed to restore CCB info in %s state.</td>
<td>Call hold failed to update the CCB in xxx state.</td>
</tr>
<tr>
<td>Local active leg’s IF index is invalid when process release in XXX state.</td>
<td>The interface index of the local active leg was invalid.</td>
</tr>
<tr>
<td>Local active leg can not be retrieved when process release in %s state.</td>
<td>The local active leg could not be found.</td>
</tr>
<tr>
<td>The last talk path index is invalid.</td>
<td>The index of the last talk path was invalid.</td>
</tr>
<tr>
<td>The founded talk path is invalid.</td>
<td>The return talk path was invalid.</td>
</tr>
<tr>
<td>Failed to find the local leg in the newest talk path.</td>
<td>The hookflash controlling leg could not be found in the newest talk path.</td>
</tr>
<tr>
<td>The leg type of input leg is illegal when process release in %s state.</td>
<td>The leg type was invalid, and it could be input or output only.</td>
</tr>
<tr>
<td>The call type of input leg is illegal when process release in %s state.</td>
<td>The call type was invalid, and it could be local call or IP call only.</td>
</tr>
<tr>
<td>Failed to play tone when remote active leg released in %s state.</td>
<td>Call hold failed to play busy tones after the remote active leg released the call.</td>
</tr>
<tr>
<td>Failed to play tone when received CUH-FAIL in %s state.</td>
<td>Call hold failed to play busy tones after a CUH_FAIL message was received.</td>
</tr>
<tr>
<td>Failed to play busy tone when remote inactive leg released in %s state.</td>
<td>Call hold failed to play busy tones after the remote held leg released the call.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td>Failed to play busy tone after fail to switch leg in %s state.</td>
<td>Call hold failed to play busy tones after failure to switch the leg.</td>
</tr>
<tr>
<td>Failed to play busy tone when receive timeout message in %s state.</td>
<td>Call hold failed to play busy tones after a TIMEOUT message was received.</td>
</tr>
<tr>
<td>Receive wrong type CMC INFORMATION message in %s state.</td>
<td>An invalid type of information message was received.</td>
</tr>
<tr>
<td>Receive wrong service ack message in %s state.</td>
<td>An incorrect SERVICE ACK message was received.</td>
</tr>
<tr>
<td>Receive wrong message in %s state.</td>
<td>An incorrect message was received.</td>
</tr>
<tr>
<td>Received wrong ack of SERVICE (CH) when process in %s state.</td>
<td>The received SERVICE (CH) ACK message was invalid.</td>
</tr>
<tr>
<td>Receive wrong timeout message in %s state.</td>
<td>The received TIMEOUT message was invalid.</td>
</tr>
<tr>
<td>Receive wrong ack of service(CUH) in %s state.</td>
<td>The received CUH ACK message was invalid.</td>
</tr>
<tr>
<td>Receive wrong CMC message in %s state.</td>
<td>The received CMC message was invalid.</td>
</tr>
<tr>
<td>MCH service already started.</td>
<td>Starting the MCH service failed because the MCH service was already started.</td>
</tr>
<tr>
<td>Failed to allocate memory for MCH CCB.</td>
<td>The FEATURE service failed to allocate memory to the MCH CCB.</td>
</tr>
<tr>
<td>Failed to delete hold timer when delete MCH CCB</td>
<td>The FEATURE service failed to delete the call hold timer after the MCH CCB was deleted.</td>
</tr>
<tr>
<td>Failed to delete unhold timer when delete MCH CCB</td>
<td>The FEATURE service failed to delete the call resume timer after the MCH CCB was deleted.</td>
</tr>
<tr>
<td>Failed to get the service switch.</td>
<td>Call hold failed to get the service configured on the current voice interface.</td>
</tr>
<tr>
<td>Can not start MCH with one call.</td>
<td>The state was incorrect, and the MCH could not be started because there was only one call.</td>
</tr>
<tr>
<td>Incorrect message type in service start process.</td>
<td>The MCH message interface function received an incorrect type of message.</td>
</tr>
<tr>
<td>MCH state is illegal.</td>
<td>The MCH state was invalid.</td>
</tr>
</tbody>
</table>

Table 118 describes output fields and messages for the debugging voice ss ch event command.

Table 118 Output from the debugging voice ss ch event command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Received message [XXX] in [XXX] state.</td>
<td>An xxx message was received in xxx state.</td>
</tr>
<tr>
<td>CH Service Interface received message [XXX] from SPL [XXX]</td>
<td>The CH interface function received an xxx message from xxx.</td>
</tr>
<tr>
<td>Send XXX message to XXX module.</td>
<td>An xxx message was sent to the xxx module.</td>
</tr>
<tr>
<td>Send XXX message to XXX module in XXX state.</td>
<td>An xxx message was sent to the xxx module in xxx state.</td>
</tr>
<tr>
<td>&quot;FAX is on, can't exchange talk path in XXX state.&quot;</td>
<td>Fax was on, and no hookflash operation was allowed.</td>
</tr>
</tbody>
</table>
Table 119 describes output fields and messages for the **debugging voice ss ch fsm** command.

### Table 119 Output from the debugging voice ss ch fsm command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CH state move to XXX from XXX</td>
<td>The CH state changed from xxx to xxx.</td>
</tr>
<tr>
<td>The service state of MCH changed from XXX to XXX.</td>
<td>The MCH state changed from xxx to xxx.</td>
</tr>
</tbody>
</table>

Table 120 describes output fields and messages for the **debugging voice ss ch info** command.

### Table 120 Output from the debugging voice ss ch info command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Successfully hold on local in XXX state.</td>
<td>The call was successfully held at the local end.</td>
</tr>
<tr>
<td>Successfully unhold on local in XXX state.</td>
<td>The call was successfully resumed at the local end.</td>
</tr>
<tr>
<td>Setup a new call when call hold.</td>
<td>A new call was set up after the call was held.</td>
</tr>
<tr>
<td>Start a new CH service successfully.</td>
<td>The CH service was successfully started.</td>
</tr>
<tr>
<td>CH service is finished at cause XXX.</td>
<td>The CH service aborted for the reason of xxx.</td>
</tr>
<tr>
<td>Send the service message successfully in XXX state.</td>
<td>A SERVICE message was successfully sent in xxx state.</td>
</tr>
<tr>
<td>Successfully send service acknowledge in XXX state.</td>
<td>A SERVICE ACK message was successfully sent in xxx state.</td>
</tr>
<tr>
<td>Hookflash message is discarded in XXX state from call-holdee.</td>
<td>A HOOKFLASH message from the held party was discarded.</td>
</tr>
<tr>
<td>Deal with the saved hookflash in XXX state.</td>
<td>The cached HOOKFLASH message was processed in xxx state.</td>
</tr>
<tr>
<td>Discard CMC_INFORMATION message in XXX state.</td>
<td>A CMC_INFORMATION message was discarded.</td>
</tr>
<tr>
<td>Received alerting message again in XXX state.</td>
<td>A CMC_ALERTING message was received again.</td>
</tr>
<tr>
<td>Can not start CT service for ct-disable config in XXX state.</td>
<td>The CT service could not be started because it was not configured.</td>
</tr>
<tr>
<td>Start DTMF detect successfully, when enter a new call.</td>
<td>DTMF detection was successfully enabled for a new call.</td>
</tr>
<tr>
<td>Close DTMF successfully, when dial the number.</td>
<td>DTMF detection was successfully disabled when the dialed number was completely matched.</td>
</tr>
<tr>
<td>Close DTMF Detect successfully, when can not matching the number dialed.</td>
<td>DTMF detection was successfully disabled when the dialed number was not matched.</td>
</tr>
<tr>
<td>Close DTMF Detect successfully, when the new call is overtime.</td>
<td>DTMF detection was successfully disabled when the dialing timer for a new call expired.</td>
</tr>
<tr>
<td>Will enter BCT service in XXX state.</td>
<td>The BCT service was to start.</td>
</tr>
<tr>
<td>To enter BCT service in XXX state.</td>
<td>The BCT service was to start.</td>
</tr>
<tr>
<td>To enter ACT service in XXX state.</td>
<td>The ACT service was to start.</td>
</tr>
<tr>
<td>To enter MCH service.</td>
<td>The MCH service was to start.</td>
</tr>
<tr>
<td>Enter the new call service.</td>
<td>A new call was set up.</td>
</tr>
<tr>
<td>Enter the hold service.</td>
<td>The call was held.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td>Enter the unhold service.</td>
<td>The call was resumed.</td>
</tr>
<tr>
<td>Finish handling, release new call info Table</td>
<td>The recorded new call information table was released when the service CCB was deleted.</td>
</tr>
<tr>
<td>Release new call info Table in XXX state.</td>
<td>The new call information table was released.</td>
</tr>
<tr>
<td>Switch leg type successfully.</td>
<td>The leg type was successfully switched.</td>
</tr>
<tr>
<td>Succeed to create XXX timer when enter new call operation.</td>
<td>The xxx timer was successfully created in handling the setup of the new call.</td>
</tr>
<tr>
<td>Find voice entity result fail and no part match exist in XXX state.</td>
<td>No voice entity was matched.</td>
</tr>
<tr>
<td>Find voice entity result fail and default handle in XXX state.</td>
<td>The voice entity was unidentifiable.</td>
</tr>
<tr>
<td>The call hold originator interface index is XXX.</td>
<td>The voice interface of the call hold originator was xxx.</td>
</tr>
<tr>
<td>Receive release message from remote inactive leg in XXX state.</td>
<td>The remote inactive leg was released.</td>
</tr>
<tr>
<td>Receive release message from remote active leg in XXX state.</td>
<td>The remote active leg was released.</td>
</tr>
<tr>
<td>Receive release from local active leg in XXX state.</td>
<td>The local active leg was released.</td>
</tr>
<tr>
<td>The second call type is IP call.</td>
<td>The second call is an IP call.</td>
</tr>
<tr>
<td>The second call type is local call.</td>
<td>The second call is a local call.</td>
</tr>
<tr>
<td>Can have transfer service.</td>
<td>The MCH service was able to be transferred to the call transfer service.</td>
</tr>
<tr>
<td>Transfer service is allowed.</td>
<td>The call transfer service was enabled on the voice interface.</td>
</tr>
<tr>
<td>Transfer service is not registered.</td>
<td>The call transfer service was not enabled on the voice interface.</td>
</tr>
<tr>
<td>Hand message to S_CT_O module successfully.</td>
<td>A message was successfully handed to the S_CT_O module for processing.</td>
</tr>
<tr>
<td>Hookflash message is discarded in [XXX] state from call-holdee.</td>
<td>A HOOKFLASH message from the held party was discarded.</td>
</tr>
<tr>
<td>Receive ack of CH-OK in XXX state.</td>
<td>A CH_OK message was received.</td>
</tr>
<tr>
<td>Receive ack of CH-FAIL in XXX state.</td>
<td>A CH_FAIL message was received.</td>
</tr>
<tr>
<td>Receive ack of CUH-OK in XXX state.</td>
<td>A CUH_OK message was received.</td>
</tr>
<tr>
<td>Receive ack of CUH-FAIL in XXX state.</td>
<td>A CUH_FAIL message was received.</td>
</tr>
<tr>
<td>Receive ack of CH-UNSUPPORT in XXX state.</td>
<td>A CH_UNSUPPORT message was received.</td>
</tr>
<tr>
<td>Receive ack of CUH-UNSUPPORT in XXX state.</td>
<td>A CUH_UNSUPPORT message was received.</td>
</tr>
<tr>
<td>Process stored hookflash after back to XXX state.</td>
<td>The cached HOOKFLASH message was processed.</td>
</tr>
<tr>
<td>Deal with the stored hookflash after back to XXX state.</td>
<td>The cached HOOKFLASH message was processed.</td>
</tr>
<tr>
<td>Restore CCB successfully in XXX state.</td>
<td>The CCB was successfully updated.</td>
</tr>
<tr>
<td>MCH CCB is created successfully.</td>
<td>The MCH CCB was successfully created.</td>
</tr>
</tbody>
</table>
Field | Description
--- | ---
MCH CCB is initialized successfully. | The MCH CCB was successfully initialized.
MCH CCB is deleted successfully. | The MCH CCB was successfully deleted.
The hookflash is counted successfully. | The hookflash was successfully cached.
The current state is XXX. | The current state of the MCH service is xxx.

Table 121 describes output fields and messages for the debugging voice ss ch timer command.

Table 121 Output from the debugging voice ss ch timer command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Timer created, TimerId : XXX, Timer length :XXX.*</td>
<td>The xxx timer with a length of xxx seconds was successfully created.</td>
</tr>
<tr>
<td>Timer deleted, TimerId : XXX.</td>
<td>The xxx timer was deleted.</td>
</tr>
<tr>
<td>Timer timeout, TimerId : XXX.</td>
<td>The xxx timer expired.</td>
</tr>
</tbody>
</table>

Examples

# Enable all types of debugging for call hold on Router A. The output similar to the following example was created when the following conditions exist:

- Call hold is enabled for both the calling party attached to Router A and called party attached to Router B.
- A call is established from Telephone A to Telephone B, and then the calling party (Telephone A) places the call on hold by pressing hookflash.

The output similar to the following messages was created when all types of debugging for call hold are enabled on Router A.

<RouterA> debugging voice ss ch all
SS CH all debugging switches are on
<RouterA>
*Nov 20 00:38:37:850 2005 RouterA SS/7/VOICE:
  CH_EVT: [0x00000000] CH Service Interface received message [CMC_INFORMATION] from SPL [SPL_DISCRIM_LGS].
  // A CMC_INFORMATION (HOOKFLASH) message was received.

*Nov 20 00:38:37:851 2005 RouterA SS/7/VOICE:
  CH_INFO: [0x00000000] Start a new CH service successfully.
  // The call hold service was successfully started.

*Nov 20 00:38:37:851 2005 RouterA SS/7/VOICE:
  CH_EVT: [0x00000000] Received message:[CMC_INFORMATION] in state[S_CH_IDLE].
  // The call hold module processed the CMC_INFORMATION (HOOKFLASH) message in the idle state.

*Nov 20 00:38:37:851 2005 RouterA SS/7/VOICE:
  CH_TMR: [0x00000000] Timer created, Timer ID: TIME_LEN_SS_CH_WAIT_SRVACK, Timer length: 20000.
  // A 20000-ms timer waiting for a call hold acknowledgment was created.
*Nov 20 00:38:37:851 2005 RouterA SS/7/VOICE:
CH_INFO: [0x00000000] Send the service message successfully in [S_CH_IDLE] state.
// A SERVICE (CH) message was sent successfully.

*Nov 20 00:38:37:851 2005 RouterA SS/7/VOICE:
CH_FSM: [0x00000000] CH state move to [S_CH_PRE] from [S_CH_IDLE].
// The state of the call hold module changed from idle to preparation.

*Nov 20 00:38:37:858 2005 RouterA SS/7/VOICE:
CH_EVT: [0x00000001] CH Service Interface received message [CMC_SERVICE_ACK] from SPL [SPL_DISCRIM_SIP].
// A SERVICE_ACK message was received.

*Nov 20 00:38:37:858 2005 RouterA SS/7/VOICE:
CH_EVT: [0x00000001] Receive CMC_SERVICE_ACK message from SA module in S_CH_PRE state.
// The call hold module processed the SERVICE_ACK message in the preparation state.

*Nov 20 00:38:37:858 2005 RouterA SS/7/VOICE:
CH_TMR: [0x00000001] Timer deleted, Timer ID: TIMER_SS_CH_WAIT_SRVCHACK.
// The timer waiting for a SERVICE(CH)_ACK message was deleted.

*Nov 20 00:38:38:55 2005 RouterA SS/7/VOICE:
CH_INFO: [0x00000000] Start DTMF detect successfully, when enter a new call.
// The call was held successfully. At this time, dialing tones were played to indicate that a new call can be originated, and the DTMF detection was enabled.

*Nov 20 00:38:38:156 2005 RouterA SS/7/VOICE:
CH_TMR: [0x00000000] Succeed to create [TIMER_SS_CH_WAIT_DIALNUM] timer when enter new call operation.
// A timer waiting for dialing the first digit was created.

*Nov 20 00:38:38:307 2005 RouterA SS/7/VOICE:
CH_FSM: [0x00000000] CH state move to [S_CH_NC] from [S_CH_PRE].
// The state of the call hold module changed from preparation to digit-collection.

*Nov 20 00:38:38:458 2005 RouterA SS/7/VOICE:
CH_INFO: [0xffffffff] Enter the new call service.
// A new call was to be originated.

# The output in the following messages was created when the calling party (Telephone A) places a new call to Telephone C (4000).

*Nov 20 01:32:15:241 2005 RouterA SS/7/VOICE:
CH_EVT: [0x00003000] CH Service Interface received message [CMC_INFORMATION] from SPL [SPL_DISCRIM_LGS].
*Nov 20 01:32:15:241 2005 RouterA SS/7/VOICE:
CH_EVT: [0x00030000] Receive CMC_INFORMATION message from SA module.

// DTMF digits were received.

*Nov 20 01:32:15:241 2005 RouterA SS/7/VOICE:

CH_EVT: [0x00030000] Receive CMC_INFORMATION message from SA module.

// The timer waiting for dialing the first digit was deleted.

*Nov 20 01:32:15:241 2005 RouterA SS/7/VOICE:

CH_EVT: [0x00030000] Receive CMC_INFORMATION message from SA module.

// The timer waiting for dialing the first digit was deleted.

*Nov 20 01:32:15:241 2005 RouterA SS/7/VOICE:

CH_EVT: [0x00030000] Receive CMC_INFORMATION message from SA module.

// The timer waiting for dialing the first digit was deleted.

*Nov 20 01:32:15:241 2005 RouterA SS/7/VOICE:

CH_EVT: [0x00030000] Receive CMC_INFORMATION message from SA module.

// The 10000-ms timer waiting for the next digit was started.

// The following repeated the above digit collection process until the number 4000 was collected and matched a voice entity.

*Nov 20 01:32:16:651 2005 RouterA SS/7/VOICE:

CH_EVT: [0x00030000] CH Service Interface received message [CMC_INFORMATION] from SPL [SPL_DISCRIM_LGS].

*Nov 20 01:32:16:651 2005 RouterA SS/7/VOICE:

CH_EVT: [0x00030000] Receive CMC_INFORMATION message from SA module.

*Nov 20 01:32:16:651 2005 RouterA SS/7/VOICE:

CH_EVT: [0x00030000] Receive CMC_INFORMATION message from SA module.

// The 10000-ms timer waiting for the next digit was started.

// The following repeated the above digit collection process until the number 4000 was collected and matched a voice entity.

*Nov 20 01:32:16:651 2005 RouterA SS/7/VOICE:

CH_EVT: [0x00030000] CH Service Interface received message [CMC_INFORMATION] from SPL [SPL_DISCRIM_LGS].

*Nov 20 01:32:16:651 2005 RouterA SS/7/VOICE:

CH_EVT: [0x00030000] Receive CMC_INFORMATION message from SA module.

*Nov 20 01:32:16:651 2005 RouterA SS/7/VOICE:

CH_EVT: [0x00030000] Receive CMC_INFORMATION message from SA module.

// The timer waiting for the next digit was deleted.

*Nov 20 01:32:16:651 2005 RouterA SS/7/VOICE:

CH_EVT: [0x00030000] CH Service Interface received message [CMC_INFORMATION] from SPL [SPL_DISCRIM_LGS].

*Nov 20 01:32:16:651 2005 RouterA SS/7/VOICE:

CH_EVT: [0x00030000] Receive CMC_INFORMATION message from SA module.

*Nov 20 01:32:16:651 2005 RouterA SS/7/VOICE:

CH_EVT: [0x00030000] Receive CMC_INFORMATION message from SA module.

// The timer waiting for the next digit was deleted.

*Nov 20 01:32:16:651 2005 RouterA SS/7/VOICE:

CH_EVT: [0x00030000] CH Service Interface received message [CMC_INFORMATION] from SPL [SPL_DISCRIM_LGS].

*Nov 20 01:32:16:651 2005 RouterA SS/7/VOICE:

CH_EVT: [0x00030000] Receive CMC_INFORMATION message from SA module.

*Nov 20 01:32:16:651 2005 RouterA SS/7/VOICE:

CH_EVT: [0x00030000] Receive CMC_INFORMATION message from SA module.

// The timer waiting for the next digit was deleted.

*Nov 20 01:32:16:651 2005 RouterA SS/7/VOICE:

CH_EVT: [0x00030000] CH Service Interface received message [CMC_INFORMATION] from SPL [SPL_DISCRIM_LGS].

*Nov 20 01:32:16:651 2005 RouterA SS/7/VOICE:

CH_EVT: [0x00030000] Receive CMC_INFORMATION message from SA module.

*Nov 20 01:32:16:651 2005 RouterA SS/7/VOICE:

CH_EVT: [0x00030000] Receive CMC_INFORMATION message from SA module.

// The timer waiting for the next digit was deleted.

*Nov 20 01:32:16:651 2005 RouterA SS/7/VOICE:

CH_EVT: [0x00030000] CH Service Interface received message [CMC_INFORMATION] from SPL [SPL_DISCRIM_LGS].

// The following repeated the above digit collection process until the number 4000 was collected and matched a voice entity.
CH_INFO: [0x00030000] Close DTMF successfully, when dial the number.
// The number matched a voice entity, and DTMF detection was disabled.

*Nov 20 01:32:17:492 2005 RouterA SS/7/VOICE:
  CH_INFO: [0x00030002] Setup a new call when call hold.
// A new call was originated while the first call was put on hold.

*Nov 20 01:32:17:492 2005 RouterA SS/7/VOICE:
  CH_FSM: [0x00030000] CH state move to [S_SCH_WA] from [S_CH_NC].
// The state of the call hold module changed from digit collection to call origination.

*Nov 20 01:32:17:557 2005 RouterA SS/7/VOICE:
  CH_EVT: [0x00030003] CH Service Interface received message [CMC_ALERTING] from SPL [SPL_DISCRIM_SIP].
*Nov 20 01:32:17:657 2005 RouterA SS/7/VOICE:
  CH_EVT: [0x00030003] Received message [CMC_ALERTING] in [S_SCH_WA] state.
// The ALERTING message was processed in the call origination state.

*Nov 20 01:32:17:808 2005 RouterA SS/7/VOICE:
  CH_FSM: [0x00030000] CH state move to [S_SCH_WC] from [S_SCH_WA].
// The state of the call hold module changed from call origination to call setup.

*Nov 20 01:32:20:497 2005 RouterA SS/7/VOICE:
  CH_EVT: [0x00030003] CH Service Interface received message [CMC_CONNECT] from SPL [SPL_DISCRIM_SIP].
*Nov 20 01:32:20:498 2005 Router A SS/7/VOICE:
  CH_EVT: [0x00030003] Received message [CMC_CONNECT] in [S_SCH_WC] state.
The CONNECT message was processed in the call setup state.

*Nov 20 01:32:20:498 2005 RouterA SS/7/VOICE:
  CH_INFO: [0x00030003] To enter MCH service.
// The new call was set up, and multiple parties were held.

*Nov 20 01:32:20:498 2005 RouterA SS/7/VOICE:
  CH_INFO: [0x00030000] CH service is finished at cause [E_CH_OK].
// The call hold service ended successfully.

The output in the following messages was created when the calling party (Telephone A) resumes the call with Telephone B by pressing hookflash a second time.

*Nov 20 01:15:05:754 2005 RouterA SS/7/VOICE:
  CH_EVT: [0x000010000] CH Service Interface received message [CMC_INFORMATION] from SPL [SPL_DISCRIM_LGS].
*Nov 20 01:15:05:754 2005 RouterA SS/7/VOICE:
  CH_EVT: [0x000010000] Receive CMC_INFORMATION message from SA module.
// A CMC_INFORMATION (HOOKFLASH) message was received.
*Nov 20 01:15:05:754 2005 RouterA SS/7/VOICE:
  CH_TMR: [0x00010000] Timer deleted, Timer ID: TIMER_SS_CH_WAIT_DIALNUM.
  // The timer waiting for dialing the first digit was deleted.

*Nov 20 01:15:05:754 2005 RouterA SS/7/VOICE:
  CH_FSM: [0x00010000] CH state move to [S_CH_O] from [S_CH_NC].
  // The state of the call hold module changed from digit collection to onhold.

*Nov 20 01:15:05:755 2005 RouterA SS/7/VOICE:
  CH_EVT: [0x00010000] Received message [CMC_INFORMATION] in [S_CH_O] state.
  // The CMC_INFORMATION (HOOKFLASH) message was processed in the onhold state.

*Nov 20 01:15:05:755 2005 RouterA SS/7/VOICE:
  CH_TMR: [0x00010000] Timer created, Timer ID: TIMER_SS_CH_WAIT_SRVCUHACK, Timer length: 20000.
  // A 20000-ms timer waiting for a SERVICE(CUH)_ACK message was created.

*Nov 20 01:15:05:755 2005 RouterA SS/7/VOICE:
  CH_FSM: [0x00010000] CH state move to [S_CUH_PRE] from [S_CH_O].
  // The state of the call hold module changed from onhold to resumption preparation.

*Nov 20 01:15:05:761 2005 RouterA SS/7/VOICE:
  CH_EVT: [0x00010001] CH Service Interface received message [CMC_SERVICE_ACK] from SPL [SPL_DISCRIM_SIP].
  // A CMC_SERVICE_ACK message was received.

*Nov 20 01:15:05:761 2005 RouterA SS/7/VOICE:
  CH_EVT: [0x00010001] Received message [CMC_SERVICE_ACK] in [S_CUH_PRE] state.
  // The CMC_SERVICE_ACK message was processed in the resumption preparation state.

*Nov 20 01:15:05:959 2005 RouterA SS/7/VOICE:
  CH_TMR: [0x00010001] Timer deleted, Timer ID: TIMER_SS_CH_WAIT_SRVCUHACK.
  // The timer waiting for a SERVICE(CUH)_ACK message was deleted.

*Nov 20 01:15:06:60 2005 RouterA SS/7/VOICE:
  CH_INFO: [0x00010000] CH service is finished at cause [E_CH_OK].
  // The call was resumed successfully, and the call hold service ended.

# Enable all types of debugging for call hold on Router B. The output similar to the following example was created when the following conditions exist:
- Call hold is enabled for both the calling party attached to Router A and called party attached to Router B.
- A call is established from Telephone A to Telephone B. The calling party (Telephone A) places the call on hold by pressing hookflash and then resumes the call by pressing hookflash again.

<RouterB> debugging voice ss ch all
SS CH all debugging switches are on

*May 18 10:42:03:469 2007 RouterB SS/7/VOICE:*
   CH_EVT: [0x00060000] CH Service Interface received message [CMC_SERVICE] from SPL [SPL_DISCRIM_SIP].
   // A call hold request message was received from the SIP module.

*May 18 10:42:03:469 2007 RouterB SS/7/VOICE:*
   CH_INFO: [0x00060000] Start a new CH service successfully.
   // The call hold service was successfully started.

*May 18 10:42:03:469 2007 RouterB SS/7/VOICE:*
   CH_EVT: [0x00060000] Received message [CMC_SERVICE] in state [S_CH_IDLE].
   // The call hold request message was processed in the idle state.

*May 18 10:42:03:469 2007 RouterB SS/7/VOICE:*
   CH_INFO: [0x00060000] Successfully hold on local in [S_CH_IDLE] state.
   // The call was successfully held at the local end.

*May 18 10:42:03:469 2007 RouterB SS/7/VOICE:*
   CH_TMR: [0x00060000] Timer created, Timer ID: TIMER_SS_CH_WAIT_MAXHOLDED, Timer length: 86400000.
   // A 86400000-ms timer was started for the held party to wait.

*May 18 10:42:03:469 2007 RouterB SS/7/VOICE:*
   CH_FSM: [0x00060000] CH state move to [S_CH_R] from [S_CH_IDLE].
   // The state of the call hold module changed from idle to held.

*May 18 10:42:03:469 2007 RouterB SS/7/VOICE:*
   CH_INFO: [0x00060000] Successfully send service acknowledge in [S_CH_R] state.
   // A call hold acknowledgment was sent.

*May 18 10:42:03:469 2007 RouterB SS/7/VOICE:*
   CH_EVT: [0x00060000] CH Service Interface received message [CMC_SERVICE] from SPL [SPL_DISCRIM_SIP].
   // A call resume request message was received from the SIP module.

*May 18 10:42:03:469 2007 RouterB SS/7/VOICE:*
   CH_EVT: [0x00060000] Received message [CMC_SERVICE] in [S_CH_R] state.
   // The call resume request was processed in the held state.

*May 18 10:42:03:469 2007 RouterB SS/7/VOICE:*
   CH_INFO: [0x00060000] Successfully unhold on local in [S_CH_R] state.
   // The locally held call was successfully resumed.
debugging voice ss conf

Use `debugging voice ss conf` to enable three-party conference debugging.

Use `undo debugging voice ss conf` to disable three-party conference debugging.

**Syntax**

```
debugging voice ss conf { all | error | event | fsm | info | timer }
undo debugging voice ss conf { all | error | event | fsm | info | timer }
```

**Default**

Three-party conference debugging is disabled.

**Views**

User view

**Default command level**

2: System level

**Parameters**

- `all`: Specifies all types of debugging for three-party conference.
- `error`: Specifies error debugging.
- `event`: Specifies event debugging.
- `fsm`: Specifies finite state machine debugging.
- `info`: Specifies information debugging.
- `timer`: Specifies timer debugging.

**Usage guidelines**

Table 122 describes output fields and messages for the `debugging voice ss conf fsm` command.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conference state move from %s to [xxx].</td>
<td>Conference state changed from xxx state to xxx state.</td>
</tr>
</tbody>
</table>

**Examples**

# Enable all types of debugging for three-party conferencing on Router B. Output similar to the following example is generated when the user at Telephone B presses "3" to enter a three-party conference under these conditions:
- Telephone A, Telephone B, and Telephone C are attached to Router A, Router B, and Router C, respectively.
- Telephone A, Telephone B, and Telephone C are enabled with call hold, call hold and three-party conferencing, and call hold, respectively.
- After Telephone A establishes a call with Telephone B, the user at Telephone B presses hookflash to place the call on hold and then calls Telephone C. Then, Telephone C goes off-hook.

```
<RouterB> debugging voice ss conf all
*Jan 10 10:55:56:817 2008 ipp44-UP SS/7/VOICE:
  CONF_INFO: [0x0000000c] Conference receive CMC_INFORMATION message.
  // The third-party conference module received a CMC_INFORMATION message.

*Jan 10 10:55:56:817 2008 ipp44-UP SS/7/VOICE:
  CONF_INFO: [0x0000000c] Conference CCB is initialized successfully.
  // The third-party conference module established and initialized three-party conference after receiving the DTMF 3 message.

*Jan 10 10:55:56:818 2008 ipp44-UP SS/7/VOICE:
  CONF_EVT: [0x0000000c] Receive CMC_INFORMATION message at S_CONF_IDLE state.
  // The third-party conference module received the CMC_INFORMATION message in the S_CONF_IDLE state.

*Jan 10 10:55:56:818 2008 ipp44-UP SS/7/VOICE:
  CONF_INFO: [0x0000000c] Succeed in getting conference parameter.
  // The third-party conference module got the conference parameter successfully.

*Jan 10 10:55:56:818 2008 ipp44-UP SS/7/VOICE:
  CONF_TMR: [0x0000000c] Timer created, Timer ID: TIMER_SS_CONF_WAIT_DRVACK, Timer length: 500.

*Jan 10 10:55:56:818 2008 ipp44-UP SS/7/VOICE:
  CONF_FSM: [0x0000000c] Conference state move from S_CONF_IDLE to S_CONF_WDRV.

*Jan 10 10:55:56:851 2008 ipp44-UP SS/7/VOICE:
  CONF_INFO: [0x0000000c] Conference receive CMC_VIM message.

*Jan 10 10:55:56:852 2008 ipp44-UP SS/7/VOICE:
  CONF_EVT: [0x0000000c] Receive CMC_VIM message at S_CONF_WDRV state.

*Jan 10 10:55:56:952 2008 ipp44-UP SS/7/VOICE:
  CONF_INFO: [0x0000000c] Succeed in initializing conference DSP.
  // The third-party conference module received the CMC_VIM message. It is a successful response, which indicates that conference DSP was initialized successfully.

*Jan 10 10:55:56:952 2008 ipp44-UP SS/7/VOICE:
  CONF_INFO: [0x0000000c] Succeed in initializing conference information.
```

241
CONF_TMR: [0x00000001] Timer created, Timer ID: TIMER_SS_CONF_WAIT_CUHACK, Timer length: 20000.

*Jan 10 10:55:57:254 2008 ipp44-UP SS/7/VOICE:
CONF_FSM: [0x0000000c] Conference state move from S_CONF_WDRV to S_CONF_WCUH.

*Jan 10 10:55:57:355 2008 ipp44-UP SS/7/VOICE:
CONF_INFO: [0x00000001] Conference receive CMC_SERVICE_ACK message.

*Jan 10 10:55:57:506 2008 ipp44-UP SS/7/VOICE:
CONF_EVT: [0x00000001] Receive CMC_SERVICE_ACK message at S_CONF_WCUH state.

*Jan 10 10:55:57:657 2008 ipp44-UP SS/7/VOICE:
CONF_INFO: [0x00000001] Conference is established successfully.

// The third-party conference module received the CMC_SERVICE_ACK message. It is a successful response, which indicates that the conference was established successfully.

*Jan 10 10:55:57:758 2008 ipp44-UP SS/7/VOICE:
CONF_FSM: [0x0000000c] Conference state move from S_CONF_WCUH to S_CONF_ACTIVE

// The conference changed from the S_CONF_WCUH state to the S_CONF_ACTIVE state.

*Jan 10 10:55:57:919 2008 ipp44-UP SS/7/VOICE:
CONF_INFO: [0x00000001] Useless channel ready message.

// The third-party conference module received the channel-ready message from the hold-on party. This message does not need to be processed.

*Jan 10 10:55:58:20 2008 ipp44-UP SS/7/VOICE:
CONF_INFO: [0x00000001] Current state is conference.

// The current state is conference.

*Jan 10 10:55:58:120 2008 ipp44-UP SS/7/VOICE:
CONF_INFO: [0x00000001] Conference receive CMC_CODECSWITCH message.

// The third-party conference module received the CMC_CODECSWITCH message during conference.

*Jan 10 10:55:58:272 2008 ipp44-UP SS/7/VOICE:
CONF_EVT: [0x00000001] Receive CMC_CODECSWITCH message at S_CONF_ACTIVE state.

// The third-party conference module received the CMC_CODECSWITCH message in the conference sub-state S_CONF_ACTIVE.

debugging voice ss cr
Use debugging voice ss cr to enable call barring debugging.
Use undo debugging voice ss cr to disable call barring debugging.

Syntax
debugging voice ss cr { all | error | info }
undo debugging voice ss cr { all | error | info }

Default
Call barring debugging is disabled.

Views
User view

Default command level
2: System level

Parameters

all: Specifies all types of debugging for call barring.
error: Specifies error debugging.
info: Specifies information debugging.

Usage guidelines

Table 123 describes output fields and messages for the debugging voice ss cr error command.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>The leg is null in CR analyse process.</td>
<td>The leg was null in call barring analysis.</td>
</tr>
</tbody>
</table>

Table 124 describes output fields and messages for the debugging voice ss cr info command.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PS type leg, Dial-out restriction service has been enabled in IP caller.</td>
<td>Outgoing call barring was enabled for the IP caller.</td>
</tr>
<tr>
<td>PS type leg, Dial-out restriction service has been disabled in IP caller.</td>
<td>Outgoing call barring was disabled for the IP caller.</td>
</tr>
<tr>
<td>PS type leg, No CR config in CR analyse process in IP caller.</td>
<td>Call barring configuration of the IP caller was unavailable.</td>
</tr>
<tr>
<td>CS type leg, Dial-in restriction service has been enabled in IP callee.</td>
<td>Incoming call barring was enabled for the IP callee.</td>
</tr>
<tr>
<td>CS type leg, Dial-in restriction service has been disabled in IP callee.</td>
<td>Incoming call barring was disabled for the IP callee.</td>
</tr>
<tr>
<td>CS type leg, No CR config in CR analyse process in IP callee.</td>
<td>Call barring configuration of the IP callee was unavailable.</td>
</tr>
<tr>
<td>Dial-out restriction service has been enabled in local caller.</td>
<td>Outgoing call barring was enabled for the local caller.</td>
</tr>
<tr>
<td>Dial-out restriction service has been disabled in local caller.</td>
<td>Outgoing call barring was disabled for the local caller.</td>
</tr>
<tr>
<td>No CR config in CR analyse process in local callee.</td>
<td>Call barring configuration of the local callee was unavailable.</td>
</tr>
<tr>
<td>Dial-in restriction service has been enabled in local callee.</td>
<td>Incoming call barring was enabled for the local callee.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Dial-in restriction service has been disabled in local callee.</td>
<td>Incoming call barring was disabled for the local callee.</td>
</tr>
<tr>
<td>No CR config in CR analyse process in local caller.</td>
<td>Call barring configuration of the local caller was unavailable.</td>
</tr>
</tbody>
</table>

**Examples**

```
# Enable all types of debugging for call barring on Router A. Call barring is enabled for Telephone A. Output similar to the following example is generated when Telephone B attached to Router B calls Telephone A attached to Router A:
<RouterA> debugging voice ss cr all
SS CR all debugging switches are on
<RouterA>
*Nov 20 02:07:05:59 2005 RouterA SS/7/VOICE:
  CR_INFO: [0x00050001] CS type leg, Dial-in restriction service has been enabled in IP callee.
// Incoming call barring was enabled, and call termination failed.
```

```
# Enable all types of debugging for call barring on Router A. Call barring is enabled for Telephone B. Output similar to the following example is generated when Telephone A attached to Router B calls Telephone B attached to Router A:
<RouterA> debugging voice ss cr all
SS CR all debugging switches are on
<RouterA>
*Nov 20 02:20:36:715 2005 RouterA SS/7/VOICE:
  CR_INFO: [0x00060001] PS type leg, Dial-out restriction service has been enabled in IP caller.
// Outgoing call barring was enabled, and call origination failed.
```

**debugging voice ss ct**

Use **debugging voice ss ct** to enable call transfer debugging.

Use **undo debugging voice ss ct** to disable call transfer debugging.

**Syntax**

```
default debugging voice ss ct { all | error | event | fsm | info | timer }
undo debugging voice ss ct { all | error | event | fsm | info | timer }
```

**Default**

Call transfer debugging is disabled.

**Views**

User view

**Default command level**

2: System level

**Parameters**

- **all**: Specifies all types of debugging for call transfer.
error: Specifies error debugging.

event: Specifies event debugging.

fsm: Specifies finite state machine debugging.

info: Specifies information debugging.

timer: Specifies timer debugging.

Usage guidelines

Table 125 describes output fields and messages for the debugging voice ss ct error command.

Table 125 Output from the debugging voice ss ct error command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Failed to find CMC ccb in CTO interface | Call transfer failed to find a CMC CCB in the call transfer originator (CTO) interface. Similar debugging information includes:  
  • Failed to find XXX in SSS.  
  • Failed to get XXX in SSS.  
  • Failed to send XXX in SSS.  
  • Failed to create XXX in SSS.  
  • Failed to restore XXX in SSS.  
  • Failed to play XXX tone in SSS.  
  • Failed to stop XXX tone in SSS.  
  • Failed to recover XXX in SSS.  
  All this information indicates that an operation (represented by xxx) failed. SSS following the preposition “in” (or sometimes the conjunction “when”) indicates the place or time that the operation failed. |
| Getting first voice entity when find entity in CT | Call transfer failed to obtain the first voice entity in the CT service. Similar debugging information is in the following format:  
  The FEATURE service failed to obtain xxx failed (when TTT) in SSS. This indicates that an operation (represented by xxx) failed. TTT indicates the time that the operation failed. SSS indicates the place where the operation failed. |
<p>| The source of the message is invalid in CTO interface | The received RELEASE message was not a local message, and the call transfer service could not be triggered. |
| Received invalid release message in CTO interface. Release cause is 3 | The received RELEASE message was not an onhook message, and the call transfer service could not be triggered. |
| The state of active leg is invalid in CTO interface | The state of the active leg is incorrect, and the call transfer could not be triggered. |
| The number 3 of talk path is invalid in CTO interface | The number of the current talk paths was incorrect, and the call transfer could not be triggered. |
| Received invalid cmc message in CTO interface | The received message was not a RELEASE message, and the call transfer service could not be triggered. |
| The state of CTO is invalid in CTO interface. State = 3 | The state of the CTO was incorrect. |</p>
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Received invalid service message when CTR ccb is null in CTR interface</td>
<td>The received SERVICE message was not a CT message, and the call service could not be triggered.</td>
</tr>
<tr>
<td>Received invalid message when CTR ccb is null in CTR interface</td>
<td>The received message was not a SERVICE message, and the call transfer recipient (CTR) could not process it.</td>
</tr>
<tr>
<td>The state of CTR is invalid in CTR interface. State = 4</td>
<td>The state of the CTR was invalid.</td>
</tr>
<tr>
<td>The role of the CT CCB is invalid when deleting ccb</td>
<td>The service state of the CT CCB to be deleted was invalid.</td>
</tr>
<tr>
<td>Malloc memory for CT CCB when creation</td>
<td>Call transfer failed to allocate memory to the CT CCB during CT CCB creation.</td>
</tr>
<tr>
<td>Can’t get matched entity when find entity in CT</td>
<td>No matching voice entity existed.</td>
</tr>
<tr>
<td>The dial peer type is invalid when fill information for new call in CT</td>
<td>The found voice entity type was invalid.</td>
</tr>
<tr>
<td>Received invalid CMC message in CTO idle status process</td>
<td>An invalid CMC message was received during the processing of the CTR’s idle state.</td>
</tr>
<tr>
<td></td>
<td>Similar debugging information is in the following format:</td>
</tr>
<tr>
<td></td>
<td>Received invalid XXX message (at TTT) in SSS. This information indicates that an invalid message was received. XXX represents the type of the message. SSS indicates the position where the invalid message is received. TTT indicates the call transfer type (bct, eact, or act).</td>
</tr>
<tr>
<td>The message set is invalid in CTO idle status process</td>
<td>An invalid type of message was received during the processing of the CTR’s idle state.</td>
</tr>
<tr>
<td></td>
<td>Similar debugging information is in the following format:</td>
</tr>
<tr>
<td></td>
<td>MMM was invalid (when/at TTT) in SSS. This information indicates that an event was invalid. MMM indicates an event, SSS indicates the position where an error occurs. TTT indicates a more specific position, state, or point of time.</td>
</tr>
<tr>
<td>Received release message from invalid source at ACT in CTO’s WACK status process</td>
<td>A RELEASE message was received from an invalid source during the processing of the CTO’s WACK state.</td>
</tr>
<tr>
<td></td>
<td>Similar debugging information is in the following format:</td>
</tr>
<tr>
<td></td>
<td>Received MMM from invalid source (at TTT) in SSS. This information indicates that a message was received from an invalid source. MMM indicates the type of the received message. SSS indicates the position where an error occurs. TTT indicates a more specific position or point of time.</td>
</tr>
<tr>
<td>The service type of service message is invalid in CTO pending status process</td>
<td>The service type of the received SERVICE message was invalid during the processing of the CTO’s pending state.</td>
</tr>
<tr>
<td>The final-recipient’s number is null in building new setup message to spl</td>
<td>The final recipient found that the forwarded-to number was null after receiving a transfer request to originate a new call.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| Can not find the replaced leg in CTT interface | The final recipient cannot find the leg to be replaced.  
Cannot find XXX in SSS.  
Similar debugging information includes:  
Cannot get XXX in SSS.  
Cannot process XXX in SSS.  
This information indicates that something cannot be done. XXX indicates the task that cannot be done. SSS indicates the position where an error occurs. |
| Invalid connect state in CTT process for CH | The connection state of the held call to be replaced was invalid. |
| The leg is already invalid after send alerting in CTT process for idle | The leg might be exceptionally released after an ALERTING message was sent.  
Similar debugging information is in the following format:  
The leg is already invalid after send MMM in SSS.  
This information indicates that the leg was exceptionally released after a message was sent. MMM indicates the type of the sent message. SSS indicates the position where or the point of time when an error occurs. |
| There is no MCH ccb in CTT process for MCH | No MCH CCB was created when a held call was replaced in MCH. |

Table 126 describes output fields and messages for the `debugging voice ss ct event` command.

**Table 126 Output from the debugging voice ss ct event command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Receive CMCRELEASE message from SA module in CTO interface</td>
<td>The CTO interface received a CMCRELEASE message from the SA module.</td>
</tr>
<tr>
<td>Receive CMC_SERVICE message from SA module in CTR interface</td>
<td>The CTR interface received a CMC_SERVICE message from the SA module.</td>
</tr>
<tr>
<td>Receive CMC_SETUP message from SA module in CTT interface</td>
<td>The CTT interface received a CMC_SETUP message from the SA module.</td>
</tr>
</tbody>
</table>
| Receive CMC_RELEASE message from SA module in originator’s IDLE state | The CTO received a CMC_RELEASE message in the idle state.  
Similar debugging information is in the following format:  
Receive MMM message from FFF module in SSS state.  
This information indicates that the xxx module received an xxx message in xxx state. MMM indicates the received message.  
FFF indicates the source module that sent the message to the ST module. SSS indicates the state of the CT module when it received the message. |
The CTO in the idle state sent a SERVICE(CT) message to the CMC module.

Similar debugging information is in the following format:

Send MMM message to FFF module (in SSS state)
This information indicates that the CTO in xxx state sent an xxx message to the xxx module. MMM indicates the message that was sent. FFF indicates the module to which the message was sent. SSS (optional) indicates the state of the CTO when it sent the message.

Table 127 describes output fields and messages for the debugging voice ss ct fsm command.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Send SERVICE(CT) message to CMC module in S_CTO_IDLE state</td>
<td>The CTO in the idle state sent a SERVICE(CT) message to the CMC module. Similar debugging information is in the following format: Send MMM message to FFF module (in SSS state) This information indicates that the CTO in xxx state sent an xxx message to the xxx module. MMM indicates the message that was sent. FFF indicates the module to which the message was sent. SSS (optional) indicates the state of the CTO when it sent the message.</td>
</tr>
</tbody>
</table>

Table 127 Output from the debugging voice ss ct fsm command

Table 128 describes output fields and messages for the debugging voice ss ct info command.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>State changed from S_CTR_IDLE to S_CTR_SCWA</td>
<td>The service state of the CTR changed from idle to SCWA. Similar debugging information is in the following format: State changed from SSS to DSS. This information indicates that the service state changed from SSS to DSS. SSS indicates the initial service state, and DSS indicates the target service state.</td>
</tr>
</tbody>
</table>

Table 128 Output from the debugging voice ss ct info command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ends service of CTO for invalid state in CTO interface</td>
<td>The CT service ended because the service state of the current CTO was invalid.</td>
</tr>
<tr>
<td>Ends service of CTR for invalid state in CTR interface</td>
<td>The CT service ended because the service state of the current CTR was invalid.</td>
</tr>
<tr>
<td>Release all calls and CT ends because of general error in CTO interface</td>
<td>All calls were released, and the CT service ended because an uncontrollable error occurred.</td>
</tr>
<tr>
<td>Release call with final-recipient because of failure of timer creation</td>
<td>The call with the final recipient was released because the timer waiting for the final recipient's release failed to be created.</td>
</tr>
<tr>
<td>CT ends because of no CMC ccb in CTO idle status process</td>
<td>The CT service ended because there was no corresponding CMC CCB in the CTO's idle state. Similar debugging information is in the following format: CT ends because of RRR in SSS. This information indicates that the CT service ended for some reason. RRR indicates the reason and SSS indicates the position where an event happened.</td>
</tr>
<tr>
<td>Release all calls and CT ends</td>
<td>The CT service ended, and all calls related to the CMC CCB were released.</td>
</tr>
<tr>
<td>The service_ack for CTO is fail in CTO's WACK status process</td>
<td>The CTR was unable to process the SERVICE_CT message and returned a failure acknowledgment.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| Receive the recipient’s release message at BCT in CTO’s WACK status process. Service process ends | A RELEASE message was received from the CTR when the CRO’s WACK state was not notified of, and the CT service ended.  
Similar debugging information is in the following format:  
Receive RRR’s MMM message at TTT in SSS. (Service process ends)  
This information indicates that an event occurred after a message was received from a role in the CT service. RRR indicates a role in the CT service. MMM indicates the message. TTT indicates a specific position, state, or point of time. SSS indicates the position where an event happened. The result can be attached at the end of the debugging information. |
| There is only one talk at ACT in CTO’s WACK status process. Service process ends | There was only one talk in the CTO’s WACK state, and the CT service ended. Similar debugging information might also appear in other states, except that the position where an event happens is different. |
| There is not only one talk path in CTR’s WREL status process         | There was more than one talk when a resume request was received from the CTO, and the call could not be resumed.                              |
| Release call with final-recipient at ACT in CTO’s WACK status process | The call with the final recipient was released when the CTO’s WACK state was notified.  
Similar debugging information is in the following format:  
Release call with RRR at TTT in SSS. (Service ends)  
This information indicates that the call with a role in the CT service was released. RRR indicates a role in the CT service. TTT indicates a specific position, state, or point of time. SSS indicates the position where an event happened. Supplementary description can be attached at the end of the debugging information. |
| Release all calls in CTO’s WACK status process                       | All related calls were released in the CTO’s WACK state.  
Similar debugging information might also appear in other states, except that the position where an event happened is different. |
| Release all calls and service process ends because of waiting service ACK message timeout | Because of SERVICE_ACK timeout, the CT service ended, and all related calls were released.  
Similar debugging information might also appear for other reasons, except that the reason for an event is different. |
| Delete the new call leg in building new setup message to spl          | The new call by the CTR failed, and the new call leg was deleted.  
| The originator has released. Will not send notify(alerting) in CTR’s SCWA status process | Because the CTO released the call, no NOTIFY (ALERTING) message was sent to it. |
| The originator has already released. Service ends and return to normal call | Because the CTO released the call, no NOTIFY message was sent. The CT service ended directly and returned to the normal call state. |
| Service ends when the only one talk path been released                | Because the only talk path was released, the CT service ended. Sometimes, the position where an event happened might be attached at the end of “in SSS” |
The original active leg is null. Service ends
The original active leg could not be found, and the CT service ended.

Release CTR ccb and CT ends in CTR’s WREL status process
The service resources of the CTR were released.

Service ends after waiting originator’s release message timeout
Because a RELEASE message was not received from the CTO within the specified time, the CT service ended and returned to the normal call state.

Release the original call after success to replace in CTT process for idle
The original call was released after replacement succeeded. Similar debugging information is in the following format: Release CCC (after TTT) in SSS.

This information indicates that a call was released. CCC indicates a talk path in the CT service. TTT indicates a specific position, state, or point of time. and SSS indicates the position where an event happened.

Delete the new leg in CTT process for CW
The new call leg was released because replacement failed. Similar debugging information might also appear in other states, except that the position where an event happens is different.

Table 129 describes output fields and messages for the `debugging voice ss ct timer` command.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Timer created, TimerId: TIMER_SS_CTR_WAIT_CTO_REL, Timer length : 100000</td>
<td>The SS_CTR_WAIT_CTO_REL timer was successfully created, with a length of 100000 milliseconds. Similar debugging information is in the following format: Timer created, TimerId : DDD, Timer length : LLL. This information indicates that the xxx timer with a length of xxx milliseconds was successfully created. DDD indicates the timer ID, and LLL indicates the timer length.</td>
</tr>
<tr>
<td>Timer deleted, TimerId : TIMER_SS_CTR_WAIT_CTO_REL</td>
<td>The SS_CTR_WAIT_CTO_REL timer was successfully deleted. Similar debugging information is in the following format: Timer deleted, TimerId: DDD. This information indicates that the xxx timer was successfully deleted. DDD indicates the timer ID.</td>
</tr>
<tr>
<td>Timer timeout, TimerId : TIMER_SS_CTR_WAIT_CTO_REL</td>
<td>The SS_CTR_WAIT_CTO_REL timer was successfully deleted. Similar debugging information is in the following format: Timer timeout, TimerId : DDD. This information indicates that the xxx timer expired. DDD indicates the timer ID.</td>
</tr>
</tbody>
</table>

Examples

Output similar to the following example is generated when the following conditions exist:
• Telephone A, Telephone B, and Telephone C are attached to Router A, Router B, and Router C, respectively.
• Telephone B and Telephone C are both enabled with call hold. Telephone A is enabled with call transfer.
• After Telephone A establishes a call with Telephone B, the user at Telephone A presses hookflash to place the call on hold and then calls Telephone C. Then, the user at Telephone A hangs up after hearing ringback tones from Telephone C.

# Enable all types of debugging for call transfer on Router A.
<RouterA> debugging voice ss ct all
SS CT all debugging switches are on
<RouterA>
*Nov 20 02:33:39:132 2005 RouterA SS/7/VOICE:
  CT_EVT: [0x00070000] Receive CMC_RELEASE message from SA module in CTO interface.
*Nov 20 02:33:39:132 2005 RouterA SS/7/VOICE:
  CT_EVT: [0x00070000] Receive CMC_RELEASE message from SA module in originator's IDLE state.
// A CMC_RELEASE message was received.

*Nov 20 02:33:39:134 2005 RouterA SS/7/VOICE:
  CT_EVT: [0x00070001] Send SERVICE(CT) message to CMC module in S_CTO_IDLE state.
// A SERVICE(CT) message was received.

*Nov 20 02:33:39:134 2005 RouterA SS/7/VOICE:
  CT_TMR: [0x00070001] Timer created, Timer ID: TIMER_SS_CTO_WAIT_ACK, Timer length: 20000.
// A 20000-ms timer waiting for a CT acknowledgment was created.

*Nov 20 02:33:39:134 2005 RouterA SS/7/VOICE:
  CT_FSM: [0x00070000] State changed from S_CTO_IDLE to S_CTO_WACK.
// The state of the CT module changed from idle to WACK.

*Nov 20 02:33:39:141 2005 RouterA SS/7/VOICE:
  CT_EVT: [0x00070001] Receive CMC_SERVICE_ACK message from SA module in CTO interface.
*Nov 20 02:33:39:141 2005 RouterA SS/7/VOICE:
  CT_EVT: [0x00070001] Receive CMC_SERVICE_ACK message from SA module in originator's WACK state.
// A CMC_SERVICE_ACK message was received.

*Nov 20 02:33:39:141 2005 RouterA SS/7/VOICE:
  CT_TMR: [0x00070001] Timer deleted, Timer ID: TIMER_SS_CTO_WAIT_ACK.
// The timer waiting for a CMC_SERVICE_ACK message was deleted.

*Nov 20 02:33:39:142 2005 RouterA SS/7/VOICE:
  CT_TMR: [0x00070001] Timer created, Timer ID: TIMER_SS_CTO_WAIT_NOTIFY, Timer length: 90000.
// A 90000-ms timer waiting for a NOTIFY message was created.
*Nov 20 02:33:39:286 2005 RouterA SS/7/VOICE:
  CT_FSM: [0x00070001] State changed from S_CTO_WACK to S_CTO_PENDING.
  // The state of the CT module changed from WACK to pending.

*Nov 20 02:33:39:438 2005 RouterA SS/7/VOICE:
  CT_EVT: [0x00070001] Receive CMC_SERVICE message from SA module in CTO interface.

*Nov 20 02:33:39:589 2005 RouterA SS/7/VOICE:
  CT_EVT: [0x00070001] Receive CMC_SERVICE message from SA module in originator's PENDING state.
  // A NOTIFY message was received.

*Nov 20 02:33:39:740 2005 RouterA SS/7/VOICE:
  CT_EVT: [0x00070001] Send CMC_SERVICE_ACK(NOTIFY_OK) message to CMC module in S_CTO_PENDING state.
  // A NOTIFY_ACK message was sent.

*Nov 20 02:33:39:840 2005 RouterA SS/7/VOICE:
  CT_TMR: [0x00070001] Timer deleted, Timer ID: TIMER_SS_CTO_WAIT_NOTIFY.
  // The timer waiting for a NOTIFY message was restarted.

*Nov 20 02:33:39:847 2005 RouterA SS/7/VOICE:
  CT_EVT: [0x00070001] Receive CMC_SERVICE message from SA module in CTO interface.

*Nov 20 02:33:39:847 2005 RouterA SS/7/VOICE:
  CT_EVT: [0x00070001] Receive CMC_SERVICE message from SA module in originator's PENDING state.
  // A NOTIFY message was received.

*Nov 20 02:33:39:847 2005 RouterA SS/7/VOICE:
  CT_EVT: [0x00070001] Send CMC_SERVICE_ACK(NOTIFY_OK) message to CMC module in S_CTO_PENDING state.
  // A NOTIFY_ACK message was sent.

*Nov 20 02:33:39:848 2005 RouterA SS/7/VOICE:
  CT_TMR: [0x00070001] Timer deleted, Timer ID: TIMER_SS_CTO_WAIT_NOTIFY.
  // The timer waiting for a NOTIFY message was deleted.

*Nov 20 02:33:39:848 2005 RouterA SS/7/VOICE:
  CT_INFO: [0x00070001] Release all calls in CTO pending status process.
  // The CTO released all related local calls, and the CT service ended.

# Enable all types of debugging for call transfer on Router B.
<RouterB> debugging voice ss ct all
SS CT all debugging switches are on

*May 18 11:40:49:15 2007 RouterB SS/7/VOICE:
  CT_EVT: [0x000b0000] Receive CMC_SERVICE message from SA module in CTR interface.
*May 18 11:40:49:15 2007 RouterB SS/7/VOICE:
  CT_EVT: [0x000b0000] Receive CMC_SERVICE message from SA module in recipient's IDLE state.

  // A CMC_SERVICE message was received.

*May 18 11:40:49:15 2007 RouterB SS/7/VOICE:
  CT_EVT: [0x000b0000] Send SERVICE_ACK(CT_OK) message to CMC module in S_CTR_IDLE state.

  // A SERVICE_ACK (CT_OK) message was sent to the CMC module in the CTR's idle state.

*May 18 11:40:49:15 2007 RouterB SS/7/VOICE:
  CT_EVT: [0x000b0002] Send SETUP message to CMC module in S_CTR_IDLE state.

  // A new call was originated according to the parameters in the CMC_SERVICE message.

*May 18 11:40:49:15 2007 RouterB SS/7/VOICE:
  CT_FSM: [0x000b0000] State changed from S_CTR_IDLE to S_CTR_SCWA.

  // The state of the CTR changed from idle to waiting-for-alerting.

*May 18 11:40:49:61 2007 RouterB SS/7/VOICE:
  CT_EVT: [0x000b0003] Receive CMC_ALERTING message from SA module in CTR interface.
*May 18 11:40:49:61 2007 RouterB SS/7/VOICE:
  CT_EVT: [0x000b0003] Receive CMC_ALERTING message from SA module in recipient's SCWA state.

  // A CMC_ALERTING message was received from the final recipient.

*May 18 11:40:49:61 2007 RouterB SS/7/VOICE:
  CT_FSM: [0x000b0003] State changed from S_CTR_SCWA to S_CTR_SCWC.

  // The state of the CTR changed from waiting-for-alerting to waiting-for-connect.

*May 18 11:40:49:61 2007 RouterB SS/7/VOICE:
  CT_EVT: [0x000b0003] Send SERVICE(NOTIFY ALERTING) message to CMC module in recipient's SCWC state.

  // A SERVICE (NOTIFY ALERTING) message was sent to the CTO, acknowledging the ALERTING message.

*May 18 11:40:49:155 2007 RouterB SS/7/VOICE:
  CT_EVT: [0x000b0000] Receive CMC_SERVICE_ACK message from SA module in CTR interface.
*May 18 11:40:49:306 2007 RouterB SS/7/VOICE:
  CT_EVT: [0x000b0000] Receive CMC_SERVICE_ACK message from SA module in recipient's SCWC state.

  // A CMC_SERVICE_ACK message was received from the CTO.
A CMC_CONNECT message was received from the final recipient.

The state of the CTR changed from waiting-for-connect to wait-for-release.

A SERVICE (NOTIFY CONNECT) message was sent to the CTO, acknowledging the CONNECT message.

A 100000-ms timer waiting for a release message from the CTO was created.

A CMC_SERVICE_ACK message was received from the CTO.

A RELEASE message was received from the CTO.

The timer waiting for a RELEASE message was deleted.

The call was transferred successfully. The CTR released the call with the CTO. The call transfer service ended.
debugging voice ss cw

Use **debugging voice ss cw** to enable call waiting debugging.

Use **undo debugging voice ss cw** to disable call waiting debugging.

**Syntax**

```
debugging voice ss cw { all | error | event | fsm | info | timer }
undo debugging voice ss cw { all | error | event | fsm | info | timer }
```

**Default**

The FEATURE service debugging is disabled.

**Views**

User view

**Default command level**

2: System level

**Parameters**

- **all**: Specifies all types of debugging for call waiting.
- **error**: Specifies error debugging.
- **event**: Specifies event debugging.
- **fsm**: Specifies finite state machine debugging.
- **info**: Specifies information debugging.
- **timer**: Specifies timer debugging.

**Usage guidelines**

Table 130 describes output fields and messages for the **debugging voice ss cw error** command.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Failed to get brother leg in judge talking process.</td>
<td>The FEATURE service failed to obtain a brother leg in talk path judgment.</td>
</tr>
<tr>
<td>The original CMC CCB is null in judge talking process.</td>
<td>The CMC CCB of the originator was null in talk path judgment.</td>
</tr>
<tr>
<td>Allocate memory failed for create CW CCB.</td>
<td>The FEATURE service failed to allocate memory to the CW CCB during CW CCB creation.</td>
</tr>
<tr>
<td>Failed to find VIM info in create CW CCB.</td>
<td>The FEATURE service failed to find VIM information during CW CCB creation.</td>
</tr>
<tr>
<td>Failed to create CW CCB in interface process.</td>
<td>The FEATURE service failed to create the CW CCB in interface processing.</td>
</tr>
<tr>
<td>Received invalid CMC message in interface process.</td>
<td>An invalid CMC message was received in interface processing.</td>
</tr>
<tr>
<td>Wrong CW service state in interface process.</td>
<td>The CW service state is incorrect in interface processing.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Get brother leg fail on CW idle state.</td>
<td>The FEATURE service failed to obtain a brother leg in the CW_IDLE state.</td>
</tr>
<tr>
<td>Get original call leg fail on CW idle state</td>
<td>The FEATURE service failed to obtain the originator leg in the CW_IDLE state.</td>
</tr>
<tr>
<td>Failed to find new talk path in idle process.</td>
<td>The FEATURE service failed to find the new talk path in the CW_IDLE state.</td>
</tr>
<tr>
<td>Failed to create CW SRVACK timer on CW idle state.</td>
<td>The FEATURE service failed to create the CW SRVACK timer in the CW_IDLE state.</td>
</tr>
<tr>
<td>Failed to create intsend alerting timer on CW pending state.</td>
<td>The FEATURE service failed to create the ALERTING timer in the CW_IDLE state.</td>
</tr>
<tr>
<td>Received invalid timer message on CW idle state.</td>
<td>An invalid timer message was received in the CW_IDLE state.</td>
</tr>
<tr>
<td>Failed to delete CW_ACK timer in delete CCB.</td>
<td>The FEATURE service failed to delete the CW_ACK timer when deleting the CCB.</td>
</tr>
<tr>
<td>Failed to delete CH_ACK timer in delete CCB.</td>
<td>The FEATURE service failed to delete the CH_ACK timer when deleting the CCB.</td>
</tr>
<tr>
<td>Failed to delete send alerting timer in delete CCB.</td>
<td>The FEATURE service failed to delete the ALERTING timer when deleting the CCB.</td>
</tr>
<tr>
<td>Failed to delete CUH_ACK timer in delete CCB.</td>
<td>The FEATURE service failed to delete the CUH_ACK timer when deleting the CCB.</td>
</tr>
<tr>
<td>Failed to delete play CW tone timer in delete CCB.</td>
<td>The FEATURE service failed to delete the CW_TONE timer when deleting the CCB.</td>
</tr>
<tr>
<td>Failed to release talk path on CW pending state.</td>
<td>The FEATURE service failed to release the talk path in the CW pending state.</td>
</tr>
<tr>
<td>Failed to get talk path on CW pending state.</td>
<td>The FEATURE service failed to obtain a talk path in the CW pending state.</td>
</tr>
<tr>
<td>Failed to restore the CCB info on CW pending state.</td>
<td>The FEATURE service failed to update the CCB information in the pending state.</td>
</tr>
<tr>
<td>Failed to get brother leg on CW pending state before sending new setup.</td>
<td>The FEATURE service failed to obtain a brother leg failed before a new SETUP message was sent.</td>
</tr>
<tr>
<td>Failed to switch call leg on CW pending state.</td>
<td>The FEATURE service failed to switch the call leg in the pending state.</td>
</tr>
<tr>
<td>Failed to play busy tone on CW pending state.</td>
<td>The FEATURE service failed to play call waiting tones in the pending state.</td>
</tr>
<tr>
<td>Received invalid CMC_INFORMATION message on CW pending state..</td>
<td>An invalid CMC_INFORMATION message was received in the pending state.</td>
</tr>
<tr>
<td>Connect Media failed on CW pending state.</td>
<td>The FEATURE service failed to connect the media in the pending state.</td>
</tr>
<tr>
<td>The leg is already invalid after failed to connect media on CW pending state.</td>
<td>The FEATURE service failed to send a CONNECT message in the pending state.</td>
</tr>
<tr>
<td>Failed to send connect message after failed to connect media on CW pending state.</td>
<td>The FEATURE service failed to send a CONNECT message failed after media connection in the pending state.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td>The leg is already invalid on CW pending state.</td>
<td>The call leg was invalid in the pending state.</td>
</tr>
<tr>
<td>Failed to send connect message on CW pending state.</td>
<td>The FEATURE service failed to send a CONNECT message in the pending state.</td>
</tr>
<tr>
<td>Failed to create intsend alerting timer on CW pending state.</td>
<td>The FEATURE service failed to create the ALERTING timer in the pending state.</td>
</tr>
<tr>
<td>The leg may be deleted on CW pending state.</td>
<td>The call leg was deleted in the pending state.</td>
</tr>
<tr>
<td>Received invalid RELEASE message on CW pre-idle state.</td>
<td>An invalid RELEASE message was received in the pre-idle state.</td>
</tr>
<tr>
<td>Received invalid SET message on CW WA state.</td>
<td>An invalid SET message was received in the waiting-for-alerting state.</td>
</tr>
<tr>
<td>Failed to Delete Timer on CW MCH state.</td>
<td>The FEATURE service failed to delete the timer in the multi-call hold state.</td>
</tr>
<tr>
<td>Failed to find active PSTN leg on CW MCH state.</td>
<td>The FEATURE service failed to find an active PSTN leg in the multi-call hold state.</td>
</tr>
<tr>
<td>Failed to Get brother Call leg on CW MCH state.</td>
<td>The FEATURE service failed to obtain a brother leg in the multi-call hold state.</td>
</tr>
<tr>
<td>Failed to create CW SRVCUHACK timer on CW organiser state.</td>
<td>The FEATURE service failed to create the CW SRVCUHACK timer in the S_CW_O state.</td>
</tr>
<tr>
<td>Received invalid CMC_SERVICE_ACK message on CW organiser state.</td>
<td>An invalid CMC_SERVICE_ACK message was received in the S_CW_O state.</td>
</tr>
<tr>
<td>Failed to stop busy tone when receiving service ack(cuh) message on CW CUHPRE state</td>
<td>The FEATURE service failed to stop playing busy tones in the S_CW_CUHPRE state after a SERVICE_ACK (CUH) message was received.</td>
</tr>
<tr>
<td>Failed to stop playing call waiting signal tone.</td>
<td>The FEATURE service failed to stop playing call waiting tones.</td>
</tr>
<tr>
<td>Failed to create CW PLAY_CALL_TONE timer in Play CW Tone process.</td>
<td>The FEATURE service failed to create the CW PLAY_CALL_TONE timer.</td>
</tr>
</tbody>
</table>

Table 131 describes output fields and messages for the **debugging voice ss cw event** command.

**Table 131 Output from the debugging voice ss cw event command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Receive %s message from %s module.</td>
<td>An xxx message was received from the xxx module.</td>
</tr>
<tr>
<td>Receive %s message from %s module in %s state.</td>
<td>An xxx message was received from the xxx module in xxx state.</td>
</tr>
</tbody>
</table>

Table 132 describes output fields and messages for the **debugging voice ss cw fsm** command.

**Table 132 Output from the debugging voice ss cw fsm command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>State changed from %s to %s .</td>
<td>The state changed from xxx to xxx.</td>
</tr>
</tbody>
</table>

Table 133 describes output fields and messages for the **debugging voice ss cw info** command.
### Table 133 Output from the debugging voice ss cw info command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>The CW CCB is initialized in create CW CCB.</td>
<td>The CW CCB was initialized successfully.</td>
</tr>
<tr>
<td>The CW CCB is deleted successfully.</td>
<td>The CW CCB was deleted successfully.</td>
</tr>
<tr>
<td>No find original call leg in judge talking process.</td>
<td>No call leg was found in talk path judgment.</td>
</tr>
<tr>
<td>The talk path number is [%d] in judge talking process.</td>
<td>The number of talk paths in talk path judgment is xxx.</td>
</tr>
<tr>
<td>Both legs are active, in judge talking process.</td>
<td>Both of the two legs were in conversation in talk path judgment.</td>
</tr>
<tr>
<td>Stop playing call waiting signal tone.</td>
<td>The FEATURE service tones were stopped from being played.</td>
</tr>
<tr>
<td>Complete to control playing call waiting signal tone.</td>
<td>The control over announcement play ended.</td>
</tr>
<tr>
<td>Send CMC_SERVICE[CW] message on CW idle state.</td>
<td>A CMC_SERVICE[CW] message was sent in the idle state.</td>
</tr>
<tr>
<td>Play sound on CW idle state when PLAY_SOUND_INTERVAL timer is out.</td>
<td>The FEATURE service tones were played in the idle state when the PLAY_SOUND_INTERVAL timer expired.</td>
</tr>
<tr>
<td>Received CW_FAIL CMC message on CW pending state.</td>
<td>A CW_FAIL message was received in the pending state.</td>
</tr>
<tr>
<td>Received CW_NOTSUPPORT CMC message on CW pending state.</td>
<td>A CW_NOTSUPPORT message was received in the pending state.</td>
</tr>
<tr>
<td>Received CW_OK CMC message on CW pending state.</td>
<td>A CW_OK message was received in the pending state.</td>
</tr>
<tr>
<td>Play sound on CW pending state when received service_ack.</td>
<td>A SERVICE_ACK message was received, and call waiting tones were played.</td>
</tr>
<tr>
<td>Received CH_FAIL CMC message on CW MCH state.</td>
<td>A CH_FAIL message was received in the multi-call hold state.</td>
</tr>
<tr>
<td>Received CH_OK CMC message on CW MCH state.</td>
<td>A CH_OK message was received in the multi-call hold state.</td>
</tr>
<tr>
<td>Received CH_UNSUPPORT CMC message on CW MCH state.</td>
<td>A CH_UNSUPPORT message was received in the multi-call hold state.</td>
</tr>
</tbody>
</table>

### Table 134 Output from the debugging voice ss cw timer command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Timer created, TimerId : %s, Timer length : %d.</td>
<td>A timer was created. The timer ID is xxx, and the timer length is xxx milliseconds.</td>
</tr>
<tr>
<td>Timer deleted, TimerId : %s.</td>
<td>A timer was deleted. The timer ID is xxx.</td>
</tr>
<tr>
<td>Timer timeout, TimerId : %s.</td>
<td>A timer expired. The timer ID is xxx.</td>
</tr>
</tbody>
</table>

### Examples

# Enable all types of debugging for call waiting on Router A. Output similar to the following example is generated when the following conditions exist:
Telephone A, Telephone B, and Telephone C are attached to Router A, Router B, and Router C, respectively.

Telephone A is enabled with call waiting.

Telephone B places a call to Telephone A that is conversation with Telephone C.

```shell
<RouterA> debugging voice ss cw all
SS CW all debugging switches are on
<RouterA>
*Nov 20 03:10:02:712 2005 RouterA SS/7/VOICE:
  CW_EVT: [0x000a0001] Receive CMC_RELEASE message from SA module.
// A CMC_RELEASE message was received for the new call (because Telephone A was busy).
```

```shell
*Nov 20 03:10:02:713 2005 RouterA SS/7/VOICE:
  CW_INFO: [0x000a0001] Both legs are active, in judge talking process.
// The conditions for enabling call waiting were met.
```

```shell
*Nov 20 03:10:02:713 2005 RouterA SS/7/VOICE:
  CW_INFO: [0x000a0001] The CW CCB is initialized in create CW CCB.
The call waiting (CW) CCB was initialized.
```

```shell
*Nov 20 03:10:02:713 2005 RouterA SS/7/VOICE:
  CW_EVT: [0x000a0001] Receive CMC_RELEASE message from SA module in IDLE state.
// The RELEASE message was processed in the idle state.
```

```shell
*Nov 20 03:10:02:713 2005 RouterA SS/7/VOICE:
  CW_INFO: [0x000a0001] Send CMC_SERVICE[CW] message on CW idle state.
// The CW service was originated. Internally, CW origination was the processing of a SERVICE (CW) message.
```

```shell
*Nov 20 03:10:02:713 2005 RouterA SS/7/VOICE:
  CW_FSM: [0x000a0001] State changed from S_CW_IDLE to S_CW_PENDING.
// The state of the CW CCB changed from idle to pending.
```

```shell
*Nov 20 03:10:02:715 2005 RouterA SS/7/VOICE:
  CW_EVT: [0x00090002] Receive CMC_SERVICE_ACK message from SA module.
*Nov 20 03:10:02:715 2005 AR30-40 SS/7/VOICE:
  CW_EVT: [0x00090002] Receive CMC_SERVICE_ACK message from SA module in PENDING state.
// A CMC_SERVICE_ACK message was received.
```

```shell
*Nov 20 03:10:02:715 2005 RouterA SS/7/VOICE:
  CW_INFO: [0xffffffff] Received CW_OK CMC message on CW pending state.
// A CW_OK CMC message was received from the peer module, which indicated that CW could be originated.
```

```shell
*Nov 20 03:10:02:857 2005 RouterA SS/7/VOICE:
  CW_INFO: [0x00090002] Play sound on CW pending state when received service_ack.
```

259
// Announcement play needed to be controlled.

*Nov 20 03:10:03:08 2005 RouterA SS/7/VOICE:
  CW_INFO: [0x00090003] Play call waiting signal tone.

// The FEATURE service tones were played.

*Nov 20 03:10:03:109 2005 RouterA SS/7/VOICE:
  CW_TMR: [0x00090003] Timer created, Timer ID: TIMER_SS_CW_PLAY_SOUND_INTERVAL, Timer length: 400.

// A 400-ms timer was started to control the interval for playing call waiting tones.

*Nov 20 03:10:03:260 2005 RouterA SS/7/VOICE:
  CW_EVT: [0x00090003] Receive TIMER_SS_CW_PLAY_SOUND_INTERVAL message from SA module.
*Nov 20 03:10:03:361 2005 RouterA SS/7/VOICE:
  CW_EVT: [0x00090003] Receive TIMER_SS_CW_PLAY_SOUND_INTERVAL message from SA module in PENDING state.

// A TIMER_SS_CW_PLAY_SOUND_INTERVAL message was received in the pending state.

*Nov 20 03:10:03:512 2005 RouterA SS/7/VOICE:
  CW_TMR: [0x00090003] Timer timeout, Timer ID: TIMER_SS_CW_PLAY_SOUND_INTERVAL.

// The timer expired.

*Nov 20 03:10:03:613 2005 RouterA SS/7/VOICE:
  CW_INFO: [0x00090003] Play Sound on CW pending state when TIMER_SS_CW_PLAY_SOUND_INTERVAL out.

// The FEATURE service tones were played again.

*Nov 20 03:10:03:774 2005 RouterA SS/7/VOICE:
  CW_INFO: [0x00090003] Stop playing call waiting signal tone.

// The FEATURE service tones stopped.

// The following was the process of playing call waiting tones repeatedly.

*Nov 20 03:10:03:875 2005 RouterA SS/7/VOICE:
  CW_TMR: [0x00090003] Timer created, Timer ID: TIMER_SS_CW_PLAY_SOUND_INTERVAL, Timer length: 400.
*Nov 20 03:10:04:26 2005 RouterA SS/7/VOICE:
  CW_EVT: [0x00090003] Receive TIMER_SS_CW_PLAY_SOUND_INTERVAL message from SA module.
*Nov 20 03:10:04:127 2005 RouterA SS/7/VOICE:
  CW_EVT: [0x00090003] Receive TIMER_SS_CW_PLAY_SOUND_INTERVAL message from SA module in PENDING state.
*Nov 20 03:10:04:278 2005 RouterA SS/7/VOICE:
  CW_TMR: [0x00090003] Timer timeout, Timer ID: TIMER_SS_CW_PLAY_SOUND_INTERVAL.
*Nov 20 03:10:04:429 2005 RouterA SS/7/VOICE:
  CW_INFO: [0x00090003] Play Sound on CW pending state when TIMER_SS_CW_PLAY_SOUND_INTERVAL out.
*Nov 20 03:10:04:580 2005 RouterA SS/7/VOICE:
  CW_INFO: [0x00090003] Play call waiting signal tone.
The call waiting tones were played for the set times.

Enable all types of debugging for call waiting on Router A. Output similar to the following example is generated when the following conditions exist:

- Telephone A, Telephone B, and Telephone C are attached to Router A, Router B, and Router C, respectively.
- Telephone A is enabled with call waiting.
- Telephone B places a call to Telephone A that is conversation with Telephone C. Then, the user Telephone A presses hookflash to answer the call from Telephone B.

```bash
<RouterA> debugging voice ss cw all
SS CW all debugging switches are on
<RouterA>
```

---

### A CMC_INFORMATION (HOOKFLASH) message was received.

```bash
*Nov 20 05:29:41:288 2005 RouterA SS/7/VOICE:
  CW_EVT: [0x000b0000] Receive CMC_INFORMATION message from SA module.
```

---

### The CMC_INFORMATION message was processed in the pending state.

```bash
*Nov 20 05:29:41:288 2005 RouterA SS/7/VOICE:
  CW_EVT: [0x000b0000] Receive CMC_INFORMATION message from SA module in PENDING state.
```

---

### The state of the CW module changed from pending to multi-call hold.

```bash
*Nov 20 05:29:41:295 2005 RouterA SS/7/VOICE:
  CW_EVT: [0x000b0001] Receive CMC_SERVICE_ACK message from SA module.
```

---

### A CMC_SERVICE_ACK message was received.

```bash
*Nov 20 05:29:41:296 2005 RouterA SS/7/VOICE:
```
CW_EVT: [0x000b0001] Receive CMC_SERVICE_ACK message from SA module in MCH state.

// The CMC_SERVICE_ACK message was processed in the multi-call hold state.

*Nov 20 05:29:41:296 2005 RouterA SS/7/VOICE:
   CW_INFO: [0x000b0001] Received CH_OK CMC message on CW MCH state.
   // A CH_OK CMC message was received.

*Nov 20 05:29:41:296 2005 RouterA SS/7/VOICE:
   CW_FSM: [0x000b0001] State changed from S_CW_MCH to S_CW_IDLE.
   // The CW service ended, and the state changed to idle.

*Nov 20 05:29:41:296 2005 RouterA SS/7/VOICE:
   CW_INFO: [0xffffffff] The CW CCB is deleted successfully.
   // The CW CCB was blocked.

debugging voice ss dr

Use debugging voice ss dr to enable door opening control service debugging.

Use undo debugging voice ss dr to disable door opening control service debugging.

Syntax

debugging voice ss dr { all | error | info | timer }
undo debugging voice ss dr { all | error | info | timer }

Default

Door opening control service debugging is disabled.

Views

User view

Default command level

2: System level

Parameters

all: Specifies all types of debugging for the door opening control service.
error: Specifies error debugging.
info: Specifies information debugging.
timer: Specifies timer debugging.

Usage guidelines

Table 135 describes output fields and messages for the debugging voice ss dr error command.

Table 135 Output from the debugging voice ss dr error command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Failed to create DR CCB</td>
<td>The door opening control service failed to create the CCB.</td>
</tr>
</tbody>
</table>
Field | Description
--- | ---
It’s not called party input password in Door-Relay service | This error information indicates that the password was entered by the calling party. However, in the door opening control service, the password should be entered by the called party.

Failed to send door-relay command to driver | The door opening control service failed to send the `door-relay` command to the driver.

Table 136 describes output fields and messages for the `debugging voice ss dr info` command.

**Table 136 Output from the debugging voice ss dr info command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>The first password is not *, and invalid to Door-Relay service</td>
<td>The password was invalid, because the first character of the password was not an asterisk (*)</td>
</tr>
</tbody>
</table>

**Examples**

# Enable all types of debugging for door opening control on Router A. Output similar to the following example is generated when the following conditions exist:
- Telephone A and Telephone B are both attached to Router A.
- The `door-relay password *012#` command is configured on the voice subscriber line for Telephone B.
- The password *012# is entered on Telephone B after Telephone A establishes a call to Telephone B.

```
<RouterA> debugging voice ss dr all
<RouterA>
*Sep 28 14:48:24:811 2009 RouterA SS/7/VOICE:
  DR_INFO: [0x00000001] Create a DR CCB.
  // The CCB of the door opening control service was created.

*Sep 28 14:48:24:811 2009 RouterA SS/7/VOICE:
  DR_TMR: [0x00000001] Timer created, Timer ID: TIMER_SS_DR_DIALINTERVAL, Timer length: 10000.
  // The timer with the ID of TIMER_SS_DR_DIALINTERVAL and the length of 10000 was created.

*Sep 28 14:48:25:101 2009 RouterA SS/7/VOICE:
  DR_TMR: [0x00000001] Timer deleted, Timer ID: TIMER_SS_DR_DIALINTERVAL.
  // The timer with the ID of TIMER_SS_DR_DIALINTERVAL was deleted.

*Sep 28 14:48:25:051 2009 RouterA SS/7/VOICE:
  DR_TMR: [0x00000001] Timer created, Timer ID: TIMER_SS_DR_DIALINTERVAL, Timer length: 10000.

*Sep 28 14:48:25:351 2009 RouterA SS/7/VOICE:
  DR_TMR: [0x00000001] Timer deleted, Timer ID: TIMER_SS_DR_DIALINTERVAL.
```
*Sep 28 14:48:25:411 2009 RouterA SS/7/VOICE:  
DR_TMR: [0x00000001] Timer created, Timer ID: TIMER_SS_DR_DIALINTERVAL, Timer length: 10000.

*Sep 28 14:48:25:541 2009 RouterA SS/7/VOICE:  
DR_TMR: [0x00000001] Timer deleted, Timer ID: TIMER_SS_DR_DIALINTERVAL.

*Sep 28 14:48:25:651 2009 RouterA SS/7/VOICE:  
DR_TMR: [0x00000001] Timer created, Timer ID: TIMER_SS_DR_DIALINTERVAL, Timer length: 10000.

*Sep 28 14:48:25:681 2009 RouterA SS/7/VOICE:  
DR_TMR: [0x00000001] Timer deleted, Timer ID: TIMER_SS_DR_DIALINTERVAL.

*Sep 28 14:48:26:041 2009 RouterA SS/7/VOICE:  
DR_INFO: [0x00000001] The password is matched successfully.  
//The password was matched.

*Sep 28 14:48:26:042 2009 RouterA SS/7/VOICE:  
DR_INFO: [0x00000001] Received door-relay password is *012#.
// The received password for the door opening control service was *012#.

*Sep 28 14:48:26:042 2009 RouterA SS/7/VOICE:  
DR_INFO: [0xffffffff] DR CCB is deleted successfully.  
// The CCB of the door opening control service was deleted successfully.

debugging voice ss ft

Use debugging voice ss ft to enable FEATURE service debugging.  
Use undo debugging voice ss ft to disable FEATURE service debugging.

Syntax

debugging voice ss ft { all | error | event | fsm | info | timer }
undo debugging voice ss ft { all | error | event | fsm | info | timer }

Default

FEATURE service debugging is disabled.

Views

User view

Default command level

2: System level

Parameters

all: Specifies all types of debugging for the FEATURE service.
error: Specifies error debugging.
event: Specifies event debugging.
fsm: Specifies finite state machine debugging.
**Usage guidelines**

Table 137 describes output fields and messages that for the *debugging voice ss ft error* command.

**Table 137 Output from the debugging voice ss ft error command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>The call leg parameter is null in invalid FT CCB state.</td>
<td>The call leg parameter was null in the invalid feature (FT) CCB state.</td>
</tr>
<tr>
<td>The call leg’s father CMC CCB parameter is null in invalid FT CCB state.</td>
<td>The call leg’s father CMC CCB parameter was null in the invalid FT CCB state.</td>
</tr>
<tr>
<td>The message set type parameter is null in invalid FT CCB state.</td>
<td>The message set type parameter was null in the invalid FT CCB state.</td>
</tr>
<tr>
<td>The message type parameter is null in invalid FT CCB state.</td>
<td>The message type parameter was null in the invalid FT CCB state.</td>
</tr>
<tr>
<td>The message parameter is null in invalid FT CCB state.</td>
<td>The message parameter was null in the invalid FT CCB state.</td>
</tr>
<tr>
<td>The return information parameter is null in invalid FT CCB state.</td>
<td>The return information parameter was null in the invalid FT CCB state.</td>
</tr>
<tr>
<td>Failed to create FT CCB when received CMC_SERVICE message.</td>
<td>The FEATURE service failed to create the FT CCB failed after a CMC_SERVICE message was received.</td>
</tr>
<tr>
<td>Failed to send feature message.</td>
<td>The FEATURE service failed to send a FEATURE message failed.</td>
</tr>
<tr>
<td>Failed to get feature index.</td>
<td>The FEATURE service failed to obtain the feature index failed.</td>
</tr>
<tr>
<td>Failed to get feature message send type.</td>
<td>The FEATURE service failed to obtain a way of sending a FEATURE message failed.</td>
</tr>
<tr>
<td>Receive unknown CMC_SERVICE message from SA module in invalid FT CCB state.</td>
<td>An unknown CMC_SERVICE message was received from the SA module in the invalid FT CCB state.</td>
</tr>
<tr>
<td>Receive unknown CMC_SERVICE_ACK message from SA module in invalid FT CCB state.</td>
<td>An unknown CMC_SERVICE_ACK message was received from the SA module in the invalid FT CCB state.</td>
</tr>
<tr>
<td>Receive unknown CMC message from SA module in invalid FT CCB state.</td>
<td>An unknown CMC message was received from the SA module in the invalid FT CCB state.</td>
</tr>
<tr>
<td>Receive unknown TIMER message from SA module in invalid FT CCB state.</td>
<td>An unknown TIMER message was received from the SA module in the invalid FT CCB state.</td>
</tr>
<tr>
<td>Receive unknown ACCP message from SA module in invalid FT CCB state.</td>
<td>An unknown ACCP message was received from the SA module in the invalid FT CCB state.</td>
</tr>
<tr>
<td>Receive unknown message type from SA module in invalid FT CCB state.</td>
<td>An unknown message was received from the SA module in the invalid FT CCB state.</td>
</tr>
<tr>
<td>The FT CCB parameter is null in SENDSRV state.</td>
<td>The FT CCB parameter was null in the SENDSRV state.</td>
</tr>
<tr>
<td>The SIP CCB is null when proceed barge in service.</td>
<td>The SIP CCB was null in the processing of the barge-in service.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Attach IPP CCB failed when proceed barge in service!</td>
<td>The FEATURE service failed to attach the IPP CCB in the processing of the barge-in service.</td>
</tr>
<tr>
<td>The IPP CCB is null when proceed barge in service.</td>
<td>The IPP CCB was null in the processing of the barge-in service.</td>
</tr>
<tr>
<td>The SIP CCB is null when proceed block barge in service.</td>
<td>The SIP CCB was null in the processing of the block-barge-in service.</td>
</tr>
<tr>
<td>Attach IPP CCB failed when proceed block barge in service!</td>
<td>The FEATURE service failed to attach the IPP CCB in the processing of the block-barge-in service.</td>
</tr>
<tr>
<td>Detach IPP CCB failed when proceed block barge in service!</td>
<td>The FEATURE service failed to detach the IPP CCB in the processing of the block-barge-in service.</td>
</tr>
<tr>
<td>The IPP CCB is null when proceed block barge in service.</td>
<td>The IPP CCB was null in the processing of the block-barge-in service.</td>
</tr>
<tr>
<td>Stop DTMF detect error.</td>
<td>The FEATURE service failed to disable DTMF detection. In this case, digits could still be collected after the match ended.</td>
</tr>
<tr>
<td>Failed to create play sound interval timer.</td>
<td>The FEATURE service failed to create an interval timer for playing announcements.</td>
</tr>
<tr>
<td>Play signal tone error.</td>
<td>The FEATURE service failed to play signal tones failed.</td>
</tr>
<tr>
<td>The IfIndex is invalid when received service_ack(feature).</td>
<td>The interface index was invalid when a SERVICE_ACK (feature) message was received.</td>
</tr>
<tr>
<td>Invalid leg type when receive CMC_ALERTING message in SENDSRV state.</td>
<td>The call leg type was invalid when a CMC_ALERTING message was received in the SENDSRV state.</td>
</tr>
<tr>
<td>Failed to create wait connect timer in SENDSRV state.</td>
<td>The FEATURE service failed to create the CONNECT timer in the SENDSRV state.</td>
</tr>
<tr>
<td>Invalid leg type when receive CMC_CONNECT message in SENDSRV state.</td>
<td>The call leg type was invalid when a CMC_CONNECT message was received in the SENDSRV state.</td>
</tr>
<tr>
<td>Failed to send reverse feature code when proceed FT_WAITSRVACK timer message.</td>
<td>The FEATURE service failed to send a reverse feature code in the processing of the FT_WAITSRVACK timer.</td>
</tr>
<tr>
<td>Failed to reverse feature code when proceed FT_WAITSRVACK timer message.</td>
<td>The FEATURE service failed to reverse the feature code in the processing of the FT_WAITSRVACK timer. In this case, no reverse feature code was sent.</td>
</tr>
<tr>
<td>Failed to stop signal tone.</td>
<td>The FEATURE service failed to stop playing signal tones.</td>
</tr>
<tr>
<td>Failed to send feature message from FT module in PRERELEASE state.</td>
<td>The FEATURE service failed to send a FEATURE message from the FT module in the pre-release state.</td>
</tr>
<tr>
<td>Can not find CMC CCB when create FT CCB.</td>
<td>No corresponding CMC CCB was found during the creation of the FT CCB.</td>
</tr>
<tr>
<td>The FT service CCB in CMC CCB is existed when create FT CCB.</td>
<td>The FT CCB associated with the CMC CCB already existed before the creation of the FT CCB.</td>
</tr>
<tr>
<td>The call state is error when create FT CCB.</td>
<td>The call state was incorrect in the creation of the FT CCB.</td>
</tr>
<tr>
<td>Allocate memory for FT service CCB failed when create FT CCB.</td>
<td>The FEATURE service failed to allocate memory to the FT CCB in creating the FT CCB.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Can not find cmc CCB when delete FT CCB.</td>
<td>The FT CCB to be deleted could not be found.</td>
</tr>
<tr>
<td>Invalid leg in call out state when send service(feature) message.</td>
<td>The leg parameter was invalid in the callout state when a SERVICE (FEATURE) message was sent.</td>
</tr>
<tr>
<td>Failed to create brother leg when send feature message.</td>
<td>The FEATURE service failed to create a brother leg when a FEATURE message was sent.</td>
</tr>
<tr>
<td>Failed to create message table to brother leg when send feature message.</td>
<td>The FEATURE service failed to create a message triggering table for a brother leg when a FEATURE message was sent.</td>
</tr>
<tr>
<td>Failed to get new talk path when send feature message.</td>
<td>The FEATURE service failed to obtain a new talk path when a FEATURE message was sent.</td>
</tr>
<tr>
<td>Failed to create wait service_ack timer when send feature message.</td>
<td>The FEATURE service failed to create the SERVICE_ACK timer when a FEATURE message was sent.</td>
</tr>
<tr>
<td>Failed to block barge in when silent monitoring other calling.</td>
<td>The FEATURE service failed to block barge-in when silent monitoring was enabled.</td>
</tr>
<tr>
<td>Failed to get brother call leg when send feature message.</td>
<td>The FEATURE service failed to obtain a brother leg when a FEATURE message was sent.</td>
</tr>
<tr>
<td>The feature message send type is invalid when send feature message.</td>
<td>The method for sending a FEATURE message was invalid.</td>
</tr>
</tbody>
</table>

Table 138 describes output fields and messages for the **debugging voice ss ft event** command.

**Table 138 Output from the debugging voice ss ft event command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Receive CMC_SERVICE message from SA module in invalid FT CCB state.</td>
<td>A CMC_SERVICE message was received from the SA module in the invalid FT CCB state.</td>
</tr>
<tr>
<td>Receive CMC_SERVICE_ACK message from SA module in SENDSRV state.</td>
<td>A CMC_SERVICE_ACK message was received from the SA module in the SENDSRV state.</td>
</tr>
<tr>
<td>Send CMC_SERVICE(Feature) message to CMC module in IDLE state.</td>
<td>a CMC_SERVICE (FEATURE) message was sent to the CMC module in the idle state.</td>
</tr>
</tbody>
</table>

Table 139 describes output fields and messages for the **debugging voice ss ft fsm** command.

**Table 139 Output from the debugging voice ss ft fsm command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>State changed from IDLE to SENDSRV.</td>
<td>The state changed from idle to SENDSRV.</td>
</tr>
</tbody>
</table>

Table 140 describes output fields and messages for the **debugging voice ss ft info** command.

**Table 140 Output from the debugging voice ss ft info command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Receive successful response of feature service.</td>
<td>A response to the FEATURE message was received, which indicated that the FEATURE service was successfully registered.</td>
</tr>
</tbody>
</table>
The result in SERVICE_ACK(Feature) message is "fail". The result in the SERVICE_ACK (FEATURE) message was fail.

The fail cause is "Unknown user". The failure cause in the SERVICE_ACK message was unknown user.

The fail cause is "9499". The failure cause in the SERVICE_ACK message was 9499. This cause code was directly displayed because it could not be parsed according to H3C's documents.

Release call in SENDSRV state, the fail cause is "No calls to supervise". The call was released in the SENDSRV state. The failure cause was no call to supervise.

Attach IPP CCB to IfIndex when proceed barge in service. The IPP CCB was attached to an interface in the processing of the barge-in service.

Attach IPP CCB to IfIndex when proceed block barge in service. The IPP CCB to was attached an interface in the processing of the block-barge-in service.

Detach IPP CCB from IfIndex when proceed block barge in service. The IPP CCB was detached from an interface in the processing of the block-barge-in service.

Begin to play result sound. Announcing the result started.

Do not play signal tone. No signal tones were played if a FEATURE_ACK message was received during conversation.

Can not reverse the operation when received CMC_RELEASE message in SENDSRV state. A CMC_RELEASE message was received in the SENDSRV state, and if the feature code could not be reversed, the call was directly released.

Release call when received CMC_RELEASE message in SENDSRV state. A CMC_RELEASE message was received in the SENDSRV state, and the call was released.

Send reverse feature code when proceed FT_WAITSRVACK timer message. A reverse feature code was sent in the processing of the FT_WAITSRVACK timer.

Stop playing result sound. Announcing the result stopped.

Create a FT CCB. An FT CCB was created.

Release the service leg source when delete FT CCB. Related leg resources were released in deleting the FT CCB.

Delete FT CCB. The FT CCB was deleted.

The feature code in the CMC_SERVICE message is "*465". The feature code in the CMC_SERVICE message was 485.

Table 141 describes output fields and messages for the debugging voice ss ft timer command.

Table 141 Output from the debugging voice ss ft timer command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Timer created, TimerId : TIMER_SSF_WAITSRVACK, Timer length : 20000.</td>
<td>A 20000-millisecond timer was created for a SERVICE_ACK message.</td>
</tr>
<tr>
<td>Timer deleted, TimerId : TIMER_SSF_WAITCONNECT.</td>
<td>The timer for a CONNECT message was deleted.</td>
</tr>
</tbody>
</table>
Timer timeout, TimerId: TIMER_SS_FT_PLAYSOUND_INTERVAL.
The interval timer for playing announcements expired. This timer was used for control over announcement play.

Examples

Output similar to the following example is generated when the user at Telephone A attached to Router A picks up the headset and dials *446# for Do Not Disturb Toggle:

# Enable debugging for FEATURE service errors on Router A.
<RouterA> debugging voice ss ft event
SS FT event debugging switch is on
<RouterA>
*May 14 15:03:35:146 2007 RouterA SS/7/VOICE:
  FT_EVT: [0x00070000] Receive CMC_SERVICE message from SA module in invalid FT CCB state.
  // A CMC_SERVICE message was received from the SA module in the invalid FT CCB state.

*May 14 15:03:35:146 2007 RouterA SS/7/VOICE:
  FT_EVT: [0x00070000] Send CMC_SERVICE(Feature) message to CMC module in IDLE state.
  // A CMC_SERVICE (FEATURE) message was sent to the CMC module in the idle state.

*May 14 15:03:35:297 2007 RouterA SS/7/VOICE:
  FT_EVT: [0x00070001] Receive CMC_SERVICE_ACK message from SA module in SENDSRV state.
  // A CMC_SERVICE_ACK message was received in the SENDSRV state.

  // The service was set successfully, and the setting result was announced. A timer was used to control announcement play, and announcements were played five times.
*May 14 15:03:35:492 2007 RouterA SS/7/VOICE:
  FT_EVT: [0x00070001] Receive TIMER_SS_FT_PLAYSOUND_INTERVAL message from SA module in SENDSRV state.
*May 14 15:03:35:692 2007 RouterA SS/7/VOICE:
  FT_EVT: [0x00070001] Receive TIMER_SS_FT_PLAYSOUND_INTERVAL message from SA module in SENDSRV state.
*May 14 15:03:35:892 2007 RouterA SS/7/VOICE:
  FT_EVT: [0x00070001] Receive TIMER_SS_FT_PLAYSOUND_INTERVAL message from SA module in SENDSRV state.
*May 14 15:03:36:92 2007 RouterA SS/7/VOICE:
  FT_EVT: [0x00070001] Receive TIMER_SS_FT_PLAYSOUND_INTERVAL message from SA module in SENDSRV state.
*May 14 15:03:36:292 2007 RouterA SS/7/VOICE:
  FT_EVT: [0x00070001] Receive TIMER_SS_FT_PLAYSOUND_INTERVAL message from SA module in SENDSRV state.

# Enable debugging for FEATURE service finite state machines on Router A.
<RouterA> debugging voice ss ft fsm
SS FT FSM debugging switch is on
<RouterA>
*May 14 15:15:53:484 2007 RouterA SS/7/VOICE:
  FT_FSM: [0x00080000] State changed from IDLE to SENDSRV.
The state changed from idle to SENDSRV.

Enable debugging for FEATURE service information on Router A.

<RouterA> debugging voice ss ft info
SS FT information debugging switch is on
<RouterA>

*May 14 15:18:27:894 2007 RouterA SS/7/VOICE:
  FT_INFO: [0x00090000] Create a FT CCB.

An FT CCB was created.

*May 14 15:18:27:894 2007 RouterA SS/7/VOICE:
  FT_INFO: [0x00090000] The feature code "*446" match to service "do not disturb".

The feature code *446 matched the do-not-disturb service.

*May 14 15:18:27:895 2007 RouterA SS/7/VOICE:
  FT_INFO: [0x00090000] Send the feature code by service message.

A feature code was sent in a SERVICE message.

*May 14 15:18:27:895 2007 RouterA SS/7/VOICE:
  FT_INFO: [0x00090000] The feature code in the CMC_SERVICE message is "*446".

The feature code in the CMC_SERVICE message was *446.

*May 14 15:18:27:909 2007 RouterA SS/7/VOICE:
  FT_INFO: [0x00090001] Receive successful response of feature service.

A successful response for the FEATURE service was received.

*May 14 15:18:27:909 2007 RouterA SS/7/VOICE:
  FT_INFO: [0x00090001] Begin to play result sound.

Announcing the result started.

*May 14 15:18:28:902 2007 RouterA SS/7/VOICE:
  FT_INFO: [0x00090001] Stop playing result sound.

Announcing the result stopped.

*May 14 15:18:28:902 2007 RouterA SS/7/VOICE:
  FT_INFO: [0x00090001] Release the service leg source when delete FT CCB.

The service leg was released during deletion of the FT CCB.

*May 14 15:18:28:903 2007 RouterA SS/7/VOICE:
  FT_INFO: [0x00090001] Delete FT CCB.

The FT CCB was deleted.

Enable debugging for FEATURE service timers on Router A.

<RouterA> debugging voice ss ft timer
SS FT timer debugging switch is on
<RouterA>
*May 14 15:28:36:787 2007 RouterA SS/7/VOICE:
  FT_TMR: [0x000a0000] Timer created, Timer ID: TIMER_SS_FT_WAITSRVACK, Timer length: 20000.
  // A 20000-ms SS_FT_WAITSRVACK timer was created.

*May 14 15:28:36:806 2007 RouterA SS/7/VOICE:
  FT_TMR: [0x000a0001] Timer deleted, Timer ID: TIMER_SS_FT_WAITSRVACK.
  // The SS_FT_WAITSRVACK timer was deleted.
  // The following repeatedly announced the successful setting of a feature service five times by repeatedly creating and deleting such a timer five times.

*May 14 15:28:37:00 2007 RouterA SS/7/VOICE:
  FT_TMR: [0x000a0001] Timer created, Timer ID: TIMER_SS_FT_PLAY_SOUND_INTERVAL, Timer length: 200.
  // A 200-ms interval timer for playing an announcement was created.

*May 14 15:28:37:00 2007 RouterA SS/7/VOICE:
  FT_TMR: [0x000a0001] Timer timeout, Timer ID: TIMER_SS_FT_PLAY_SOUND_INTERVAL.
  // The interval timer expired.

Use debugging voice ss hg to enable hunt group debugging.
Use undo debugging voice ss hg to disable hunt group debugging.

Syntax

devbugging voice ss hg { all | error | event | info }
undo debugging voice ss hg { all | error | event | info }
Default

Hunt group debugging is disabled.

Views

User view

Default command level

2: System level

Parameters

all: Specifies all types of debugging for hunt group.
error: Specifies error debugging.
event: Specifies event debugging.
info: Specifies information debugging.

Usage guidelines

Table 142 describes output fields and messages for the debugging voice ss hg error command.

Table 142 Output from the debugging voice ss hg error command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>The Leg parameter received is null.</td>
<td>The pointer to the current leg was null.</td>
</tr>
<tr>
<td>The CCB parameter is null.</td>
<td>The pointer to the CCB was null.</td>
</tr>
<tr>
<td>The brother leg is null.</td>
<td>The pointer to the brother leg was null.</td>
</tr>
<tr>
<td>Failed to fill in the in leg setup information.</td>
<td>The hunt group module failed to update the call leg information.</td>
</tr>
<tr>
<td>The message type parameter is illegal.</td>
<td>The message type was invalid.</td>
</tr>
<tr>
<td>The call information table in CCB is invalid.</td>
<td>The pointer to the call information table was null.</td>
</tr>
<tr>
<td>Failed to get call information table by index.</td>
<td>The hunt group module failed to obtain a call information table by index.</td>
</tr>
</tbody>
</table>

Table 143 describes output fields and messages for the debugging voice ss hg event command.

Table 143 Output from the debugging voice ss hg event command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Receive CMC_RELEASE message from SA module.</td>
<td>A CMC_RELEASE message was received from the SA module.</td>
</tr>
</tbody>
</table>

Table 144 describes output fields and messages for the debugging voice ss hg info command.

Table 144 Output from the debugging voice ss hg info command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create a setup message and send it.</td>
<td>A SETUP message was sent.</td>
</tr>
<tr>
<td>Receive message from in leg from SA.</td>
<td>A message was received from the input leg.</td>
</tr>
<tr>
<td>The release cause is normal clearing.</td>
<td>A NORMAL_CLEARING message was received.</td>
</tr>
<tr>
<td>The out leg is ps type.</td>
<td>The output leg was a PS leg.</td>
</tr>
</tbody>
</table>
Field | Description
--- | ---
Receive release message from SA, but is of wrong type. | A RELEASE message was received.
Can not get the new entity. | The hunt group module failed to obtain the index of a new entity.
Can not find the new entity. | The hunt group module failed to find an available entity.

Examples

# Enable all types of debugging for the hunt group on Router A. Output similar to the following example is generated when Telephone B attempts to call the number 3000 under these conditions:
- Telephone A and Telephone C are both attached to Router A. Telephone B is attached to Router B.
- Router B has two voice entities (1000 and 1001) to reach the telephone number 3000:
  - Entity 1000 has a priority of 1, and it is bound to line 8/0.
  - Entity 1001 has a priority of 3, and it is bound to line 8/1.
- Hunt group is enabled on the voice subscriber line for Telephone A.
- Telephone A is in off-hook state.
<RouterA> debugging voice ss hg all
SS HG all debugging switches are on
<RouterA>

// Voice entity 1000 was first matched according to the dial plan. However, Telephone A was off-hook and unavailable.
*Nov 20 06:44:25:533 2005 RouterA SS/7/VOICE:
HG_EVT: [0x00110001] Receive CMC_RELEASE message from SA module.

// A CMC_RELEASE message was received.

*Nov 20 06:44:25:533 2005 RouterA SS/7/VOICE:
HG_INFO: [0x00110001] The CMC_RELEASE message's release cause: No idle voice port!

// The release cause was no idle voice port.

*Nov 20 06:44:25:533 2005 RouterA SS/7/VOICE:
HG_INFO: [0x00110000] Create a setup message and send it.

// Then voice entity 1001 was matched, and a new call was originated.

debugging voice ss mwi

Use debugging voice ss mwi to enable MWI debugging.

Use undo debugging voice ss mwi to disable MWI debugging.

Syntax

debugging voice ss mwi {all | error | event | info}
undo debugging voice ss mwi { all | error | event | info}

Default

MWI debugging is disabled.
Views

User view

Default command level

2: System level

Parameters

all: Specifies all types of debugging for message waiting indication (MWI).
error: Specifies error debugging.
event: Specifies event debugging.
info: Specifies information debugging.

Usage guidelines

Table 145 describes output fields and messages for the debugging voice ss mwi error command.

Table 145 Output from the debugging voice ss mwi error command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MessagesAccount is invalid.</td>
<td>The message account was invalid.</td>
</tr>
<tr>
<td>Failed to process NOTIFY in SUB or MWI model.</td>
<td>MWI failed to process the NOTIFY message in SUB or MWI model.</td>
</tr>
<tr>
<td>The REALM is not matchable.</td>
<td>The REALM field was not matched.</td>
</tr>
<tr>
<td>Failed to start subscribe.</td>
<td>MWI failed to propose a subscription.</td>
</tr>
<tr>
<td>Failed to stop subscribe.</td>
<td>MWI failed to withdraw a subscription.</td>
</tr>
<tr>
<td>Failed to add item of number information.</td>
<td>MWI failed to add the number item.</td>
</tr>
<tr>
<td>Failed to add IfIndex information.</td>
<td>MWI failed to add the interface index information.</td>
</tr>
<tr>
<td>Failed to find IfIndex information.</td>
<td>MWI failed to find the interface index information.</td>
</tr>
<tr>
<td>Failed to update AuthInfo.</td>
<td>MWI failed to update the authentication information.</td>
</tr>
<tr>
<td>Failed to mark IfIndex</td>
<td>MWI failed to mark the interface index.</td>
</tr>
</tbody>
</table>

Table 146 describes output fields and messages for the debugging voice ss mwi event command.

Table 146 Output from the debugging voice ss mwi event command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>The server of MWI is disable.</td>
<td>MWI server was disabled.</td>
</tr>
<tr>
<td>The mode of NOTIFY(MWI) is XX, the substate is XXX.</td>
<td>The processing mode of the NOTIFY message was xx, and the state of the subscription was xxx.</td>
</tr>
<tr>
<td>Received response for RESUBSCRIBE or UNSUBSCRIBE from stack.</td>
<td>The response to RESUBSCRIBE or UNSUBSCRIBE from the stack was received.</td>
</tr>
<tr>
<td>Start the subscription.</td>
<td>A subscription was proposed.</td>
</tr>
<tr>
<td>Stop the subscription.</td>
<td>A subscription was withdrawn.</td>
</tr>
<tr>
<td>Received XX event in XXX state.</td>
<td>The xx event was processed in xxx state.</td>
</tr>
</tbody>
</table>
Reach the max redirect times limit.  
The maximum number of redirection times was reached.

Table 147 describes output fields and messages for the **debugging voice ss mwi info** command.

### Table 147 Output from the debugging voice ss mwi info command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>The IfIndex is XXX.</td>
<td>The interface index was xxx.</td>
</tr>
<tr>
<td>Needed to update vpIndex of entity.</td>
<td>MWI needed to update the virtual path index of the entity is needed.</td>
</tr>
<tr>
<td>Handle undo enable MWI server.</td>
<td>The configuration of the MWI server was being removed.</td>
</tr>
<tr>
<td>Needed to mark all playtone of IfIndex.</td>
<td>MWI needed to mark all play tones of interface index.</td>
</tr>
<tr>
<td>Can not find IfIndex information.</td>
<td>Interface index information was not found.</td>
</tr>
</tbody>
</table>

**Examples**

# Enable all types of debugging for MWI. When the telephone at 2001 sends a SUBSCRIBE to the server, output similar to the following example is generated:

```plaintext
<Sysname> debug voice ss mwi all
Enable MWI all debugging functions
<Sysname>
*Jan 25 10:05:24:02 2008 Sysname SIP/7/VOICE:
MWI_Info: Handle add IfIndex: The IfIndex is 2883968.
*Jan 25 10:05:24:02 2008 Sysname SIP/7/VOICE:
  // An interface index of 2883968 was added.

MWI_Info: Handle add number: The entity index is 2001.
  // Adding a number of 2001 was being handled.

*Jan 25 10:05:24:02 2008 Sysname SIP/7/VOICE:
MWI_Info: Handle add number[0xffffffff]: Need to add new item of number information. The entity index is 2001.
  // A new number of 2001 needed to be added.

*Jan 25 10:05:24:03 2008 Sysname SIP/7/VOICE:
MWI_Info: Handle add number[0x0000]: Succeed to add item of number information. The number is 2001.
  // Number 2001 was successfully added.

*Jan 25 10:05:24:03 2008 Sysname SIP/7/VOICE:
MWI_Event: Dispatch to FSM[0x0000]: Process the event of ADD_NUM in IDLE state.
  // The ADD_NUM event was processed in the IDLE state.
```
*Jan 25 10:05:24:45 2008 Sysname SIP/7/VOICE:
MWI_Info: Handle execute timeout: The execute count is 0.

// The Execute timer timed out.

*Jan 25 10:05:24:46 2008 Sysname SIP/7/VOICE:
MWI_Event: Dispatch to FSM[0x0000]: Process the event of ADD_TIMEOUT in WAIT_ADD_TIMEOUT state.

// The ADD_TIMEOUT event was processed in the WAIT_ADD_TIMEOUT state.

*Jan 25 10:05:24:46 2008 Sysname SIP/7/VOICE:
MWI_Info: Handle execute timeout: Delete execute timer.

// The Execute timer timed out and was deleted.

*Jan 25 10:05:24:51 2008 Sysname SIP/7/VOICE:
MWI_Info: Handle subscribe feedback: The index is 0.

// Subscription feedback was being processed.

*Jan 25 10:05:24:145 2008 Sysname SIP/7/VOICE:
MWI_Event: Dispatch to FSM[0x0000]: Process the event of FEEDBACK in WAIT_SUB_FEEDBACK state.

// The FEEDBACK event was received in the WAIT_SUB_FEEDBACK state.

*Jan 25 10:05:24:296 2008 Sysname SIP/7/VOICE:
MWI_Info: Handle NOTIFY(Active or Penging): The number is 2001.

// The NOTIFY (Active or Penging) message was being processed.

debugging voice ss sa

Use debugging voice ss sa to enable service analysis module debugging.
Use undo debugging voice ss sa to disable service analysis module debugging.

Syntax

debugging voice ss sa { all | error | event | info }
undo debugging voice ss sa { all | error | event | info }

Default
Service analysis module debugging is disabled.

Views
User view

Default command level
2: System level

Parameters
all: Specifies all types of debugging for the service analysis module.
error: Specifies error debugging.
**event**: Specifies event debugging.

**info**: Specifies information debugging.

### Usage guidelines

Table 148 describes output fields and messages for the **debugging voice ss sa error** command.

#### Table 148 Output from the debugging voice ss sa error command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Failed to get brother leg</td>
<td>The service analysis module failed to obtain the brother leg of the current leg.</td>
</tr>
<tr>
<td>The type of leg is invalid in checking setup message sending</td>
<td>The leg type was invalid when a SETUP message was sent.</td>
</tr>
<tr>
<td>The leg state is invalid in checking setup message sending</td>
<td>The leg state was invalid when a SETUP message was sent.</td>
</tr>
<tr>
<td>Failed to find the brother leg in checking setup message sending</td>
<td>The service analysis module failed to obtain a brother leg when a SETUP message was sent.</td>
</tr>
<tr>
<td>The leg is invalid in checking leg after message sending</td>
<td>The leg was invalid when a SETUP message was sent.</td>
</tr>
<tr>
<td>The in leg may be released later in checking leg after message sending</td>
<td>The input leg might have been exceptionally released after a SETUP message was sent.</td>
</tr>
<tr>
<td>Failed to find the cmc ccb in checking leg after message sending</td>
<td>The service analysis module failed to find the corresponding CMC CCB in checking the leg validity after a SETUP message was sent.</td>
</tr>
<tr>
<td>Get talk path information error when update DTMF</td>
<td>The service analysis module failed to obtain a talk path in updating the DTMF settings.</td>
</tr>
<tr>
<td>The IF index is invalid in create hash table process</td>
<td>The interface index was invalid in creating a HASH table.</td>
</tr>
<tr>
<td>Get service configuration failed in create hash table process</td>
<td>The service analysis module failed to obtain the service configuration of an interface in creating a HASH table.</td>
</tr>
<tr>
<td>Allocate memory for message item table failed in create hash table process</td>
<td>The service analysis module failed to allocate memory for message entries in creating a HASH table.</td>
</tr>
<tr>
<td>Get message attribute failed in search hash table process</td>
<td>The service analysis module failed to obtain the message attribute in searching the HASH table.</td>
</tr>
</tbody>
</table>

Table 149 describes output fields and messages for the **debugging voice ss sa event** command.

#### Table 149 Output from the debugging voice ss sa event command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Receive %s message from CMC module in XX state.</td>
<td>The SA module received an xxx message in xxx state from the CMC module.</td>
</tr>
<tr>
<td>No matched message item was found, send it to CMC module</td>
<td>The message mismatched the service and was sent to the CMC module.</td>
</tr>
<tr>
<td>Message item matched correctly, send it to XX module</td>
<td>The message matched the service and was sent to the xxx module.</td>
</tr>
</tbody>
</table>
Table 150 describes output fields and messages for the debugging voice ss sa info command.

Table 150 Output from the debugging voice ss sa info command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>No need to attach call ID to VIM</td>
<td>The CALL ID did not need to be bound to the VIM module.</td>
</tr>
<tr>
<td>To updating DTMF isn’t processed in CS to CS call</td>
<td>Updating the DTMF setting was unnecessary for a local call.</td>
</tr>
<tr>
<td>Success to send DTMF capability to SPL</td>
<td>The service analysis module succeeded in sending the DTMF capability to the SPL.</td>
</tr>
<tr>
<td>Message trigger table has been created successfully</td>
<td>A message triggering table was created successfully.</td>
</tr>
<tr>
<td>Message trigger table has been deleted successfully</td>
<td>The message triggering table was deleted.</td>
</tr>
<tr>
<td>The IF index of this leg is invalid, find it from brother leg</td>
<td>The interface index of the input call leg was invalid, and an interface index was to be found from the brother leg.</td>
</tr>
<tr>
<td>The IF index of this leg is valid, SA will create message table by this leg’s IF index</td>
<td>The interface index of the input call leg was invalid, and the SA module created a message triggering table according to the interface index.</td>
</tr>
</tbody>
</table>

Examples

# Enable all types of debugging for the service analysis module on Router A. Output similar to the following example is generated when the following conditions exist:

- Telephone A, Telephone B, and Telephone C are attached to Router A, Router B, and Router C, respectively.
- Telephone A is enabled with call waiting.
- Telephone B places a call to Telephone A that is in conversation with Telephone C.

<RouterA> debugging voice ss sa all
SS SA all debugging switches are on
<RouterA>
*Nov 20 07:42:18:390 2005 RouterA SS/7/VOICE:
SA_INFO: [0x00180000] The IF index of this leg is invalid, find it from brother leg.
// A new call was coming in, and a message triggering table needed to be created. The basis for table creation was found: voice subscriber line.

*Nov 20 07:42:18:390 2005 RouterA SS/7/VOICE:
SA_INFO: [0x00180000] The IF index of brother leg is valid, SA will create message table by brother leg’s IF index.
// A message triggering table was created according to the local voice subscriber line obtained by finding a voice entity.

*Nov 20 07:42:18:390 2005 RouterA SS/7/VOICE:
SA_INFO: [0x00180000] Message trigger table has been created successfully.
// A message triggering table was created successfully.
SA_EVT: [0x00180000] Receive CMC_SETUP message from CMC module in S_IDLE state.
// The SETUP message was processed in the normal state.

*Nov 20 07:42:18:390 2005 RouterA SS/7/VOICE:
SA_EVT: [0x00180000] No matched message item is found, send it to CMC module.
// No matched message was found, and the normal call flow was applied.

*Nov 20 07:42:18:390 2005 RouterA SS/7/VOICE:
SA_INFO: [0x00180001] The brother IF index of this leg is invalid, find it from brother leg.
// A message triggering table would be created for the output leg.

*Nov 20 07:42:18:391 2005 RouterA SS/7/VOICE:
SA_INFO: [0x00180001] The IF index of this leg is valid, SA will create message table by this leg's IF index.
// A message triggering table would be created for the corresponding voice subscriber line of the leg.

*Nov 20 07:42:18:391 2005 RouterA SS/7/VOICE:
SA_INFO: [0x00180001] Message trigger table has been created successfully.
// A message triggering table was created successfully.

*Nov 20 07:42:18:392 2005 RouterA SS/7/VOICE:
SA_EVT: [0x00180001] Receive CMC_RELEASE message from CMC module in S_IDLE state.
// A RELEASE message was received because the current port was busy.

*Nov 20 07:42:18:534 2005 RouterA SS/7/VOICE:
SA_EVT: [0x00180001] Message item matched correctly, send it to CW module.
// The received message was correctly matched and forwarded to the CW module for processing.

debugging voice ss sm

Use debugging voice ss sm to enable service match module debugging.
Use undo debugging voice ss sm to disable service match module debugging.

Syntax

debugging voice ss cm { all | error | event | fsm | info | timer }
undo debugging voice ss cm { all | error | event | fsm | info | timer }

Default

Service match module debugging is disabled.

Views

User view

Default command level

2: System level
Parameters

- **all**: Specifies all types of debugging for the service match module.
- **error**: Specifies error debugging.
- **event**: Specifies event debugging.
- **fsm**: Specifies finite state machine debugging.
- **info**: Specifies information debugging.
- **timer**: Specifies timer debugging.

Usage guidelines

Table 151 describes output fields and messages for the **debugging voice ss sm error** command.

### Table 151: Output from the debugging voice ss sm error command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>The call leg parameter is null when enter into SM module.</td>
<td>The pointer to the call leg was null when the processing of the SM module started.</td>
</tr>
<tr>
<td>The message set type parameter is null when enter into SM module.</td>
<td>The pointer to the message set type was null when the processing of the SM module started.</td>
</tr>
<tr>
<td>The message type parameter is null when enter into SM module.</td>
<td>The pointer to the message type was null when the processing of the SM module started.</td>
</tr>
<tr>
<td>The message parameter is null when enter into SM module.</td>
<td>The pointer to the message was null when the processing of the SM module started.</td>
</tr>
<tr>
<td>The return information parameter is null when enter into SM module.</td>
<td>The pointer to the return information was null when the processing of the SM module started.</td>
</tr>
<tr>
<td>Can not find CMC CCB when enter into SM module.</td>
<td>The pointer to the father CMC CCB of the call leg was null when the processing of the SM module started.</td>
</tr>
<tr>
<td>The SM CCB service state is error when enter into SM module.</td>
<td>The state of the CCB was incorrect when the processing of the SM module started.</td>
</tr>
<tr>
<td>The leg type is invalid when receive CMC_SETUP message in invalid SM CCB state.</td>
<td>The call leg type in the received CMC_SETUP message was invalid when the SM CCB was invalid.</td>
</tr>
<tr>
<td>The setup type in CMC_SETUP message is invalid.</td>
<td>The SETUP type in the CMC_SETUP message is incorrect.</td>
</tr>
<tr>
<td>The service register DTMF number in CMC_SETUP message is invalid.</td>
<td>The DTMF code for service registration is incorrect in the CMC_SETUP message.</td>
</tr>
<tr>
<td>Create SM CCB failed when receive CMC_SETUP message.</td>
<td>The service match module failed to create an SM CCB when receiving a CMC_SETUP message.</td>
</tr>
<tr>
<td>Failed to create wait dial interval timer.</td>
<td>The service match module failed to create a timer waiting for dialing the next digit failed.</td>
</tr>
<tr>
<td>The information type in CMC_INFORMATION message is invalid.</td>
<td>The information type in a CMC_INFORMATION message was incorrect.</td>
</tr>
<tr>
<td>Could not get feature service switch, set feature service switch to false.</td>
<td>The service match module failed to obtain a feature service toggle. By default, a feature service is disabled.</td>
</tr>
<tr>
<td>Create SM CCB failed when receive CMC_INFORMATION message.</td>
<td>The service match module failed to create an SM CCB when receiving a CMC_INFORMATION message.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Failed to create dial interval number in invalid SM CCB state.</td>
<td>The service match module failed to create a timer waiting for dialing the next digit when no SM CCB existed.</td>
</tr>
<tr>
<td>Receive unknown CMC message from SM module in RECVCODE state.</td>
<td>An unknown CMC message was received in the RECVCODE state.</td>
</tr>
<tr>
<td>Receive unknown TIMER message from SM module in RECVCODE state.</td>
<td>An unknown TIMER message was received in the RECVCODE state.</td>
</tr>
<tr>
<td>Receive unknown ACCP message from SM module in RECVCODE state.</td>
<td>An unknown ACCP message was received in the RECVCODE state.</td>
</tr>
<tr>
<td>Receive unknown message type from SM module in RECVCODE state.</td>
<td>An unknown message type was received in the RECVCODE state.</td>
</tr>
<tr>
<td>The SM CCB parameter is null in number match process.</td>
<td>The pointer to the SM CCB is null in the service match process.</td>
</tr>
<tr>
<td>Failed to create wait alerting timer of in leg in RECVCODE state.</td>
<td>The service match module failed to create a WAIT_ALERTING timer for an input leg in the RECVCODE state.</td>
</tr>
<tr>
<td>The number process result is error in RECVCODE state.</td>
<td>An error occurred during number processing in the RECVCODE state.</td>
</tr>
<tr>
<td>Failed to stop signal tone.</td>
<td>The service match module failed to stop announcement play.</td>
</tr>
<tr>
<td>Failed to create play sound interval timer.</td>
<td>The service match module failed to create an interval timer for playing announcements.</td>
</tr>
<tr>
<td>Could not accept '*' when the cancel affix number template is null.</td>
<td>The asterisk * was unacceptable when the cancel the additional code template item was null.</td>
</tr>
<tr>
<td>Receive '*' when both affix number template are null.</td>
<td>An asterisk * was received when two additional code templates were null.</td>
</tr>
<tr>
<td>The operate type is invalid when received join character.</td>
<td>The operation type was invalid when an asterisk was received.</td>
</tr>
<tr>
<td>Failed to save supplement service config information.</td>
<td>The service match module failed to save the configuration of a supplementary service.</td>
</tr>
<tr>
<td>The operate type is error when proceed the match result.</td>
<td>The operation type was incorrect in the processing of the match result.</td>
</tr>
<tr>
<td>Feature service is disable, can not send feature message.</td>
<td>The FEATURE service was disabled, and no FEATURE message could be sent.</td>
</tr>
<tr>
<td>Failed to get feature service switch.</td>
<td>The service match module failed to obtain a FEATURE service toggle.</td>
</tr>
<tr>
<td>The service type is error.</td>
<td>The service type was incorrect.</td>
</tr>
<tr>
<td>Failed to match any service.</td>
<td>No service could be matched.</td>
</tr>
<tr>
<td>There is a null pointer error in number match process.</td>
<td>There was a null pointer in the service match process.</td>
</tr>
<tr>
<td>The call information table index is invalid, construct feature message failed.</td>
<td>The call information table index was invalid, and constructing a FEATURE message failed.</td>
</tr>
<tr>
<td>Failed to get voice entity for call.</td>
<td>The service match module failed to obtain a voice entity for the call.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>The feature string is too long after substitute the called number.</td>
<td>The FEATURE string exceeded the limited length after called number substitution.</td>
</tr>
<tr>
<td>Could not find any voice entity.</td>
<td>No voice entity could be matched.</td>
</tr>
<tr>
<td>Stop DTMF detect error.</td>
<td>Disabling DTMF detection failed.</td>
</tr>
<tr>
<td>Play signal tone error when succeed to set local service.</td>
<td>The service match module failed to play announcements after a local service was set successfully.</td>
</tr>
<tr>
<td>Create wait dial interval timer failed when continue to receive number.</td>
<td>The service match module failed to create a timer for dialing the next digit.</td>
</tr>
<tr>
<td>Service register is failed and terminated.</td>
<td>The service registration failed and the registration flow ended.</td>
</tr>
<tr>
<td>An error is made when proceed service register.</td>
<td>An error occurred during service registration.</td>
</tr>
<tr>
<td>The received DTMF number is invalid.</td>
<td>The received DTMF code was invalid.</td>
</tr>
<tr>
<td>Feature service is permitted, will check the feature template record.</td>
<td>The FEATURE service was enabled, and the FEATURE template record would be checked.</td>
</tr>
<tr>
<td>Feature service is denied, do not check the feature template record.</td>
<td>The FEATURE service was disabled, and the FEATURE template record would not be checked.</td>
</tr>
<tr>
<td>Matched service code all, id = 4, name = call waiting.</td>
<td>The feature code was completely matched. The ID of the matched feature is 4, and the name is call waiting.</td>
</tr>
<tr>
<td>Matched service code partly, id = 4, name = call waiting.</td>
<td>The feature code was partially matched. The ID of the matched feature is 4, and the name is call waiting.</td>
</tr>
<tr>
<td>The current received service code is &quot;5&quot;.</td>
<td>The DTMF digit currently received was 5.</td>
</tr>
<tr>
<td>The affix number is null when matched an optional affix item.</td>
<td>A feature with an optional additional code was matched, but the additional code was null.</td>
</tr>
<tr>
<td>The affix number is exceed max password length.</td>
<td>The additional code exceeded the maximum password length.</td>
</tr>
<tr>
<td>The length of call number is exceed max length.</td>
<td>The call number exceeded the maximum length.</td>
</tr>
<tr>
<td>Have not received any digital affix number when proceed join DTMF number.</td>
<td>No additional code consisting of digits was received in the processing of the asterisk * .</td>
</tr>
<tr>
<td>The service state must be RECVMNUM when received &quot;*&quot; in service match process.</td>
<td>The service state must be RECVCODE when an asterisk * was being received.</td>
</tr>
<tr>
<td>You must input number before input &quot;*&quot; in RECVMNUM state.</td>
<td>A digital number must be received before an asterisk in the RECVCODE state.</td>
</tr>
<tr>
<td>Check the first affix number item when matched feature service.</td>
<td>The first additional code was checked when a FEATURE service was matched.</td>
</tr>
<tr>
<td>The &quot;<em>&quot; has existed in the number when receive &quot;</em>&quot;.</td>
<td>An asterisk already existed in the additional code when the asterisk was being received.</td>
</tr>
<tr>
<td>The received number is exceed max length and the &quot;*&quot; could not be the last number.</td>
<td>The received number exceeded the maximum length, and no asterisk could appear at the end of a feature code.</td>
</tr>
<tr>
<td>Failed to set feature service operate type.</td>
<td>The service match module failed to set an operation type for a feature service.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>The affix number template not allow to input '*'.</td>
<td>No asterisk was allowed in the additional code template.</td>
</tr>
<tr>
<td>Match failed, the affix number’s length must between 1 and 31.</td>
<td>The match failed. The additional code must be 1 to 31 characters in length.</td>
</tr>
<tr>
<td>Match failed, the affix number is not 4 digits or state is not RECVCODE.</td>
<td>The match failed. The additional code was not a four-digit code, and the state was not RECVCODE.</td>
</tr>
<tr>
<td>Match failed in RECVCODE state, there is no affix number in the matched service.</td>
<td>The match failed in the RECVCODE state, and the additional code of the matched service was null.</td>
</tr>
<tr>
<td>Invalid affix number template of local service when match is terminated.</td>
<td>The additional code template of the matched local service was invalid.</td>
</tr>
<tr>
<td>Match failed, needs an affix number item and length is between 1 and 31.</td>
<td>The match failed. The matched service required an additional code whose length must be within 1 to 31 characters.</td>
</tr>
<tr>
<td>Service state is error when received terminal number.</td>
<td>The service state was incorrect when a terminator was received.</td>
</tr>
<tr>
<td>The received number is null when received terminal number.</td>
<td>The received number was null when a terminator was received.</td>
</tr>
<tr>
<td>Match failed, the count of '*' in affix number must be one.</td>
<td>The match failed. The number of asterisks in the additional code must be 1.</td>
</tr>
<tr>
<td>Match failed, the position of '*' is not in the middle of affix number string.</td>
<td>The match failed. The asterisk must not be at the head or end of an additional code.</td>
</tr>
<tr>
<td>Have not match any service when match is terminated.</td>
<td>No service was matched.</td>
</tr>
<tr>
<td>The matched service is not local service when save configuration.</td>
<td>The matched service in the FT CCB was a non-local supplementary service when the configuration was saved.</td>
</tr>
<tr>
<td>Invalid interface index when save configuration.</td>
<td>The interface index was invalid when the configuration was being saved.</td>
</tr>
<tr>
<td>Can not get vim PCB when save configuration.</td>
<td>The service match module failed to obtain the VIM CCB when the configuration was being saved.</td>
</tr>
<tr>
<td>The vim PCB is null when save configuration.</td>
<td>The obtained VIM CCB was null when the configuration was being saved.</td>
</tr>
<tr>
<td>The dial out restriction password is error.</td>
<td>The password for outgoing call barring was incorrect.</td>
</tr>
<tr>
<td>The operate type is invalid when save configuration.</td>
<td>The operation type was invalid when the configuration was being saved.</td>
</tr>
<tr>
<td>The service type is invalid when save config information.</td>
<td>The service type was invalid when the configuration was being saved.</td>
</tr>
<tr>
<td>The config information pointer is null when save call-forwarding configuration.</td>
<td>The pointer to the configuration information was null when the call forwarding configuration was being saved.</td>
</tr>
<tr>
<td>The forward number parameter is null when save call-forwarding configuration.</td>
<td>The pointer to the forwarded-to number was null when the call forwarding configuration was being saved.</td>
</tr>
<tr>
<td>The CF service type parameter is error when save call-forwarding configuration.</td>
<td>The CF service type was incorrect when the call forwarding configuration was being saved.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>invalid operate type parameter when save call-forwarding configuration.</td>
<td>The operation type was invalid when the call forwarding configuration was being saved.</td>
</tr>
<tr>
<td>Can not find CMC CCB when create SM CCB.</td>
<td>The service match module failed to obtain the CMC CCB when an SM CCB was being created.</td>
</tr>
<tr>
<td>The SM service CCB pointer in CMC CCB is existed when create SM CCB.</td>
<td>The pointer to the SM CCB in the CMC CCB already existed when an SM CCB was being created.</td>
</tr>
<tr>
<td>The call state is error when create SM CCB.</td>
<td>The call state is incorrect during creation of an SM CCB.</td>
</tr>
<tr>
<td>Could not get feature service switch when create SM CCB.</td>
<td>Failed to get the FEATURE service toggle during creation of an SM CCB.</td>
</tr>
<tr>
<td>Failed to allocate memory for SM CCB.</td>
<td>The service match module failed to allocate memory to the SM CCB.</td>
</tr>
<tr>
<td>Get LGS dial interval timer length failed.</td>
<td>The service match module failed to obtain the LGS timer for dialing the next digit.</td>
</tr>
<tr>
<td>Can not find CMC CCB when delete SM CCB.</td>
<td>No CMC CCB could be found during deletion of the SM CCB.</td>
</tr>
<tr>
<td>The affix item index parameter is invalid when set operate type.</td>
<td>The additional item index was incorrect during setting of an operation type.</td>
</tr>
<tr>
<td>Could not set the operate type of local service.</td>
<td>Setting an operation type was not allowed for a local service.</td>
</tr>
<tr>
<td>The feature code parameter is null when parse feature code.</td>
<td>The feature code to be parsed was null.</td>
</tr>
<tr>
<td>The feature code string is exceed the max length when parse feature code.</td>
<td>The feature code to be parsed exceeded the maximum length.</td>
</tr>
<tr>
<td>The number template index point parameter is null when parse feature code.</td>
<td>The number template index was null when a feature code was being parsed.</td>
</tr>
<tr>
<td>The feature index point parameter is null when parse feature code.</td>
<td>The FEATURE index was null when a feature code was being parsed.</td>
</tr>
<tr>
<td>The first number is not start feature number when parse feature code.</td>
<td>The initial character was not one of a feature code when a feature code was being parsed.</td>
</tr>
<tr>
<td>There is an invalid character in the feature code string.</td>
<td>Illegal characters existed in a feature code.</td>
</tr>
<tr>
<td>The fifth character in the feature code string must be '*' or null.</td>
<td>The fifth character in a feature code must be * or null.</td>
</tr>
<tr>
<td>The affix number template item is invalid when parse feature code.</td>
<td>An item in the additional code template was invalid when a feature code was being parsed.</td>
</tr>
<tr>
<td>Could not match any feature service by feature code &quot;*410&quot;.</td>
<td>The feature code *410 could not match any FEATURE service.</td>
</tr>
<tr>
<td>The feature message type point parameter is null when get message send type.</td>
<td>The pointer to the FEATURE message type was null during getting of the message send type.</td>
</tr>
<tr>
<td>The feature index is invalid when get message send type.</td>
<td>The pointer to the feature index was null during getting of the message send type.</td>
</tr>
<tr>
<td>The source feature code parameter is null when reverse code.</td>
<td>The pointer to the source feature code was null during reversal of a feature code.</td>
</tr>
</tbody>
</table>
The destination feature code parameter is null when reverse code.
The pointer to the destination feature code was null during reversal of a feature code.

The feature index is invalid when reverse feature code.
The feature index obtained through parsing was invalid during reversal of a feature code.

Failed to reverse code because of invalid operate type.
Reversing a feature code failed because the operation type was invalid.

Can not find join character when reverse feature code.
No asterisk could be found during reversal of a feature code.

The cancel affix number template item is invalid when reverse feature code.
The cancel additional code template item was invalid during reversal of a feature code.

Failed to reverse code! this operation need more information.
Reversing a feature code failed because more information was required.

Matched feature service, but it could not be reversed.
A feature service was matched, but the feature code could be reversed.

The in call leg’s state is invalid in get call state.
The state of the input call leg was invalid during getting of the call state.

The out call leg’s state is invalid in get call state.
The state of the output call leg was invalid during getting of the call state.

The call leg’s leg type is invalid in get call state.
The call leg type was invalid during getting of the call state.

Table 152 describes output fields and messages for the debugging voice ss sm event command.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Receive CMC_SETUP message from SA module in invalid SM CCB state.</td>
<td>A CMC_SETUP message was received from the SA module in the invalid SM CCB state.</td>
</tr>
<tr>
<td>Receive CMC_INFORMATION message from SA module in RECVCODE state.</td>
<td>A CMC_INFORMATION message was received from the SA module in the RECVCODE state.</td>
</tr>
<tr>
<td>Send CMC_SERVICE(feature) message to SA module.</td>
<td>A CMC_SERVICE(feature) message was sent to the SA module.</td>
</tr>
</tbody>
</table>

Table 153 describes output fields and messages for the debugging voice ss sm fsm command.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>State changed from IDLE to RECVCODE.</td>
<td>The state changed from idle to RECVCODE.</td>
</tr>
</tbody>
</table>

Table 154 describes output fields and messages for the debugging voice ss sm info command.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stop playing result sound.</td>
<td>The service match module stopped announcing the result of service registration.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>The feature service is denied, transmit the ACCP_INFORMATION message</td>
<td>The FEATURE service was disabled, and an ACCP_INFORMATION message was transparently transmitted to the peer end.</td>
</tr>
<tr>
<td>to other side.</td>
<td></td>
</tr>
<tr>
<td>Feature service is permit, hold back the ACCP_INFORMATION message.</td>
<td>The FEATURE service was enabled, and an ACCP_INFORMATION message was intercepted.</td>
</tr>
<tr>
<td>Receive ACCP_INFORMATION message when SM CCB is invalid and not in</td>
<td>An ACCP_INFORMATION message was received when the SM CCB was invalid and no call was set up.</td>
</tr>
<tr>
<td>talking state.</td>
<td></td>
</tr>
<tr>
<td>The brother SPL is not SIP when in invalid SM CCB state.</td>
<td>The signaling protocol at the peer end was not SIP when the SM CCB was invalid.</td>
</tr>
<tr>
<td>Receive hookflash message and hand over to SA module in RECVCODE state.</td>
<td>A HOOKFLASH message was received in the RECVCODE state and then forwarded to the SA module for processing.</td>
</tr>
<tr>
<td>Receive hookflash message and keep it back in RECVCODE state.</td>
<td>A HOOKFLASH message was received and kept in the RECVCODE state.</td>
</tr>
<tr>
<td>Stop receiving DTMF number.</td>
<td>The service match module stopped receiving the DTMF code.</td>
</tr>
<tr>
<td>Clear received number string, Begin to receive affix number.</td>
<td>The received number string was cleared, and receiving an additional code started.</td>
</tr>
<tr>
<td>Succeed to save supplement service config information.</td>
<td>The supplementary service configuration was saved successfully.</td>
</tr>
<tr>
<td>The feature message send type is ACCP_SERVICE.</td>
<td>A FEATURE message was sent in an ACCP_SERVICE message.</td>
</tr>
<tr>
<td>The number is partially matched, continue to receive DTMF number.</td>
<td>The number was partially matched, and receiving the DTMF code continued.</td>
</tr>
<tr>
<td>Succeed to substitute the call number for feature service.</td>
<td>Number substitution for the FEATURE service succeeded.</td>
</tr>
<tr>
<td>Succeed to match local supplement service, service name = call waiting.</td>
<td>A local supplementary service was successfully matched, and the service name was call waiting.</td>
</tr>
<tr>
<td>Begin to play result sound.</td>
<td>The service match module stopped started announcing the registration result.</td>
</tr>
<tr>
<td>Succeed to match feature service, service name = Do not Disturb Toggle.</td>
<td>A FEATURE service was successfully matched, and the service name was do-no-disturb toggle.</td>
</tr>
<tr>
<td>The feature code in the service message is “*446”.</td>
<td>The feature code in a CMC_SERVICE message was *446.</td>
</tr>
<tr>
<td>Matched partly when receive digital, continue to receive affix number.</td>
<td>The feature code was partially matched after a digital number was received, and then the additional code continued to be received.</td>
</tr>
<tr>
<td>Succeed to match service when the template defines an affix number item.</td>
<td>A service with an additional code was successfully matched.</td>
</tr>
<tr>
<td>Succeed to match service when the affix number is 4 digits password.</td>
<td>A service with a four-digit additional code was successfully matched.</td>
</tr>
<tr>
<td>Succeed to match service, no affix number and the service state is</td>
<td>A service without any additional code was successfully matched in the RECVCODE state.</td>
</tr>
<tr>
<td>RECVCODE.</td>
<td></td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td>The matched service is execute type, so only check the first affix number item.</td>
<td>An execution service was matched, and only the first additional code in the template was checked.</td>
</tr>
<tr>
<td>Check the first affix number item when matched feature service.</td>
<td>The first additional code was checked when a FEATURE service was matched.</td>
</tr>
<tr>
<td>Succeed to match service when the template defines an affix number item.</td>
<td>A service with an additional code was successfully matched.</td>
</tr>
<tr>
<td>Match Successfully, the template defines two compulsory affix number item.</td>
<td>A service with two additional codes was successfully matched.</td>
</tr>
<tr>
<td>Match successfully in RECVCODE state, when matched an optional affix item.</td>
<td>A service with optional additional codes was successfully matched in the RECVCODE state.</td>
</tr>
<tr>
<td>Match successfully in RECVNUM state, when matched an optional affix item.</td>
<td>A service with optional additional codes was successfully matched in the RECVNUM state.</td>
</tr>
<tr>
<td>Match successfully, when the affix number is null and the service state is RECVCODE.</td>
<td>A service without any additional code was successfully matched in the RECVCODE state.</td>
</tr>
<tr>
<td>Enable call waiting service.</td>
<td>The CW service was enabled.</td>
</tr>
<tr>
<td>Disable call waiting service.</td>
<td>The CW service was disabled.</td>
</tr>
<tr>
<td>Enable call-forwarding unconditional service.</td>
<td>The call forwarding unconditional (CFU) was enabled.</td>
</tr>
<tr>
<td>Enable call-forwarding on busy service.</td>
<td>The call forwarding on busy (CFB) service was enabled.</td>
</tr>
<tr>
<td>Enable call-forwarding no reply service.</td>
<td>The call forwarding no reply (CFNR) was service enabled.</td>
</tr>
<tr>
<td>Enable call-forwarding not available service.</td>
<td>The call forwarding not available service was enabled.</td>
</tr>
<tr>
<td>Disable call-forwarding unconditional service.</td>
<td>The call forwarding unconditional (CFU) service was disabled.</td>
</tr>
<tr>
<td>Disable call-forwarding on busy service.</td>
<td>The call forwarding on busy (CFB) service was disabled.</td>
</tr>
<tr>
<td>Disable call-forwarding no reply service.</td>
<td>The call forwarding no reply (CFNR) service was disabled.</td>
</tr>
<tr>
<td>Disable call-forwarding not available service.</td>
<td>The call forwarding not available service was disabled.</td>
</tr>
<tr>
<td>Create a SM CCB.</td>
<td>An SM CCB was created.</td>
</tr>
<tr>
<td>Delete the SM CCB.</td>
<td>An SM CCB was created.</td>
</tr>
<tr>
<td>Send the feature code by SETUP message.</td>
<td>A feature code was sent in a SETUP message.</td>
</tr>
<tr>
<td>Succeed to reverse code, the reverse Feature code is &quot;*465&quot;.</td>
<td>The feature code was successfully reversed, and the reversed feature code was *465.</td>
</tr>
</tbody>
</table>

Table 155 describes output fields and messages for the **debugging voice ss sm timer** command.
### Table 155 Output from the debugging voice ss sm timer command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Timer created, TimerId: TIMER_SS_SM_DIALINTERVAL, Timer length: 11000.</td>
<td>An 11-second timer for dialing the next digit was created.</td>
</tr>
<tr>
<td>Timer deleted, TimerId: TIMER_INLEG_WAIT_ALERTING.</td>
<td>The WAIT_ALERTING timer for the input leg was deleted.</td>
</tr>
<tr>
<td>Timer timeout, TimerId: TIMER_SS_SM playsound interval.</td>
<td>The interval timer for playing announcements expired.</td>
</tr>
</tbody>
</table>

### Examples

Output similar to the following example is generated when the user at Telephone A attached to Router A picks up the headset and dials *58# to enable call waiting on the voice subscriber line:

```
# Enable debugging for service module events on Router A.
<RouterA> debugging voice ss sm event
SS SM event debugging switch is on
<RouterA>
*May 14 11:41:32:601 2007 RouterA SS/7/VOICE:
  SM_EVT: [0x00000000] Receive CMC_SETUP message from SA module in invalid SM CCB state.
  // A CMC_SETUP message was received from the SA module in the invalid SM CCB state.

*May 14 11:41:32:991 2007 RouterA SS/7/VOICE:
  SM_EVT: [0x00000000] Receive CMC_INFORMATION message from SA module in RECVCODE state.
  // A CMC_INFORMATION message was received from the SA module in the RECVCODE state. The following repeated this process to collect the DTMF code.

*May 14 11:41:33:561 2007 RouterA SS/7/VOICE:
  SM_EVT: [0x00000000] Receive CMC_INFORMATION message from SA module in RECVCODE state.
*May 14 11:41:34:281 2007 RouterA SS/7/VOICE:
  SM_EVT: [0x00000000] Receive CMC_INFORMATION message from SA module in RECVCODE state.
  // The service was set successfully, and the setting result was announced. A timer was used to control announcement play, and announcements were played five times.

*May 14 11:41:34:481 2007 RouterA SS/7/VOICE:
  SM_EVT: [0x00000000] Receive TIMER_SS_SM playsound interval message from SA module in RECVCODE state.
  // The interval timer for playing announcement expired in the RECVCODE state.

*May 14 11:41:34:681 2007 RouterA SS/7/VOICE:
  SM_EVT: [0x00000000] Receive TIMER_SS_SM playsound interval message from SA module in RECVCODE state.
*May 14 11:41:34:881 2007 RouterA SS/7/VOICE:
  SM_EVT: [0x00000000] Receive TIMER_SS_SM playsound interval message from SA module in RECVCODE state.
*May 14 11:41:35:81 2007 RouterA SS/7/VOICE:
  SM_EVT: [0x00000000] Receive TIMER_SS_SM playsound interval message from SA module in RECVCODE state.
*May 14 11:41:35:281 2007 RouterA SS/7/VOICE:
```

288
SM_EVT: [0x00000000] Receive TIMER_SS_SM_PLAY_SOUND_INTERVAL message from SA module in
RECV_CODE state.

# Enable debugging for service match module finite state machines on Router A.
<RouterA>
<RouterA> debugging voice ss_sm_fsm
SS SM FSM debugging switch is on
<RouterA>
*May 14 11:50:46:497 2007 RouterA SS/7/VOICE:
  SM_FSM: [0x00010000] State changed from IDLE to RECV_CODE.

// The state changed from idle to RECV_CODE.

# Enable debugging for service match module information on Router A.
<RouterA> debugging voice ss_sm_info
SS SM information debugging switch is on
<RouterA>
*May 14 14:00:32:10 2007 RouterA SS/7/VOICE:
  SM_INFO: [0x00020000] Create a SM CCB.

// An SM CCB was created.

*May 14 14:00:32:110 2007 RouterA SS/7/VOICE:
  SM_INFO: [0x00030000] Digit matching was in progress in the service registration state.

// Digit matching was in progress in the service registration state.

*May 14 14:00:32:110 2007 RouterA SS/7/VOICE:
  SM_INFO: [0x00030000] Feature service is permitted, will check the feature template record.

// The FEATURE service was enabled, and the FEATURE service template would be checked.

*May 14 14:00:32:110 2007 RouterA SS/7/VOICE:
  SM_INFO: [0x00030000] Matched service code partly, id = 0, name = call forwarding universal.
*May 14 14:00:32:111 2007 RouterA SS/7/VOICE:
  SM_INFO: [0x00030000] Matched service code partly, id = 4, name = call waiting.
*May 14 14:00:32:111 2007 RouterA SS/7/VOICE:
  SM_INFO: [0x00030000] Matched service code partly, id = 5, name = call restriction in.
*May 14 14:00:32:111 2007 RouterA SS/7/VOICE:
  SM_INFO: [0x00030000] Matched service code partly, id = 6, name = call restriction out.

// Service matching was in progress.

*May 14 14:00:32:112 2007 RouterA SS/7/VOICE:
  SM_INFO: [0x00030000] The current received service code is "5".

// The DTMF digit currently received was 5.

*May 14 14:00:32:112 2007 RouterA SS/7/VOICE:
  SM_INFO: [0x00030000] The number is partially matched, continue to receive DTMF number.

// The DTMF digit was matched, and the next DTMF digit was to be collected and matched against the
template. The process above was repeated.
May 14 14:00:32:350 2007 RouterA SS/7/VOICE:
SM_INFO: [0x00030000] Match number in service register mode.

May 14 14:00:32:350 2007 RouterA SS/7/VOICE:
SM_INFO: [0x00030000] Feature service is permitted, will check the feature template record.

May 14 14:00:32:393 2007 RouterA SS/7/VOICE:
SM_INFO: [0x00030000] Matched service code all, id = 4, name = call waiting.

May 14 14:00:32:544 2007 RouterA SS/7/VOICE:
SM_INFO: [0x00030000] The number is partially matched, continue to receive DTMF number.

May 14 14:00:32:694 2007 RouterA SS/7/VOICE:
SM_INFO: [0x00030000] Match number in service register mode.

May 14 14:00:32:844 2007 RouterA SS/7/VOICE:
SM_INFO: [0x00030000] Succeed to match service, no affix number and the service state is RECVCODE.

// The feature code was matched. There was no additional code, and the service state was RECVCODE.

May 14 14:00:32:994 2007 RouterA SS/7/VOICE:
SM_INFO: [0x00030000] Enable call waiting service.

// The CW service was enabled.

May 14 14:00:33:94 2007 RouterA SS/7/VOICE:
SM_INFO: [0x00030000] Succeed to save supplement service config information.

// The supplementary service configuration was saved successfully.

May 14 14:00:33:244 2007 RouterA SS/7/VOICE:
SM_INFO: [0x00030000] Succeed to match local supplement service, service name = call waiting.

// The local supplementary service (CW) was matched successfully.

May 14 14:00:33:344 2007 RouterA SS/7/VOICE:
SM_INFO: [0x00030000] Close DTMF successfully, when received terminal number in local-priority mode.

// When a terminator was received in local-priority mode, DTMF detection was disabled.

May 14 14:00:33:504 2007 RouterA SS/7/VOICE:
SM_INFO: [0x00030000] Begin to play result sound.

// The result was announced.

May 14 14:00:33:654 2007 RouterA SS/7/VOICE:
SM_INFO: [0x00030000] Stop playing result sound.

// Result announcement stopped.

May 14 14:00:33:804 2007 RouterA SS/7/VOICE:
SM_INFO: [0x00030000] Delete the SM CCB.

// The SM CCB was deleted.
# Enable debugging for service match module timers on Router A.

<RouterA> debugging voice ss sm timer
SS SM timer debugging switch is on<AR19>
<RouterA>

*May 14 14:50:44:589 2007 RouterA SS/7/VOICE:
    SM_TMR: [0x00060000] Timer created, Timer ID: TIMER_SS_SM_DIALINTERVAL, Timer length: 11000.
    // A 11000-ms interval timer for dialing the next digit was created.

*May 14 14:50:46:658 2007 RouterA SS/7/VOICE:
    SM_TMR: [0x00060000] Timer deleted, Timer ID: TIMER_INLEG_WAIT_ALERTING.
    // The WAIT_ALERTING timer for the input leg was deleted.

*May 14 14:50:46:658 2007 RouterA SS/7/VOICE:
    SM_TMR: [0x00060000] Timer deleted, Timer ID: TIMER_SS_SM_DIALINTERVAL.
    // The interval timer for dialing the next digit was deleted.

    // The following repeated the timer creation and deletion process until a service was matched successfully.

*May 14 14:50:46:659 2007 RouterA SS/7/VOICE:
    SM_TMR: [0x00060000] Timer created, Timer ID: TIMER_SS_SM_DIALINTERVAL, Timer length: 11000.
*May 14 14:50:47:168 2007 RouterA SS/7/VOICE:
    SM_TMR: [0x00060000] Timer deleted, Timer ID: TIMER_INLEG_WAIT_ALERTING.
*May 14 14:50:47:168 2007 RouterA SS/7/VOICE:
    SM_TMR: [0x00060000] Timer deleted, Timer ID: TIMER_SS_SM_DIALINTERVAL.
*May 14 14:50:47:169 2007 RouterA SS/7/VOICE:
    SM_TMR: [0x00060000] Timer created, Timer ID: TIMER_SS_SM_DIALINTERVAL, Timer length: 11000.
*May 14 14:50:48:519 2007 RouterA9 SS/7/VOICE:
    SM_TMR: [0x00060000] Timer created, Timer ID: TIMER_SS_SM_DIALINTERVAL, Timer length: 11000.
*May 14 14:50:48:519 2007 RouterA SS/7/VOICE:
    SM_TMR: [0x00060000] Timer created, Timer ID: TIMER_INLEG_WAIT_ALERTING.
*May 14 14:50:48:519 2007 RouterA SS/7/VOICE:
    SM_TMR: [0x00060000] Timer deleted, Timer ID: TIMER_SS_SM_DIALINTERVAL.

*May 14 14:50:48:520 2007 RouterA SS/7/VOICE:
    SM_TMR: [0x00060000] Timer created, Timer ID: TIMER_SS_SM_PLAY_SOUND_INTERVAL, Timer length: 200.
    // A 200-ms interval timer for playing announcements was created.

*May 14 14:50:48:719 2007 RouterA SS/7/VOICE:
    SM_TMR: [0x00060000] Timer timeout, Timer ID: TIMER_SS_SM_PLAY_SOUND_INTERVAL.
    // The interval timer for playing announcements expired.

    // The following repeatedly announced the successful setting of a feature service four more times by repeatedly creating and deleting such a timer four times.

*May 14 14:50:48:719 2007 RouterA SS/7/VOICE:
    SM_TMR: [0x00060000] Timer created, Timer ID: TIMER_SS_SM_PLAY_SOUND_INTERVAL, Timer length: 200.
*May 14 14:50:48:918 2007 RouterA SS/7/VOICE:
debugging voice ss rc

Use **debugging voice ss rc** to enable release control debugging.

Use **undo debugging voice ss rc** to disable release control debugging.

**Syntax**

```
debugging voice ss rc { all | error | event | info | timer }
undo debugging voice ss rc { all | error | event | info | timer }
```

**Default**

Release control debugging is disabled.

**Views**

User view

**Default command level**

2: System level

**Parameters**

- **all**: Specifies all types of debugging for release control.
- **error**: Specifies error debugging.
- **event**: Specifies event debugging.
- **info**: Specifies information debugging.
- **timer**: Specifies timer debugging.

**Usage guidelines**

Table 156 describes output fields and messages for the **debugging voice ss rc error** command.
Table 156 Output from the debugging voice ss rc error command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Allocate memory failed for creating</td>
<td>The release control module failed to allocate memory for creating release</td>
</tr>
<tr>
<td>Release-control CB.</td>
<td>control block (RCB).</td>
</tr>
<tr>
<td>Receive invalid message of %s.</td>
<td>An invalid message of %s was received.</td>
</tr>
</tbody>
</table>

Table 157 describes output fields and messages for the **debugging voice ss sm event** command.

Table 157 Output from the debugging voice ss rc event command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Process the message of %s in %s state.</td>
<td>The % message was processed in %s state.</td>
</tr>
</tbody>
</table>

Table 158 describes output fields and messages for the **debugging voice ss sm info** command.

Table 158 Output from the debugging voice ss rc info command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>The Release-control CB is initialized</td>
<td>The initialization of RCB was successful.</td>
</tr>
<tr>
<td>successfully.</td>
<td></td>
</tr>
<tr>
<td>The Release-control CB is deleted</td>
<td>The RCB was successfully deleted.</td>
</tr>
<tr>
<td>successfully.</td>
<td></td>
</tr>
<tr>
<td>Release control receive %s message.</td>
<td>Message %s was received.</td>
</tr>
<tr>
<td>Can’t start release control for invalid</td>
<td>The release control module failed to start the</td>
</tr>
<tr>
<td>message or Call-leg type.</td>
<td>release control services because of invalid</td>
</tr>
<tr>
<td></td>
<td>message or wrong call-leg type.</td>
</tr>
<tr>
<td>Receive invalid message in %s state.</td>
<td>The invalid message was received in %s state.</td>
</tr>
</tbody>
</table>

Table 159 describes output fields and messages for the **debugging voice ss sm timer** command.

Table 159 Output from the debugging voice ss rc timer command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Timer created, TimerId : %s, Timer length</td>
<td>Timer %s with the duration of %d was created.</td>
</tr>
<tr>
<td>%d.</td>
<td></td>
</tr>
</tbody>
</table>

Examples

# Enable all types of debugging for release control on Router A. Output similar to the following example is generated when Telephone A hangs up after Telephone B successfully places a call with Telephone A under these conditions:

- Route A and Router B are reachable to each other. Telephone A and Telephone B are attached to Route A and Router B, respectively.
- The on-hook delay time of the called party (Telephone A) has an on-hook delay time of 90 seconds and is enabled with the calling party control service.

```
<RouteA> debugging voice ss rc all
<RouterA>
*Jul 7 19:17:37:231 2008 RouteA SS/7/VOICE:
   RC_INFO: [0x00090001] Release control receive CMC_RELEASE message.
```
A RELEASE message was received.

*Jul 7 19:17:37:232 2008 RouteA SS/7/VOICE:
   RC_INFO: [0x00090001] The Release control CB is initialized successfully.

The initialization of RCB is successful.

*Jul 7 19:17:37:232 2008 RouteA SS/7/VOICE:
   RC_EVT: [0x00090001] Process the message of CMC_RELEASE in S_RC_IDLE state.

The RELEASE message was received in idle state.

*Jul 7 19:17:37:232 2008 RouteA SS/7/VOICE:
   RC_TMR: [0x000000ca] Timer created, Timer ID: RC_DELAY_ONHOOK, Timer length: 90000.

The on-hook delay timer with the duration of 90 seconds was created.

*Jul 7 19:17:37:232 2008 RouteA SS/7/VOICE:
   RC_TMR: [0x00000145] Timer created, Timer ID: RC_DELAY_SENDSETUP, Timer length: 50.

The call delay timer was created.

*Jul 7 19:17:37:272 2008 RouteA SS/7/VOICE:
   RC_INFO: [0x00090001] Release control receive RC_DELAY_SENDSETUP message.

The call delay timer timed out.

*Jul 7 19:17:37:272 2008 RouteA SS/7/VOICE:
   RC_EVT: [0x00090001] Process the message of RC_DELAY_SENDSETUP in S_RC_DELAY_START state.

The call delay timer timed out in start-delay state.

*Jul 7 19:17:37:273 2008 RouteA SS/7/VOICE:
   RC_TMR: [0x000090001] Timer deleted, Timer ID: RC_DELAY_SEND_SETUP.

The call delay timer was deleted.

*Jul 7 19:17:37:321 2008 RouteA SS/7/VOICE:
   RC_INFO: [0x00090001] Release control receive CMC_ALERTING message.

An ALERTING message was received.

*Jul 7 19:17:37:321 2008 RouteA SS/7/VOICE:
   RC_EVT: [0x00090001] Process the message of CMC_ALERTING in S_RC_WAIT_OFFHOOK state.

The call delay timer timed out in wait-for-off-hook state.

Telephone A goes off-hook now.

<RouteA>
*Jul 7 19:17:39:242 2008 RouteA SS/7/VOICE:
   RC_INFO: [0x00090001] Release control receive CMC_CONNECT message.
// A CONNECT message was received.

*Jul  7 19:17:39:242 2008 RouteA SS/7/VOICE:
   RC_EVT: [0x00090001] Process the message of CMC_CONNECT in S_RC_WAIT_OFFHOOK state.

// A CONNECT message was received in wait-for-off-hook state.

*Jul  7 19:17:39:242 2008 RouteA SS/7/VOICE:
   RC_TMR: [0x00090001] Timer deleted, Timer ID: RC_DELAY_ONHOOK.

// The on-hook delay timer was deleted.

*Jul  7 19:17:39:242 2008 RouteA SS/7/VOICE:
   RC_INFO: [0x00090000] The Release-control CB is deleted successfully.

// The RCB was successfully deleted.

Telephone A goes off-hook again.

<RouteA>

*Jul  7 19:17:44:551 2008 RouteA SS/7/VOICE:
   RC_INFO: [0x00090001] Release control receive CMC_RELEASE message.

// A RELEASE message was received.

*Jul  7 19:17:44:551 2008 RouteA SS/7/VOICE:
   RC_INFO: [0x00090001] Can't start release control for invalid message or Call-leg type.

// The calling party control function no longer took effect.

debugging voice ss join

Use debugging voice ss join to enable Join header services debugging.
Use undo debugging voice ss join to disable Join header services debugging.

Syntax

debugging voice ss join { all | error | event | fsm | info | timer }
undo debugging voice ss join { all | error | event | fsm | info | timer }

Default

Join header services debugging is disabled.

Views

User view

Default command level

2: System level

Parameters

all: Specifies all types of debugging for Join header services.
error: Specifies error debugging.
event: Specifies event debugging.
**fsm**: Specifies finite state machine debugging.

**info**: Specifies information debugging.

**timer**: Specifies timer debugging.

### Examples

# Enable all types of debugging for Join header services on Router A. Output similar to the following example is generated when Telephone C dials \*425*1000# to monitor Telephone A under these conditions:

- Telephone A at 1000 is attached to Router A, on which all types of debugging for Join header related services are enabled.
- Telephone B is attached to Router B.
- Telephone C is attached to Router C, on which all types of for Feature services are enabled.

```
<RouterA> debugging voice ss join all
*Jun 24 20:34:06:544 2009 MSR30-12 SS/7/VOICE:
  JOIN_INFO: [0x00020000] JOIN CCB is created successfully.
  // Join CCB was successfully created.

*Jun 24 20:34:06:544 2009 MSR30-12 SS/7/VOICE:
  JOIN_EVT: [0x00020000] Receive CMC_SETUP message at JOIN_IDLE state.

*Jun 24 20:34:06:544 2009 MSR30-12 SS/7/VOICE:
  JOIN_EVT: [0x00020000] Send CMC_CONNECT message to CMC module.
  // A CONNECT message was replied.

*Jun 24 20:34:06:555 2009 MSR30-12 SS/7/VOICE:
  JOIN_EVT: [0x00020000] Receive CMC_CHANNEL_READY message at JOIN_IDLE state.
  // The timer for CHANNEL_READY message was created to obtain the negotiated media information.

*Jun 24 20:34:06:555 2009 MSR30-12 SS/7/VOICE:
  JOIN_EVT: [0x00020000] Receive CMC_CHANNEL_READY message at JOIN_IDLE state.
  // The CHANNEL_READY message was received.

*Jun 24 20:34:06:555 2009 MSR30-12 SS/7/VOICE:
  JOIN_EVT: [0x00020000] Receive CMC_CHANNEL_READY message at JOIN_IDLE state.
  // The timer for DRVACK message was created.

*Jun 24 20:34:06:566 2009 MSR30-12 SS/7/VOICE:
  JOIN_EVT: [0x00000001] Receive CMC_VIM message at JOIN_WDRV state.
  // The state of the Join service changed from IDLE to WDRV.
```
*Jun 24 20:34:06:566 2009 MSR30-12 SS/7/VOICE:*
  
  JOIN_TMR: [0x00000001] Timer deleted, Timer ID: TIMER_SS_JOIN_WAIT_DRVACK.

*Jun 24 20:34:06:667 2009 MSR30-12 SS/7/VOICE:*
  
  JOIN_INFO: [0x00000001] Succeed to initialize DSP in WDRV state.
  
  // DSP was initialized successfully.

*Jun 24 20:34:06:817 2009 MSR30-12 SS/7/VOICE:*
  
  JOIN_FSM: [0x00000001] Join state moves from JOIN_WDRV to JOIN_ACTIVE.
  
  // The state of the Join service changed from WDRV to ACTIVE.
Call-watch debugging commands

The output description tables in this document only contain fields and messages that require an explanation.

debugging call-watch event

Use **debugging call-watch event** to enable call-watch event debugging.

Use **undo debugging call-watch event** to disable call-watch event debugging.

**Syntax**

```
debugging call-watch event
undo debugging call-watch event
```

**Default**

Call-watch event debugging is disabled.

**Views**

User view

**Default command level**

1: Monitor level

**Usage guidelines**

Table 160 describes output fields and messages for the **debugging call-watch event** command.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Call-Watch group <code>group-number</code> contains the interface <code>interface-type interface-number</code></td>
<td>Call-Watch group specified by the <code>group-number</code> argument contained the monitoring rule that monitored a local interface. The local interface is specified by the <code>interface-type interface-number</code> argument.</td>
</tr>
<tr>
<td><code>interface-type interface-number</code> link-protocol is <code>state</code></td>
<td>The link-layer state of the interface was specified by the <code>state</code> argument, which can be up or down.</td>
</tr>
<tr>
<td>Call-Watch group <code>group-number</code> from state 1 to state 2</td>
<td>The state of the monitor group specified by the <code>group-number</code> argument changed from state 1 to state 2.</td>
</tr>
<tr>
<td>controller ×× state change to Watch-Out</td>
<td>The state of the E1 or T1 interface changed to Watch-Out.</td>
</tr>
<tr>
<td>controller ×× associate group <code>group-number</code></td>
<td>The E1/T1 interface was associated with the monitor group specified by <code>group-number</code>.</td>
</tr>
<tr>
<td>controller ×× state from No-Watch-Out to Watch-Out</td>
<td>The state of the controller interface changed from No-Watch-Out to Watch-Out.</td>
</tr>
</tbody>
</table>
Examples

# Enable call-watch event debugging, and enable terminal monitoring. Output similar to the following example is generated when VLAN interface 111 is shut down under the following conditions exist:

- Interface T1 2/0 is associated with call-watch group 1.
- Call-watch group 1 monitors VLAN interface 111.

```bash
<Sysname> debugging call-watch event
<Sysname> terminal debugging
<Sysname> terminal monitor
```

- Output similar to the following messages is generated when you shut down VLAN interface 111:

  ```bash
  <Sysname> system-view
  [Sysname] interface vlan-interface 111
  [Sysname-Vlan-interface111] shutdown
  [Sysname-Vlan-interface111]
  %May 28 16:49:54:933 2008 Sysname IFNET/4/LINK UPDOWN:
  Vlan-interface111: link status is DOWN
  %May 28 16:49:54:933 2008 Sysname IFNET/4/UPDOWN:
  Line protocol on the interface Vlan-interface111 is DOWN
  %May 28 16:49:54:933 2008 Sysname IFNET/4/LINK UPDOWN:
  T1 2/0: link status is DOWN
  // The states of VLAN interface 111 and interface T1 2/0 changed to down.
  
  *0.34919499 Sysname CW/7/debug2:
  Call-Watch debug information: Call-Watch group 1 contains the interface Vlan-interface111
  Call-Watch debug information: Vlan-interface111 link-protocol is DOWN
  *0.34919499 Sysname CW/7/debug2:
  Call-Watch debug information: Call-Watch group 1 from UP to DOWN
  *0.34919499 Sysname CW/7/debug2:
  Call-Watch debug information: controller T1 2/0 state change to WATCH-OUT
  *0.34919499 Sysname CW/7/debug2:
  Call-Watch debug information: controller T1 2/0 associate group 1
  Call-Watch debug information: controller T1 2/0 state from No-Watch-Out to Watch-Out
  // The state of call-watch group 1 changed from up to down, and the state of interface T1 2/0 changed from No-Watch-Out to Watch-Out.

- Output similar to the following messages is generated when you bring up VLAN interface 111:

  ```bash
  [Sysname-Vlan-interface111] undo shutdown
  [Sysname-Vlan-interface111]
  %May 28 16:50:07:987 2008 Sysname IFNET/4/LINK UPDOWN:
  Vlan-interface111: link status is UP
  %May 28 16:50:07:987 2008 Sysname IFNET/4/UPDOWN:
  Line protocol on the interface Vlan-interface111 is UP
  // The state of VLAN interface 111 changed to up.
  
  *0.34932553 Sysname CW/7/debug2:
  Call-Watch debug information: Call-Watch group 1 contains the interface Vlan-interface111
  Call-Watch debug information: Vlan-interface111 link-protocol is UP
  *0.34932553 Sysname CW/7/debug2:
  Call-Watch debug information: Call-Watch group 1 from DOWN to UP
  ```
Call-Watch debug information: controller T1 2/0 associate group 1
Call-Watch debug information: controller T1 2/0 state from Watch-Out to No-Watch-Out

// The state of call-watch group 1 changed from up to down, and the state of interface T1 2/0 changed from No-Watch-Out to Watch-Out.

[Sysname-Vlan-interface111]
%May 28 16:50:14:379 2008 Sysname IFNET/4/LINK UPDOWN:
  T1 2/0: link status is UP

// The state of interface T1 1/0 changed to up.
CFD debugging commands

The output description tables in this document only contain fields and messages that require an explanation.

debugging cfd

Use **debugging cfd** to enable CFD debugging.

Use **undo debugging cfd** to disable CFD debugging.

**Syntax**

debugging cfd { error | fsm ais | { ais-track link-status packet | all | fsm | cci | fng | lbi | mcc | mme | rmep } | packet { receive | send } } [ interface interface-type interface-number ]

undo debugging cfd { error | fsm ais | { ais-track link-status packet | all | fsm | cci | fng | lbi | mcc | mme | rmep } | packet { receive | send } } [ interface interface-type interface-number ]

**Default**

CFD debugging is disabled.

**Views**

User view

**Default command level**

2: System level

**Parameters**

- **ais**: Specifies debugging for AIS.
- **ais-track link-status packet**: Specifies debugging for EAIS.
- **all**: Specifies all types of debugging for CFD.
- **cci**: Specifies debugging for the continuity check state machine.
- **error**: Specifies debugging for CFD errors.
- **fng**: Specifies debugging for the error alarm state machine.
- **fsm**: Specifies debugging for CFD state machines.
- **interface interface-type interface-number**: Specifies a port by its type and number.
- **lbi**: Specifies debugging for the loopback state machine.
- **mcc**: Specifies debugging for the cross connection CCM state machine.
- **mme**: Specifies debugging for the error CCM state machine.
- **packet**: Specifies debugging for the CFD packets.
- **receive**: Specifies debugging for received CFD packets.
- **rmep**: Specifies debugging for the remote MEP state machine.
- **send**: Specifies debugging for sent CFD packets.
Usage guidelines

If neither the send keyword nor the receive keyword are specified, debugging for both received and sent CFD packets is enabled.

If none of the ais, cci, fng, lbi, mcc, mme, and rmep keywords are specified, debugging for all CFD state machines is enabled.

If you do not specify a port, debugging for all ports is enabled.

If you enable debugging for sent packets on a port with inbound MEPs configured, only CCM packets are printed, while LT and LB packets are not printed. If you enable debugging for received packets on the port, none of CCM packets, LT packets, and LB packets are printed. To view all the packets, enable CFD debugging for both sent packets and received packets on all the ports.

If you enable CFD packet debugging on a device that supports sending short-interval CCM messages, the system outputs debugging information only when no port is specified. If the interval field value in the CCM messages is smaller than 4, the system samples one incoming packet and one outgoing packet per second for debugging.

Table 161 describes output fields and messages for the debugging cfd error command.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CCM</td>
<td>Continuity check message.</td>
</tr>
<tr>
<td>LBM</td>
<td>Loopback message.</td>
</tr>
<tr>
<td>LBR</td>
<td>Loopback reply messages.</td>
</tr>
<tr>
<td>LTM</td>
<td>Linktrace message.</td>
</tr>
<tr>
<td>LTR</td>
<td>Linktrace reply message.</td>
</tr>
<tr>
<td>TDM</td>
<td>One-way delay measurement.</td>
</tr>
<tr>
<td>DMM</td>
<td>Delay measurement message.</td>
</tr>
<tr>
<td>DMR</td>
<td>Delay measurement reply.</td>
</tr>
<tr>
<td>AIS</td>
<td>Alarm indication signal.</td>
</tr>
<tr>
<td>TST</td>
<td>Test PDU.</td>
</tr>
<tr>
<td>LMM</td>
<td>Loss measurement message.</td>
</tr>
<tr>
<td>LMR</td>
<td>Loss measurement reply.</td>
</tr>
<tr>
<td>PDU</td>
<td>Protocol data unit.</td>
</tr>
<tr>
<td>Origin Information TLV’s Origin MAC is multicast address</td>
<td>The source MAC address of the source information TLV is a multicast address.</td>
</tr>
<tr>
<td>Origin Information TLV’s Target MAC is multicast address</td>
<td>The destination MAC address of the source information TLV is a multicast address.</td>
</tr>
<tr>
<td>Origin Information TLV’s TTL is 0</td>
<td>The TTL of the source information TLV is 0, which is invalid.</td>
</tr>
<tr>
<td>Reply Information TLV’s return TTL is 255</td>
<td>The TTL of the reply information TLV is 255, which is invalid.</td>
</tr>
<tr>
<td>Reply Information TLV’s Relay is invalid</td>
<td>The relay value of the reply information TLV is invalid.</td>
</tr>
<tr>
<td>Reply Ingress TLV’s Ingress Action is invalid</td>
<td>The ingress action of the reply ingress TLV is invalid.</td>
</tr>
</tbody>
</table>
Field | Description
--- | ---
Reply Ingress TLV’s Ingress MAC is multicast address | The destination MAC address in the reply ingress TLV is a multicast address.
Reply Egress TLV’s Egress Action is invalid | The egress action of the reply egress TLV is invalid.
Reply Egress TLV’s Egress MAC is multicast address | The destination MAC address in the reply egress TLV is a multicast address.
Sequence number is not accordant | The sequence number does not match.
MEP which is found is disabled | The MEP used to process CMM packets is not enabled.
Failed to send LTM | Failed to invoke underlying interface to send LTM packets.
Chip does not support MAC entry | The chip does not support MAC entry searching.

Table 162 describes output fields and messages for the **debugging cfd ais-track link-status packet** command.

Table 162 Output from the debugging cfd ais-track link-status packet command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Send EAIS Packet via Port <strong>interface-type</strong> interface-name</td>
<td>An EAIS packet was sent out of port <strong>interface-type</strong> interface-name.</td>
</tr>
<tr>
<td>Pkt length</td>
<td>Packet length.</td>
</tr>
</tbody>
</table>

Table 163 describes output fields and messages for the **debugging cfd fsm** command.

Table 163 Output from the debugging cfd fsm command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AIS</td>
<td>Alarm indication signal state machine.</td>
</tr>
<tr>
<td>CCI</td>
<td>Continuity check state machine.</td>
</tr>
<tr>
<td>FNG</td>
<td>Error alarm state machine.</td>
</tr>
<tr>
<td>LBI</td>
<td>Loopback state machine.</td>
</tr>
<tr>
<td>MCC</td>
<td>Cross connection CCM state machine.</td>
</tr>
<tr>
<td>MME</td>
<td>Error CCM state machine.</td>
</tr>
<tr>
<td>RMEP</td>
<td>Remote MEP state machine.</td>
</tr>
<tr>
<td>FSM</td>
<td>State machine.</td>
</tr>
<tr>
<td>Port:<strong>interface-type</strong> interface-name</td>
<td>Port where the MEP is configured.</td>
</tr>
<tr>
<td>SI:SID</td>
<td>Service instance where the MEP belongs to.</td>
</tr>
<tr>
<td>MEP:MEPID</td>
<td>MEP.</td>
</tr>
<tr>
<td>State machine:<strong>State machine name</strong></td>
<td>Current state machine.</td>
</tr>
<tr>
<td>Prestate:<strong>State machine status</strong></td>
<td>Previous state of the state machine.</td>
</tr>
<tr>
<td>Curstate:<strong>State machine status</strong></td>
<td>Current state of the state machine.</td>
</tr>
</tbody>
</table>
Table 164 describes output fields and messages for the **debugging cfd packet** command.

### Table 164 Output from the debugging cfd packet command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Send/Receive CFD Packet via port <strong>interface-type</strong> <strong>interface-number</strong></td>
<td>The device sent/received a CFD packet through a port.</td>
</tr>
<tr>
<td>Pkt length</td>
<td>Packet length.</td>
</tr>
</tbody>
</table>

**Example**

# Enable debugging for EAIS. When you enable port status-AIS collaboration, and configure the EAIS frame transmission level and period, output similar to the following example is generated:

```
<Sysname> debugging cfd ais-track link-status packet
*0.99118462 Sysname EAIS/7/pkt:
Send EAIS Packet via Port Ethernet 1/1:
  00 21 04 00 00
Pkt length: 5
// The device sent an EAIS packet out port Ethernet1/1
```

# Enable the debugging for sent and received CFD packets. The output in this example was created when the following conditions exist:

- Configure level-0 MDs, MAs, and service instances on the device.
- Configure Ethernet 1/1 as an outbound MEP, setting the MEP ID to 100.
- Configure the AIS frame transmission level as 5 for the service instance.
- Enable the CFD service and CCM sending, and use the related commands to send LTM and LBM packets to the MED configured on the remote end.

```
<Sysname> debugging cfd packet
*0.99737167 Sysname CFD/8/Pkt:
Send CFD Packet via port Ethernet1/1:
  00 05 00 04 00 64 95 28 08 00 0d 40 00 e0 fc 00 65 10 00 e0 fc 00 65 65 00
Pkt length: 25
// The device sent a CFD packet out of Ethernet 1/1. The value **05** in the beginning of the packet indicates that the packet is a LTM packet.
```

```
*0.99977386 Sysname CFD/8/Pkt:
Send CFD Packet via port Ethernet1/1:
  00 03 00 04 00 c8 d5 64 01 00 00 00
Pkt length: 12
// The device sent a CFD packet out of Ethernet 1/1. The value **01** in the beginning of the packet indicates that the packet is a CCM packet.
```
// The device sent a CFD packet out of Ethernet 1/1. The value 03 in the beginning of the packet indicates that the packet is a LBM packet.

*0.17263340 Sysname CFD/8/Pkt:
Receive CFD Packet via port Ethernet1/1:
00 04 00 04 00 1a 77 f8 04 00 05 3f 00 00 00 00
05 00 07 00 00 e0 fc 52 ba a0 00
Pkt length: 27

// The device received a CFD packet through Ethernet 1/1. The value 04 in the beginning of the packet indicates that the packet is a LTR packet.

*0.84350000 Sysname CFD/8/Pkt:
Receive CFD Packet via port Ethernet1/1:
00 02 00 04 00 c8 d5 64 01 00 00 00 77 77 77 77
77 77 77 77 77 77 77 77 77 77 77 77 77 77
Pkt length: 46

// The device received a CFD packet through Ethernet 1/1. The value 02 in the beginning of the packet indicates that the packet is a LBR packet. The 7s in the rear of the packet were filled by the drive layer module to make the packet be of the specific length.

*0.99723467 Sysname CFD/7/Pkt:
Send CFD Packet via Port Ethernet 1/1:
00 2d 00 10 12 0e AB 2f 09 34 56 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 Pkt length: 21

// The device sent a CFD frame out of port Ethernet1/1. The value 2d in the beginning of the packet indicates that the packet is a 1DM packet.

*0.99737367 Sysname CFD/7/Pkt:
Send CFD Packet via Port Ethernet 1/1:
00 2f 00 20 12 0e AB 2f 09 34 56 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 00 00 00 Pkt length: 37

// The device sent a CFD frame out of port Ethernet1/1. The value 2f in the beginning of the packet indicates that the packet is a DMM packet.

*0.94874267 Sysname CFD/7/Pkt:
Receive CFD Packet via Port Ethernet 1/1:
00 2e 00 20 12 0e AB 2f 09 34 56 70 12 0e AB 2f 09 44 76 A0 12 0e AB 2f 09 44 76 A0 00
Pkt length: 37

// The device received a CFD frame on port Ethernet1/1. The value 2e in the beginning of the packet indicates that the packet is a DMR packet.

*0.99982603 Sysname CFD/7/Pkt:
Send CFD Packet via Port Ethernet 1/1:
00 21 04 00 00
Pkt length: 5

// The device sent a CFD frame out of port Ethernet1/1. The value 21 in the beginning of the packet indicates that the packet is an AIS packet.

*0.93433612 Sysname CFD/7/Pkt: Send CFD Packet via Port Ethernet 1/1:
// The device sent a CFD frame out of port Ethernet1/1. The value 25 in the beginning of the packet indicates that the packet is a TST packet.

*0.92437364 Sysname CFD/7/Pkt : Send CFD Packet via Port Ethernet 1/1:
00 2b 00 0c 00 00 00 2f 00 00 00 00 00 00 00 00
Pkt length: 17

// The device sent a CFD frame out of port Ethernet1/1. The value 2b in the beginning of the packet indicates that the packet is a LMM packet.

*0.96487912 Sysname CFD/7/Pkt : Send CFD Packet via Port Ethernet 1/1:
00 2a 00 0c 00 00 00 2f 00 00 00 1f 00 00 00 1a 00
Pkt length: 17

// The device sent a CFD frame out of port Ethernet1/1. The value 2a in the beginning of the packet indicates that the packet is a LMR packet.

# Enable CFD on the device and configure the corresponding MD, MA, and MEP. Enable debugging for CFD errors. Send packets carrying unknown TLVs to the configured MEP.

<Sysname> debugging cfd error
*0.99118462 Sysname CFD/8/Err: LMM/LTR Packet error: Package have unknown TLV

// The LTM packets/LTR packets received by CFD contained unknown TLVs.

# Enable CFD on the device and configure the corresponding MD, MA, MP, and MEP. Enable debugging for AIS. Enable AIS, and configure the AIS frame transmission level and period.

<Sysname> debugging cfd fsm ais
*0.99118462 Sysname CFD/7/Fsm:
SI: 1
MEP ID: 1001
State machine: AIS Prestate: IDLE Curstate: NO_RECEIVE

// The AIS state machine of CFD transited from the IDLE state to the NO_RECEIVE state.

# Enable CFD on the device and configure the corresponding MD, MA, MP, and MEP. Enable debugging for the CFD CCI state machine. Enable the configured MEP, and enable CC on the MEP.

<Sysname> debugging cfd fsm cci
*0.99118462 Sysname CFD/8/Fsm:
Port: Ethernet1/1 SI:1 MEP:140
State machine: CCI Prestate: CCI_IDLE Curstate: CCI_WAITING

// The CFD CCI state machine transited from the CCI_IDLE state to the CCI_WAITING state.

# Enable CFD on the device and configure the corresponding MD, MA, MP, and MEP. Enable debugging for the CFD MME state machine. The MEP on the device received the error CCM packets from other devices.

<Sysname> debugging cfd fsm mme
*0.99118462 Sysname CFD/8/Fsm:
Port: Ethernet1/1 SI:1 MEP:140
State machine: MME Prestate: ERRCCM_NO_DEFECT Curstate: ERRCCM_DEFECT
// The CFD MME state machine transited from the ERRCCM_NO_DEFECT state to the ERRCCM_DEFECT state.
Cluster management debugging commands

The output description tables in this document only contain fields and messages that require an explanation.

debugging ndp packet

Use **debugging ndp packet** to enable NDP packet debugging.

Use **undo debugging ndp packet** to disable NDP packet debugging.

**Syntax**

```
debugging ndp packet
undo debugging ndp packet
```

**Default**

NDP packet debugging is disabled.

**Views**

User view

**Default command level**

1: Monitor level

**Usage guidelines**

Table 165 describes output fields and messages for the **debugging ndp packet** command.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lenth</td>
<td>NDP packet length.</td>
</tr>
<tr>
<td>TTL</td>
<td>Aging time of an NDP packet on the receiving device.</td>
</tr>
</tbody>
</table>

**Examples**

```
# Enable debugging for NDP packets.
<Sysname> debugging ndp packet
*0.348945802 Sysname NDP/8/PKT:
  Interface Ethernet1/1 Send Packet Lenth: 165  TTL: 180
*0.348946139 Sysname NDP/8/PKT:
  Interface Ethernet1/2 Rcvd Packet Lenth: 138  TTL: 180

// The device sent an NDP packet out of Ethernet 1/1, and then it received an NDP packet from Ethernet 1/2.
```

debugging ntdp

Use **debugging ntdp** to enable NTDP debugging.

Use **undo debugging ntdp** to disable NTDP debugging.
Syntax

debugging ntdp { all | data | error | message | packet }
undo debugging ntdp { all | data | error | message | packet }

Default

NTDP debugging is disabled.

Views

User view

Default command level

1: Monitor level

Parameters

all: All types of NTDP debugging.
data: Debugging for NTDP data operation.
error: Debugging for NTDP errors.
message: Debugging for NTDP messages.
packet: Debugging for NTDP packets.

Usage guidelines

Table 166 describes output fields and messages for the debugging ntdp data command.

Table 166 Output from the debugging ntdp data command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Operation type.</td>
</tr>
</tbody>
</table>

Table 167 describes output fields and messages for the debugging ntdp error command.

Table 167 Output from the debugging ntdp error command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TYPE</td>
<td>Error type.</td>
</tr>
</tbody>
</table>

Table 168 describes output fields and messages for the debugging ntdp message command.

Table 168 Output from the debugging ntdp message command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Receiving Msg</td>
<td>NTDP received a message.</td>
</tr>
<tr>
<td>Sending Msg</td>
<td>NTDP sent a message.</td>
</tr>
</tbody>
</table>

Table 169 describes output fields and messages for the debugging ntdp packet command.

Table 169 Output from the debugging ntdp packet command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TYPE</td>
<td>Packet type.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>TTL</td>
<td>Hop count of the NTDP topology collection request packet.</td>
</tr>
<tr>
<td>Dest MAC</td>
<td>Destination MAC address.</td>
</tr>
<tr>
<td>Snd Port</td>
<td>Outgoing port.</td>
</tr>
<tr>
<td>Segment/Serial</td>
<td>Fragment information.</td>
</tr>
</tbody>
</table>

**Examples**

# Enable NTDP packet debugging.
```bash
<Sysname> debugging ntdp message
```

*0.349815585 Sysname NTDP/8/Debug_Rcv_Msg:
  Receiving Msg: Topology-Candidate Request

  // A topology collection request message was received from the candidate device.

*0.349816750 Sysname NTDP/8/Debug_Snd_Msg:
  Sending Msg:Candidate List

  // A list message was sent to the cluster module.

**debugging cluster**

Use `debugging cluster` to enable cluster debugging.

Use `undo debugging cluster` to disable cluster debugging.

**Syntax**

```bash
debugging cluster { all | event | handshake | member | mrc | nat | packet | state }
undo debugging cluster { all | event | handshake | member | mrc | nat | packet | state }
```

**Default**

Cluster debugging is disabled.

**Views**

User view

**Default command level**

1: Monitor level

**Parameters**

- `all`: All types of cluster debugging.
- `event`: Debugging for cluster events.
- `handshake`: Debugging for cluster handshake packets.
- `member`: Debugging for cluster member list. This debugging is not in use, and its content is output in log.
- `mrc`: Debugging for cluster Member Remote Control (MRC) packets.
- `nat`: Debugging for cluster NAT rule configurations.
- `packet`: Debugging for cluster management packets.
- `state`: Debugging for cluster state changes.
Usage guidelines

Table 170 describes output fields and messages for the debugging cluster event command.

**Table 170 Output from the debugging cluster event command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ucStringt</td>
<td>Information description.</td>
</tr>
</tbody>
</table>

Table 171 describes output fields and messages for the debugging cluster handshake command.

**Table 171 Output from the debugging cluster handshake command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>usClusterMode</td>
<td>The cluster mode of the device.</td>
</tr>
<tr>
<td>ucTopState</td>
<td>Flag of a topology change.</td>
</tr>
<tr>
<td>ucMemberMac</td>
<td>MAC address of the member device.</td>
</tr>
<tr>
<td>ucCommanderMac</td>
<td>MAC address of the management device.</td>
</tr>
<tr>
<td>ucMemberName</td>
<td>System name of the member device.</td>
</tr>
<tr>
<td>ucClusterName</td>
<td>Cluster name.</td>
</tr>
</tbody>
</table>

Table 172 describes output fields and messages for the debugging cluster member command.

**Table 172 Output from the debugging cluster member command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>One Member Added to MemberList</td>
<td>One member was deleted from the member list.</td>
</tr>
<tr>
<td>One Member Deleted to MemberList</td>
<td>One member was deleted from the member list.</td>
</tr>
<tr>
<td>ID</td>
<td>Member ID.</td>
</tr>
<tr>
<td>MAC</td>
<td>MAC address of the member device.</td>
</tr>
<tr>
<td>IP</td>
<td>Private IP address of the member device.</td>
</tr>
</tbody>
</table>

Table 173 describes output fields and messages for the debugging cluster mrc command.

**Table 173 Output from the debugging cluster mrc command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ucMAC</td>
<td>MAC address of a managed device.</td>
</tr>
<tr>
<td>ucTTL</td>
<td>Number of hops the packet traveled in the network.</td>
</tr>
<tr>
<td>ucClusterCommandMAC</td>
<td>MAC address of the cluster management device.</td>
</tr>
<tr>
<td>ucControlByte</td>
<td>Control command.</td>
</tr>
<tr>
<td>ulParaLen</td>
<td>Command parameter length.</td>
</tr>
<tr>
<td>ucPara</td>
<td>Length-variable parameter.</td>
</tr>
<tr>
<td>ulMaxHops</td>
<td>Maximum number of hops of the management device.</td>
</tr>
<tr>
<td>ulSurviveTime</td>
<td>The reserving time of the management VLAN information.</td>
</tr>
</tbody>
</table>
### Table 174: Output from the debugging cluster nat command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ulExtendedLen</td>
<td>Length of the extended field.</td>
</tr>
<tr>
<td>CheckSumVal</td>
<td>Total checksum of the cluster data.</td>
</tr>
<tr>
<td>CmdFlag</td>
<td>Return flag.</td>
</tr>
<tr>
<td>CmdRec</td>
<td>Return word.</td>
</tr>
</tbody>
</table>

Table 174 describes output fields and messages for the **debugging cluster nat** command.

### Table 175: Output from the debugging cluster packet command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAT_Server_Add</td>
<td>A NAT server was added.</td>
</tr>
<tr>
<td>NAT_Server_Del</td>
<td>A NAT server was removed.</td>
</tr>
<tr>
<td>IfIndex</td>
<td>Interface index.</td>
</tr>
<tr>
<td>GlobalPort</td>
<td>Global port number.</td>
</tr>
<tr>
<td>GlobalIP</td>
<td>Global IP address.</td>
</tr>
<tr>
<td>InsidePort</td>
<td>Internal port number.</td>
</tr>
<tr>
<td>InsideIP</td>
<td>Internal IP address.</td>
</tr>
<tr>
<td>Protocol</td>
<td>Protocol number.</td>
</tr>
</tbody>
</table>

Table 175 describes output fields and messages for the **debugging cluster packet** command.
Table 176 describes output fields and messages for the `debugging cluster state` command.

**Table 176 Output from the debugging cluster state command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>Member ID.</td>
</tr>
<tr>
<td>Pre state</td>
<td>The previous state.</td>
</tr>
<tr>
<td>State</td>
<td>The current state.</td>
</tr>
<tr>
<td>Time</td>
<td>Hold time of the current state.</td>
</tr>
</tbody>
</table>

**Examples**

# Enable cluster packet debugging.

```bash
<Sysname> debugging cluster packet
*0.93214687 h3c_0.Sysname CLST/8/Debug_Send_Pkt:
    Sending cluster packet:
    Clst-head :
    Ver: 2  Mode: 0 PktTyte: 1
    FrmNoFlag: 0 FrameNo: 0
    CheckSumFlag : 1 CheckSumVal :47184
    CmdFlag : 0 CmdRec: 0
    SrcMac :00e0-fc58-c438
    DstMac :00e0-fc00-5502,
    Mgn_head :
    MainCmd: 1 SubCmd: 1 DateLen: 36,
    Content :
    0-- 68 33 63 00 00 00 00 00 00 00 00 00 00 00
    1-- 00 00 00 00 00 00 00 00 00 00 00 00 00 00
    2-- 00 00 00 00 00 00 00 00 00 00 00 00 00 00
    3-- 00 00 00 00 00 00 00 00 00 00 00 00 00 00
    4-- 00 01 00 0

// The management device sent a packet cluster packet.
```

# Enable NTDP packet debugging.

```bash
<Sysname> debugging ntdp packet
*Apr 27 04:48:33:871 2000 Sysname NTDP/7/Debug_Send_Pkt:
    Sending NTDP packet:TYPE-- Request  TTL--3  Dest MAC--Multicast  Snd Port--Ethernet1/1
    Segment/Serial--00/00

// An NTDP topology collection request was sent.
```
debugging cluster ext

Use `debugging cluster ext` to enable SNMP configuration synchronization debugging. Use `undo debugging cluster ext` to disable SNMP configuration synchronization debugging.

**Syntax**

```
debugging cluster ext { event | packet }
undo debugging cluster ext { event | packet }
```

**Default**

SNMP configuration synchronization debugging is disabled.

**Views**

User view

**Default command level**

1: Monitor level

**Parameters**

- `event`: Debugging for SNMP configuration synchronization events.
- `packet`: Debugging for SNMP configuration synchronization packets.

**Usage guidelines**

Table 177 describes output fields and messages for the `debugging cluster ext event` command.

**Table 177 Output from the debugging cluster ext event command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Error: packet length is error</td>
<td>The packet length is incorrect.</td>
</tr>
</tbody>
</table>

Table 178 describes output fields and messages for the `debugging cluster ext packet` command.

**Table 178 Output from the debugging cluster ext packet command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sending/Receiving Extension Pkt</td>
<td>An extended packet was sent or received.</td>
</tr>
<tr>
<td>MainCmd</td>
<td>Main command word, which indicates the packet type:</td>
</tr>
<tr>
<td></td>
<td>0—SNMP configuration synchronization request.</td>
</tr>
<tr>
<td></td>
<td>1—SNMP configuration synchronization response.</td>
</tr>
<tr>
<td>SrcMAC</td>
<td>Source MAC address of the packet.</td>
</tr>
<tr>
<td>DstMAC</td>
<td>Destination MAC address of the packet.</td>
</tr>
<tr>
<td>ParaLen</td>
<td>Length of the sub-packet.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>PortIndex</td>
<td>Port index.</td>
</tr>
<tr>
<td>PortName</td>
<td>Port name.</td>
</tr>
<tr>
<td>SynSubCmd</td>
<td>Command word of the synchronization sub-packet:</td>
</tr>
<tr>
<td></td>
<td>• 0—Configures a read community.</td>
</tr>
<tr>
<td></td>
<td>• 1—Configures a write community.</td>
</tr>
<tr>
<td></td>
<td>• 2—Configures an SNMPv3 group name.</td>
</tr>
<tr>
<td></td>
<td>• 3—Creates or updates the MIB view that is shared by the cluster.</td>
</tr>
<tr>
<td></td>
<td>• 4—Adds a user to the SNMPv3 group that is shared by the cluster.</td>
</tr>
<tr>
<td>ulFlag</td>
<td>Undo flag:</td>
</tr>
<tr>
<td></td>
<td>• 0—Not to undo.</td>
</tr>
<tr>
<td></td>
<td>• 1—To undo.</td>
</tr>
<tr>
<td>ComName</td>
<td>Community name.</td>
</tr>
<tr>
<td>ComView</td>
<td>MIB view for the community name.</td>
</tr>
<tr>
<td>ComReadFlag</td>
<td>Flags of the SNMP configuration synchronization results:</td>
</tr>
<tr>
<td>ComWriteFlag</td>
<td>• 0x00—Configuration succeeded.</td>
</tr>
<tr>
<td>GroupFlag</td>
<td>• 0x01—Configuration failed.</td>
</tr>
<tr>
<td>MibViewFlag</td>
<td>• 0xff—No change.</td>
</tr>
<tr>
<td>UsmUserFlag</td>
<td>Flag of packet authentication and privacy:</td>
</tr>
<tr>
<td></td>
<td>• 1—Authentication, no privacy.</td>
</tr>
<tr>
<td></td>
<td>• 2—Authentication and privacy.</td>
</tr>
<tr>
<td></td>
<td>• 3—No authentication, no privacy.</td>
</tr>
<tr>
<td>GroupName</td>
<td>SNMPv3 group name.</td>
</tr>
<tr>
<td>GroupRView</td>
<td>Read only MIB view accessible to the SNMP group.</td>
</tr>
<tr>
<td>GroupWView</td>
<td>Write MIB view accessible to the SNMP group.</td>
</tr>
<tr>
<td>GroupNView</td>
<td>Notify MIB view for the SNMP group. The SNMP users in the group can send</td>
</tr>
<tr>
<td></td>
<td>traps only for the nodes in the notify MIB view.</td>
</tr>
<tr>
<td>AuthFlag</td>
<td>Flag of packet authentication and privacy:</td>
</tr>
<tr>
<td></td>
<td>• 1—Authentication, no privacy.</td>
</tr>
<tr>
<td></td>
<td>• 2—Authentication and privacy.</td>
</tr>
<tr>
<td></td>
<td>• 3—No authentication, no privacy.</td>
</tr>
<tr>
<td>MibViewName</td>
<td>MIB view name.</td>
</tr>
<tr>
<td>MibViewTree</td>
<td>ID of the MIB subtree for the MIB view.</td>
</tr>
<tr>
<td>IncludeFlag</td>
<td>Access privilege flag for the MIB subtree in the MIB view:</td>
</tr>
<tr>
<td></td>
<td>• 1—Included. All objects in the MIB subtree are accessible in the MIB</td>
</tr>
<tr>
<td></td>
<td>view.</td>
</tr>
<tr>
<td></td>
<td>• 2—Not included. None of the objects in the MIB subtree is accessible in</td>
</tr>
<tr>
<td></td>
<td>the MIB view.</td>
</tr>
<tr>
<td>SnmpName</td>
<td>Name of the newly added SNMPv3 group user.</td>
</tr>
<tr>
<td>GroupName</td>
<td>Name of the SNMPv3 group to which the new user was added.</td>
</tr>
<tr>
<td>AuthPass</td>
<td>Authentication password of the newly added SNMPv3 group user.</td>
</tr>
<tr>
<td>PrivPass</td>
<td>Privacy password of the newly added SNMPv3 group user.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>AuthFlag</td>
<td>Authentication mode of the newly added SNMPv3 group user:</td>
</tr>
<tr>
<td></td>
<td>• 1—MD5.</td>
</tr>
<tr>
<td></td>
<td>• 2—SHA.</td>
</tr>
<tr>
<td>PacketLen</td>
<td>Data length.</td>
</tr>
</tbody>
</table>

**Examples**

In the following examples, Device A is the management device in a cluster and it is communicating with a managed member device on the whitelist.

# Enable SNMP configuration synchronization event debugging on both Device A and its managed device. When a command is issued to create a read-community for the cluster, output similar to the following example is generated if the configuration is not successful:

```
<DeviceA> debugging cluster ext event
<DeviceA> system-view
[DeviceA] cluster
[DeviceA-cluster] cluster-snmp-agent community read read
*0.27122800 DeviceA CLST/7/Debug_Cluster_Ext_Info:
  Cluster Extension Information:
  : Error: packet length is error.
  // The management device received an illegal packet.
*0.27122800 DeviceA CLST/7/Debug_Cluster_Ext_Info:
  Cluster Extension Information:
  : Failed to config command on local device.
  // Configuration failed because SNMP configuration on the management device is full or for other reasons.
*0.27122800 DeviceA CLST/7/Debug_Cluster_Ext_Info:
  Cluster Extension Information:
  : WEB: Failed to send packet.
  // Cluster management failed to send the packet because of interface error.
```

# Enable SNMP configuration synchronization packet debugging on Device A. When a read-community is created for the cluster, output similar to the following example is generated:

```
<DeviceA> debugging cluster ext packet
<DeviceA> system-view
[DeviceA] cluster
[DeviceA-cluster] cluster-snmp-agent community read read
*0.5614402 DeviceA CLST/8/Debug_Ext_Pkt_Send:
  Send Extension Pkt:
  MainCmd : 0
  SrcMAC : 00e0-fc00-3000
```
Member 1 succeeded in the read-community configuration.
Finish to synchronize the command.

```
// Device A sent a request to configure the SNMP read-community on the managed device.

*0.5614740 DeviceA CLST/8/Debug_Ext_Pkt_Recv:
Receiving Extension Pkt:

MainCmd : 1
SrcMAC : 00e0-fc00-2000
DstMAC : 00e0-fc00-3000
ParaLen : 21
PortIndex : 1
PortName : Ethernet1/1

SynSubCmd: 0
ulFlag: 0
PacketLen : 6
ComReadFlag : 0x00
ComWriteFlag: 0xff
GroupFlag : 0xff
MibViewFlag : 0xff
UsmUserFlag : 0xff
WebUserFlag : 0xff
```

// Device A received from managed device the response of successful SNMP read-community configuration.
Connection limit debugging commands

The output description tables in this document only contain fields and messages that require an explanation.

debugging connection-limit

Use **debugging connection-limit** to enable debugging for connection limit.

Use **undo debugging connection-limit** to disable debugging for connection limit.

**Syntax**

```
debugging connection-limit
undo debugging connection-limit
```

**Default**

Debugging for connection limit is disabled.

**Views**

User view

**Default command level**

1: Monitor level

**Usage guidelines**

Table 179 describes output fields and messages for the **debugging connection-limit** command.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ConnLmt Query:</td>
<td>Connection request from a user.</td>
</tr>
<tr>
<td>Protocol</td>
<td>Protocol number.</td>
</tr>
<tr>
<td>Sourcelp</td>
<td>Source IP address.</td>
</tr>
<tr>
<td>DestinationIp</td>
<td>Destination IP address.</td>
</tr>
<tr>
<td>Vpn</td>
<td>MPLS VPN instance to which the source IP address belongs.</td>
</tr>
<tr>
<td>Policy</td>
<td>Connection limit policy index.</td>
</tr>
<tr>
<td>Limit</td>
<td>Connection limit rule index.</td>
</tr>
</tbody>
</table>

**Stat-Type**

Connection statistics type:

- **1**—Collects statistics by source address.
- **2**—Collects statistics by destination address.
- **4**—Collects statistics by service type.
- **7**—Collects statistics by combination of source address, destination address, and service type.
- **8**—Collects statistics by source address and uses the global connection limits (the default).
### Field Description

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Current-amount</td>
<td>Number of connections.</td>
</tr>
<tr>
<td>ConnLmt Inc:</td>
<td>Connection limit statistics entry is incremented by 1.</td>
</tr>
<tr>
<td>Limit-Flag</td>
<td>Whether new connections are allowed:</td>
</tr>
<tr>
<td></td>
<td>• 0—Yes.</td>
</tr>
<tr>
<td></td>
<td>• 1—No.</td>
</tr>
<tr>
<td>The limit limit-id is enabled.</td>
<td>Connection limit rule limit-id has been enabled successfully.</td>
</tr>
<tr>
<td>The limit limit-id is disabled.</td>
<td>Connection limit rule limit-id has been disabled successfully.</td>
</tr>
<tr>
<td>Matched limit: limit-id.</td>
<td>Connection limit rule limit-id is matched.</td>
</tr>
</tbody>
</table>

### Examples

- **Single core device:**

  # On a NAT-enabled device, enable connection limit debugging. Output similar to the following example is generated when a user initiates a connection under the following conditions:

  - Connection limit policy 0 is configured on the device.
  - Connection limit rule 0 is configured in the policy to collect statistics on and limit user connections from the segment 192.168.0.0/24. The upper limit and lower limit is 1 and 0. The connection statistics type is by source address.

  <Sysname> debugging connection-limit
  *0.1295578 Sysname CONNLMT/7/debug:
  ConnLmt Query: Policy:0 Stat-type:1
  Connection limit received a connection request from 192.168.0.210 to 2.2.2.2. The source IP address does not belong to any MPLS VPN instance.

  *0.1295578 Sysname CONNLMT/7/debug:
  ConnLmt Query: Stat-type:1
  Connection limit rule 0 in connection limit policy 0 was applied to user connections.

  *0.1295578 Sysname CONNLMT/7/debug:
  ConnLmt Query: Stat-type:1
  The connection statistics type is 1, which means connection limit limits connections by source address.

  *0.1295578 Sysname CONNLMT/7/debug:
  ConnLmt Query: Current amount of connection is 0!

  *0.1295578 Sysname CONNLMT/7/debug:
  ConnLmt Inc: Protocol:6 SourceIp:192.168.0.210 DestinationIp:2.2.2.2
  SourcePort:1405 DestinationPort:21 Vpn:None Stat-type:1
  A user connection between 192.168.0.210 and 2.2.2.2 was established.

  *0.1313500 Sysname CONNLMT/7/debug:
  ConnLmt Query: Protocol:1 SourceIp:192.168.0.210 DestinationIp:2.2.2.2
  SourcePort:None DestinationPort:None Vpn:None
  A connection request from 192.168.0.210 to 2.2.2.2 arrived. The source IP address does not belong to any MPLS VPN instance.

  *0.1313500 Sysname CONNLMT/7/debug:
  Current-amount:1 Limit-Flag:1!
// There is one connection at present, and the number of connections has reached the upper limit. No more connections are allowed.

- Multi-core device:
  # Enable connection limit debugging on the device. Output similar to the following example is generated when TCP connections are set up between the source and the destination under the following condition: Connection limit policy 0 with connection limit rule 0 is configured.
  <Sysname> debugging connection-limit
  *Jun 30 16:43:56:971 2008 Sysname DPCONNLMT/7/CONNLMT_ENABLE_LIMIT: The limit 0 is enabled.
  // Connection limit rule 0 was enabled.
  // Connection limit rule 0 was matched.
Content filtering debugging commands

debugging content-filter

NOTE:
Support for the command depends on the device model.

Use debugging content-filter to enable debugging for content filtering.
Use undo debugging content-filter to disable debugging for content filtering.

Syntax
debugging content-filter { all | error | event | packet } [ acl acl-number ]
undo debugging content-filter { all | error | event | packet }

Default
Debugging for content filtering is disabled.

Views
User view

Default command level
1: Monitor level

Parameters
  all: Specifies all types of content filtering debugging.
  error: Specifies error debugging of content filtering.
  event: Specifies event debugging of content filtering.
  packet: Specifies packet debugging of content filtering.
  acl acl-number: Specifies the ACL to be used to filter the debugging information. The acl-number argument is the ACL number in the range of 2000 to 3999.

Usage guidelines
If you execute the command multiple times and specify different ACLs, the most recent configuration takes effect.

Examples
The output in the following examples was created under the following conditions:
- On the firewall, configure a content filtering policy template policy1, which includes an HTTP filtering policy and an SMTP filtering policy. In the HTTP filtering policy, URL IP blocking is enabled.
- On the firewall, configure an interzone policy named aaa, with the source zone Trust, destination zone Untrust, and filter action Permit.

# Enable content filtering event debugging. When a host in zone Trust accesses an HTTP server in zone Untrust through the server’s IP address, output similar to the following example is generated:
<sysname> debugging content-filter event
HTTP URL IP filtering dropped the packet.

// HTTP URL IP blocking dropped the packet.

# Enable packet debugging of content filtering. When a host in zone Trust sends an email to a mail server in zone Untrust, output similar to the following example is generated:

<sysname> debugging content-filter packet

*May 14 17:48:30:141 2009 H3C DPDPI/7/PACKET:
Receiving packet. Session state : 0 , Application type : 5

// An SMTP packet was received. The session state of the packet is 0, and the application type is 5.

# Enable error debugging of content filtering. When a host in zone Trust sends an HTTP packet with the packet length over 1500 bytes to the server in zone Untrust, output similar to the following example is generated:

<sysname> debugging content-filter error

*May 14 17:48:30:141 2009 H3C DPDPI/7/ERROR:
Invalid HTTP packet length.

// HTTP packet length is invalid.
COPS debugging commands

debugging cops client-type

Syntax

Centralized devices:

debugging cops client-type dot1x \{ all | error | event | packet \}
undo debugging cops client-type dot1x \{ all | error | event | packet \}

Distributed devices—Centralized IRF devices:

debugging cops client-type dot1x \{ all | error | event | packet \} \{ slot slot-number \}
undo debugging cops client-type dot1x \{ all | error | event | packet \} \{ slot slot-number \}

Distributed IRF devices:

debugging cops client-type dot1x \{ all | error | event | packet \} \{ chassis chassis-number slot slot-number \}
undo debugging cops client-type dot1x \{ all | error | event | packet \} \{ chassis chassis-number slot slot-number \}

Default

Debugging for COPS is disabled.

Views

User view

Default command level

1: Monitor level

Parameters

client-type: Specifies debugging for client services.
dot1x: Specifies debugging for the 802.1X service.
all: Specifies all types of debugging for COPS.
error: Specifies debugging for errors.
event: Specifies debugging for events.
packet: Specifies debugging for packets.
slot slot-number: Specifies debugging for a card. The slot-number argument represents the number of the slot that holds the card. (Distributed devices.)
slot slot-number: Specifies debugging for an IRF member device. The slot-number argument represents the ID of the IRF member device in the IRF fabric. (Centralized IRF devices.)
chassis chassis-number slot slot-number: Specifies debugging for a card in an IRF member device. The chassis-number argument represents the ID of the IRF member device, and the slot-number argument represents the number of the slot that holds the card. (Distributed IRF devices.)
Usage guidelines

Use the **debugging cops** command to enable debugging for COPS.

Use the **undo debugging cops** command to disable debugging for COPS.

Table 180 describes output fields and messages for the **debugging cops client-type dot1x error** command.

### Table 180 Output from the debugging cops client-type dot1x error command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Failed to create packet</td>
<td>Failed to create COPS packet, which can be SSC, DRQ, KA, CC, REQ, OPN, or RPT.</td>
</tr>
<tr>
<td>Failed to send packet</td>
<td>Failed to send COPS packet, which can be SSC, DRQ, KA, CC, REQ, OPN, or RPT.</td>
</tr>
<tr>
<td>Failed to process packet</td>
<td>Failed to process COPS packet, which can be CAT, CC, SSQ, or DEC.</td>
</tr>
<tr>
<td>Failed to malloc connection struct</td>
<td>Failed to allocated connection structure.</td>
</tr>
</tbody>
</table>

0 describes output fields and messages for the **debugging cops event** command.

### Table 181 Output from the debugging cops event command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Creating packet</td>
<td>Created a COPS packet, which can be SSC, DRQ, KA, CC, REQ, OPN, or RPT.</td>
</tr>
<tr>
<td>Sending packet</td>
<td>Sent a COPS packet, which can be SSC, DRQ, KA, CC, REQ, OPN, or RPT.</td>
</tr>
<tr>
<td>Processing packet</td>
<td>Processed a COPS packet, which can be CAT, CC, SSQ, or DEC.</td>
</tr>
</tbody>
</table>

0 describes output fields and messages for the **debugging cops packet** command.

### Table 182 Output from the debugging cops packet command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet sent:</td>
<td>Information of the sent packet, including a packet header and multiple objects.</td>
</tr>
<tr>
<td>Packet received:</td>
<td>Information of the received packet, including a packet header and multiple objects.</td>
</tr>
<tr>
<td>Header:</td>
<td>Packet header information, including:</td>
</tr>
<tr>
<td>version</td>
<td>version.</td>
</tr>
<tr>
<td>flags</td>
<td>flag.</td>
</tr>
<tr>
<td>op-code</td>
<td>pkt-type.</td>
</tr>
<tr>
<td>client-type</td>
<td>client-type.</td>
</tr>
<tr>
<td>msg-length</td>
<td>msg-len.</td>
</tr>
<tr>
<td>obj-len</td>
<td>Object information, including:</td>
</tr>
<tr>
<td>c-num</td>
<td>obj-len: Object length.</td>
</tr>
<tr>
<td>c-type</td>
<td>class-num Class ID.</td>
</tr>
<tr>
<td>data</td>
<td>class-type: Object type.</td>
</tr>
<tr>
<td>contexts</td>
<td>contexts: Object contents.</td>
</tr>
</tbody>
</table>

Examples

# Enable debugging for COPS events. Output similar to the following example is generated when a COPS scheme is referenced for 802.1X:
<Sysname> debugging cops client-type dot1x event
*Feb 8 11:08:28:046 2010 Sysname COPS/7/EVENT:
Creating Timer, OPN timer created.
*Feb 8 11:08:28:078 2010 Sysname COPS/7/EVENT:
Creating Timer, KA send-timer created.
*Feb 8 11:08:28:078 2010 Sysname COPS/7/EVENT:
Creating Timer, KA wait-timer created.
*Feb 8 11:08:28:078 2010 Sysname COPS/7/EVENT:
Creating client type, set client connection state to DOWN.

# Enable debugging for COPS errors. Output similar to the following example is generated when a
COPS scheme that does not exist is referenced for 802.1X:
<Sysname> debugging cops client-type dot1x error
*Feb 8 11:08:28:078 2010 Sysname COPS/7/ERROR:
Failed to get scheme.

# Enable debugging for COPS packets. Output similar to the following example is generated when a
COPS scheme is referenced for 802.1X:
<Sysname> debugging cops client-type dot1x packet
*Mar 5 17:01:38:547 2010 Sysname COPS/7/PACKET: Packet sent:
Header: version=1; flags=0; op-code=1; client-type=16385; msg-length=56
obj-len=12; c-num=1; c-type=1; data=0e 00 00 00 5b 03 00 00 5b 03 00 00
obj-len=8; c-num=2; c-type=1; data=01 00 02 00
obj-len=26; c-num=9; c-type=1; data=03 00 03 61 62 63 06 00 0d 45 74 68 65 74 30 2f 34 2f 31
74 30 2f 34 2f 31
Customizable IVR debugging commands

The output description tables in this document only contain fields and messages that require an explanation.

debugging voice ivr fes

Use debugging voice ivr fes to enable FES debugging in the IVR system.

Use undo debugging voice ivr fes to disable FES debugging in the IVR system.

Syntax

debugging voice ivr fes { all | error | event | fsm | info | timer }
undo debugging voice fes { all | error | event | fsm | info | timer }

Default

FES debugging is disabled.

Views

User view

Default command level

2: System level

Parameters

all: Specifies all types of debugging for the flow execute system (FES).
error: Specifies error debugging.
event: Specifies event debugging.
fsm: Specifies finite state machine debugging.
info: Specifies information debugging.
timer: Specifies timer debugging.

Usage guidelines

Table 183 describes output fields and messages for the debugging voice ivr fes error command.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IVR Node is not service node.</td>
<td>The current node was not a Service node.</td>
</tr>
<tr>
<td>The call number is null, LocalId = %d.</td>
<td>No number was configured for the node.</td>
</tr>
<tr>
<td>Channel ready result is failure, CallId = %d.</td>
<td>The IVR failed to open media channel.</td>
</tr>
</tbody>
</table>

Table 184 describes output fields and messages for the debugging voice ivr fes event command.
### Table 184 Output from the debugging voice ivr fes event command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CMC --&gt; IVR : Accp Setup.</td>
<td>The IVR received an Accp Setup message from the CMC.</td>
</tr>
<tr>
<td>CMC --&gt; IVR : Accp Channel Ready Ack.</td>
<td>The IVR received an Accp Channel Ready Ack message from the CMC.</td>
</tr>
<tr>
<td>CMC --&gt; IVR : Accp Fax Voice Switch.</td>
<td>The IVR received an Accp Fax Voice Switch message from the CMC.</td>
</tr>
<tr>
<td>CMC --&gt; IVR : Accp Information.</td>
<td>The IVR received an Accp Information message from the CMC.</td>
</tr>
<tr>
<td>CMC --&gt; IVR : Accp Service.</td>
<td>The IVR received an Accp Service message from the CMC.</td>
</tr>
<tr>
<td>CMC --&gt; IVR : Accp Service Ack.</td>
<td>The IVR received an Accp Service Ack message from the CMC.</td>
</tr>
<tr>
<td>CMC --&gt; IVR : Accp Release.</td>
<td>The IVR received an Accp Release message from the CMC.</td>
</tr>
<tr>
<td>CMC --&gt; IVR : Accp Release Complete.</td>
<td>The IVR received an Accp Release Complete message from the CMC.</td>
</tr>
<tr>
<td>IVR --&gt; CMC: Accp SetupAck.</td>
<td>The IVR sent an Accp Setup Ack message to the CMC.</td>
</tr>
<tr>
<td>IVR --&gt; CMC: Accp Alert.</td>
<td>The IVR sent an Accp Alerting message to the CMC.</td>
</tr>
<tr>
<td>IVR --&gt; CMC: Accp Connect.</td>
<td>The IVR sent an Accp Connect message to the CMC.</td>
</tr>
<tr>
<td>FES --&gt; CMC: Accp Information.</td>
<td>The IVR sent an Accp Information message to the CMC.</td>
</tr>
<tr>
<td>IVR --&gt; CMC: Accp Release.</td>
<td>The IVR sent an Accp Release message to the CMC.</td>
</tr>
<tr>
<td>IVR --&gt; CMC: Accp RelComp.</td>
<td>The IVR sent an Accp Release Complete message to the CMC.</td>
</tr>
<tr>
<td>IVR --&gt; CMC: Accp Channel Ready.</td>
<td>The IVR sent an Accp Channel Ready message to the CMC.</td>
</tr>
<tr>
<td>IVR --&gt; CMC: Accp FaxVocSwch Ack.</td>
<td>The IVR sent an Accp Fax Voice Switch Ack message to the CMC.</td>
</tr>
<tr>
<td>IVR --&gt; CMC: Accp Service.</td>
<td>The IVR sent an Accp Service message to the CMC.</td>
</tr>
<tr>
<td>IVR --&gt; CMC: Accp SrvAck.</td>
<td>The IVR sent an Accp Service Ack message to the CMC.</td>
</tr>
</tbody>
</table>

Table 185 describes output fields and messages for the **debugging voice ivr fes fsm** command.

### Table 185 Output from the debugging voice ivr fes fsm command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>%s --&gt; %s, CallId = %d.</td>
<td>The IVR call status changed from x to xx, and the call ID is xxx.</td>
</tr>
</tbody>
</table>

Table 186 describes output fields and messages for the **debugging voice ivr fes info** command.
### Table 186 Output from the debugging voice ivr fes info command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input error, RepeatTimes = %d, InputErrorTimes = %d.</td>
<td>Subscriber input errors: the allowed maximum number of repeated input errors was xx, and the actual number of repeated input errors was xxx.</td>
</tr>
<tr>
<td>Timeout, RepeatTimes = %d, TimeoutTimes = %d.</td>
<td>Subscriber timeout time: the allowed maximum number of repeated subscriber timeout was xx, and the actual number of repeated subscriber timeout was xxx.</td>
</tr>
<tr>
<td>Jump configure is invalid, Nodeld = %d.</td>
<td>The configuration of the Jump node was invalid.</td>
</tr>
<tr>
<td>Call configure is invalid, Nodeld = %d.</td>
<td>The configuration of the Call node was invalid.</td>
</tr>
<tr>
<td>Service configure is invalid, Nodeld = %d.</td>
<td>The configuration of the Service node was invalid.</td>
</tr>
<tr>
<td>(%s) -&gt; (%s), LocalId = %d.</td>
<td>The IVR operation state changed from x to xx, and the local ID was xxx.</td>
</tr>
</tbody>
</table>

### Examples

# Enable all types of debugging for the FES. The output similar to the following example was created when the following conditions exist:

- The root node of the IVR system is a Call node with a node ID of 10.
- The root node is configured to play a media file named `callSecDial_g729.wav` (media resource ID is 1000) and originate normal secondary calls by matching the terminator #.

```
<Sysname> debugging voice ivr fes all
Enable IVR_FES all debugging functions
*Jun 27 11:20:26:75 2008 71_C IVRF/7/VOICE:
*Jun 27 11:20:26:75 2008 71_C IVRF/7/VOICE:
// The IVR_FES sent an Accp SetupACK message to the CMC.
*Jun 27 11:20:26:75 2008 71_C IVRF/7/VOICE:
// The IVR_FES sent an Accp Alert message to the CMC.
*Jun 27 11:20:26:76 2008 71_C IVRF/7/VOICE:
// The IVR_FES sent an Accp Connect message to the CMC.
*Jun 27 11:20:26:76 2008 71_C IVRF/7/VOICE:
IVR_FES_Timer: Start timer, TmrId = 977, TmrType = WAIT CHANNEL READY ACK , TmrLength = 150000.
// The timer for receiving a Channel Ready Ack message was started.
```
IVR_FES_Fsm: IDLE --> WAIT_CHY_ACK, CallId = 131073, LocalId = 0.

// The state of the call FSM changed from IDLE to WAIT_CHY_ACK.
*Jun 27 11:20:26:79 2008 71_C IVRF/7/VOICE:
IVR_FES_Event: CMC --> FES : Accp Channel Ready Ack.

// The IVR_FES received an Accp Channel Ready Ack message from the CMC.
*Jun 27 11:20:26:80 2008 71_C IVRF/7/VOICE:
IVR_FES_Timer: Delete Timer, TmrId = 977, TmrType = WAIT CHANNEL READY ACK.

// The timer for receiving a Channel Ready Ack message was deleted.
*Jun 27 11:20:26:80 2008 71_C IVRF/7/VOICE:
IVR_FES_Event: FES --> CMC: Accp Information.

// The IVR_FES sent an Accp Information message to the CMC.
*Jun 27 11:20:26:80 2008 71_C IVRF/7/VOICE:
IVR_FES_Fsm: WAIT_CHY_ACK --> ACTIVE, CallId = 131073, LocalId = 0.

// The state of the call FSM changed from WAIT_CHY_ACK to ACTIVE.
*Jun 27 11:20:26:80 2008 71_C IVRF/7/VOICE:
IVR_FES_Info: The node id is not in the stack, NodeId = 10.

// Node 10 was not in the node stack and was pushed into the stack.
*Jun 27 11:20:26:90 2008 71_C IVRF/7/VOICE:
IVR_FES_Timer: Start Timer, TmrId = 154, TmrType = CALL WAIT INPUT, TmrLength = 10000.

// The subscriber input timeout timer under the node was started.
*Jun 27 11:20:30:322 2008 71_C IVRF/7/VOICE:
IVR_FES_Info: CALL_PLAY_MEDIA --> CALL_WAIT_INPUT, LocalId = 0.

// The state of the node FSM changed from CALL_PLAY_MEDIA to CALL_WAIT_INPUT.

• Output similar to the following messages is generated when the subscriber (500) hangs up without originating a secondary call:
  *Jun 27 11:20:34:42 2008 71_C IVRF/7/VOICE:

// The IVR_FES received an Accp Release message from the CMC.
  *Jun 27 11:20:34:42 2008 71_C IVRF/7/VOICE:
  IVR_FES_Event: FES --> CMC: Accp RelComp.

// The IVR_FES sent an Accp Release Complete message to the CMC.
  *Jun 27 11:20:34:43 2008 71_C IVRF/7/VOICE:
  IVR_FES_Timer: Delete Timer, TmrId = 154, TmrType = CALL WAIT INPUT.

// The subscriber input timeout timer under the node was deleted.

• Output similar to the following messages is generated when the subscriber at 500 dials 600# to originate a secondary call and the subscriber at 600 picks up the ringing phone.
  *Jun 27 12:05:11:214 2008 71_C IVRF/7/VOICE:
  IVR_FES_Event: CMC --> FES : Accp Information.

// After the subscriber presses key 6, the IVR_FES received an Accp Information message from CMC.
  *Jun 27 12:05:11:214 2008 71_C IVRF/7/VOICE:
  IVR_FES_Timer: Delete Timer, TmrId = 325, TmrType = CALL WAIT INPUT.
// The subscriber input timeout timer under the node was deleted.
*Jun 27 12:05:11:215 2008 71_C IVRF/7/VOICE:
IVR_FES_Timer: Start Timer, TmrId = 250, TmrType = CALL WAIT INPUT INTERVAL, TmrLength = 10000.
// The timer specifies the interval between key-presses under the Call node was started.
*Jun 27 12:05:11:215 2008 71_C IVRF/7/VOICE:
IVR_FES_Info: CALL_WAIT_INPUT --> CALL_WAIT_INPUT, LocalId = 0.
// The state of the node FSM was CALL_WAIT_INPUT.
*Jun 27 12:05:12:114 2008 71_C IVRF/7/VOICE:
IVR_FES_Event: CMC --> FES : Accp Information.
// After the subscriber pressed key 0, the IVR_FES received an Accp Information message from CMC.
*Jun 27 12:05:12:114 2008 71_C IVRF/7/VOICE:
IVR_FES_Timer: Delete Timer, TmrId = 250, TmrType = CALL WAIT INPUT INTERVAL.
// The timer specified the interval between key-presses under the Call node was deleted.
*Jun 27 12:05:12:115 2008 71_C IVRF/7/VOICE:
IVR_FES_Timer: Start Timer, TmrId = 385, TmrType = CALL WAIT INPUT INTERVAL, TmrLength = 10000.
// The timer specifies the interval between key-presses under the Call node was started.
*Jun 27 12:05:12:115 2008 71_C IVRF/7/VOICE:
IVR_FES_Info: CALL_WAIT_INPUT --> CALL_WAIT_INPUT, LocalId = 0.
// The state of the node FSM was CALL_WAIT_INPUT.
*Jun 27 12:05:12:714 2008 71_C IVRF/7/VOICE:
IVR_FES_Event: CMC --> FES : Accp Information.
// When the subscriber pressed key 0, the IVR_FES received an Accp Information message from CMC.
*Jun 27 12:05:12:714 2008 71_C IVRF/7/VOICE:
IVR_FES_Timer: Delete Timer, TmrId = 385, TmrType = CALL WAIT INPUT INTERVAL.
// The timer specified the interval between key-presses under the Call node was deleted.
*Jun 27 12:05:12:715 2008 71_C IVRF/7/VOICE:
IVR_FES_Timer: Start Timer, TmrId = 232, TmrType = CALL WAIT INPUT INTERVAL, TmrLength = 10000.
// The timer specifies the interval between key-presses under the Call node was started.
*Jun 27 12:05:12:715 2008 71_C IVRF/7/VOICE:
IVR_FES_Info: CALL_WAIT_INPUT --> CALL_WAIT_INPUT, LocalId = 0.
// The state of the node FSM was CALL_WAIT_INPUT.
*Jun 27 12:05:13:555 2008 71_C IVRF/7/VOICE:
IVR_FES_Event: CMC --> FES : Accp Information.
// When the subscriber pressed key #, the IVR_FES received an Accp Information message from CMC.
*Jun 27 12:05:13:555 2008 71_C IVRF/7/VOICE:
IVR_FES_Timer: Delete Timer, TmrId = 232, TmrType = CALL WAIT INPUT INTERVAL.
// The timer specified the interval between key-presses under the Call node was deleted.
*Jun 27 12:05:13:555 2008 71_C IVRF/7/VOICE:
IVR_FES_Info: CALL_WAIT_INPUT --> CALL, LocalId = 0.
// The state of the node FSM changed from CALL_WAIT_INPUT to CALL.
*Jun 27 12:05:13:555 2008 71_C IVRF/7/VOICE:
IVR_FES_Timer: Start Timer, TmrId = 317, TmrType = WAIT SERVICE ACK, TmrLength = 20000.

// The timer for receiving a Service Ack message was started.

*Jun 27 12:05:13:556 2008 71_C IVRF/7/VOICE:

// The IVR_FES sent an Accp Service message to the CMC.

*Jun 27 12:05:13:556 2008 71_C IVRF/7/VOICE:
IVR_FES_Fsm: ACTIVE --> WAIT_SERVICE_ACK, CallId = 720897, LocalId = 0.

// The state of the call FSM changed from ACTIVE to WAIT_SERVICE_ACK.

*Jun 27 12:05:13:558 2008 71_C IVRF/7/VOICE:
IVR_FES_Info: Receive message from SA module.

// The IVR_FES received a message from the SA module.

*Jun 27 12:05:13:559 2008 71_C IVRF/7/VOICE:
IVR_FES_Info: IDLE status process.

// The IVR system processed services in the IDLE state.

*Jun 27 12:05:13:560 2008 71_C IVRF/7/VOICE:
IVR_FES_Event: CMC --> FES : Accp Service Ack.

// The IVR_FES received an Accp Service Ack message from the CMC.

*Jun 27 12:05:13:560 2008 71_C IVRF/7/VOICE:
IVR_FES_Timer: Delete Timer, TmrId = 317, TmrType = WAIT SERVICE ACK.

// The timer for receiving a Service Ack message was deleted.

*Jun 27 12:05:13:561 2008 71_C IVRF/7/VOICE:
IVR_FES_Info: Service response status is ok, CallId = 720897.

// The voice service is executed successfully.

*Jun 27 12:05:13:561 2008 71_C IVRF/7/VOICE:

// The IVR_FES sent an Accp Release message to the CMC.

*Jun 27 12:05:13:561 2008 71_C IVRF/7/VOICE:
IVR_FES_Timer: Start Timer, TmrId = 195, TmrType = WAIT RELEASE COMPLETE, TmrLength = 6000.

// The timer for receiving an Accp Release Complete message was started.

*Jun 27 12:05:13:561 2008 71_C IVRF/7/VOICE:
IVR_FES_Fsm: WAIT_SERVICE_ACK --> RELEASE, CallId = 720897, LocalId = 0.

// The state of the call FSM changed from WAIT_SERVICE_ACK to RELEASE.

*Jun 27 12:05:13:563 2008 71_C IVRF/7/VOICE:

// The IVR_FES received an Accp Release Complete message form the CMC.

*Jun 27 12:05:13:564 2008 71_C IVRF/7/VOICE:
IVR_FES_Timer: Delete Timer, TmrId = 195, TmrType = WAIT RELEASE COMPLETE.

// The timer for receiving an Accp Release Complete message was deleted.

*Jun 27 12:05:13:613 2008 71_C IVRF/7/VOICE:
IVR_FES_Info: Receive message from SA module.

// The IVR_FES received a message from the SA module.
debugging voice ivr mps

Use **debugging voice ivr mps** to enable MPS debugging in the IVR system.

Use **undo debugging voice ivr mps** to disable MPS debugging in the IVR system.

**Syntax**

```
debugging voice ivr mps { all | error | event | timer }
undo debugging voice ivr mps { all | error | event | timer }
```

**Default**

MPS debugging is disabled.

**Views**

User view

**Default command level**

2: System level

**Parameters**

- **all**: Specifies all types of debugging for the media play system (MPS).
- **error**: Specifies error debugging.
- **event**: Specifies event debugging.
- **timer**: Specifies timer debugging.

**Usage guidelines**

Table 187 describes output fields and messages for the **debugging voice ivr mps error** command.

**Table 187 Output from the debugging voice ivr mps error command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Failed to get media play ID, MediaPlayld = %d.</td>
<td>The IVR failed to obtain the idle media play ID; the obtained media play ID is xxx.</td>
</tr>
<tr>
<td>Failed to find PlayCB, IfIndex = %d.</td>
<td>The IVR failed to find the node in the play control block; the FES module call ID is xxx.</td>
</tr>
<tr>
<td>Failed to calculate packet size, CodecType = %d PacketPeriod = %d.</td>
<td>The IVR failed to calculate packet size: the codec type is x, and the packetization period is xxx.</td>
</tr>
<tr>
<td>Failed to get filename by media ID, CodecType = %d MediaID = %d.</td>
<td>The IVR failed to obtain the media resource file name by using the media resource ID: the media resource ID is xxx.</td>
</tr>
</tbody>
</table>

Table 188 describes output fields and messages for the **debugging voice ivr mps event** command.
Table 188 Output from the debugging voice ivr mps event command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Free read control block, CodecType = %d, FileName = %s.</td>
<td>The read control block was released: the codec type was x and the filename was xxx.</td>
</tr>
<tr>
<td>Free resource control block, CodecType = %d, FileName = %s.</td>
<td>The resource control block was released: the codec type was x and the filename was xxx.</td>
</tr>
</tbody>
</table>

Examples

# Enable all types of debugging for the MPS. The output similar to the following example was created when the following conditions exist:

- The root node of the IVR system is a Call node with a node ID of 10.
- The root node is configured to play a media file named callSecDial_g729.wav (media resource ID is 1000) and originate normal secondary calls by matching the terminator #.

```bash
<Sysname> debugging voice ivr mps all
Enable IVR_MPS all debugging functions
```

*Jun 27 14:18:04:144 2008 71_C IVRM/7/VOICE:
IVR_MPS_Event: Start playing media, MediaID = 1000, FESCallID = 0.

- Output similar to the following messages is generated when the subscriber at 500 attempts to dial the IVR access number 800.

  // The media file with the media resource ID 1000 started to play.
  (*Jun 27 14:18:04:144 2008 71_C IVRM/7/VOICE:
  IVR_MPS_Event: Create resource control block, CodecType = 0x0, FileName = cfa0:/wav/g729r8/callSecDial_g729.wav.
  // The resource control block was created: the codec type is set to 0 (the g729r8 codec), and the media file path is specified as cfa0:/wav/g729r8/callSecDial_g729.wav.
  *Jun 27 14:18:04:145 2008 71_C IVRM/7/VOICE:
  IVR_MPS_Event: Create read control block, CodecType = 0x0, FileName = cfa0:/wav/g729r8/callSecDial_g729.wav.
  // The read control block was created.
  *Jun 27 14:18:04:145 2008 71_C IVRM/7/VOICE:
  IVR_MPS_Timer: Create timer, TmrId = 982, TmrType = LOOP TIMER, TmrLength = 500.
  // The IVR MPS was preparing for the play.
  *Jun 27 14:18:04:145 2008 71_C IVRM/7/VOICE:
  IVR_MPS_Timer: Create timer, TmrId = 483, TmrType = LOOP TIMER, TmrLength = 10.
  // The IVR_MPS timer was created.
  *Jun 27 14:18:04:632 2008 71_C IVRM/7/VOICE:
  IVR_MPS_Timer: Delete timer, TmrId = 982, TmrType = LOOP TIMER, TmrLength = 500.
  // The IVR MPS was preparing for the play.
  *Jun 27 14:18:04:652 2008 71_C IVRM/7/VOICE:
  IVR_MPS_Timer: Expire timer No. = 483, MediaID = 1000, FESCallID = 0.
  // The system printed the timer information for the transmission of every 500 packets for each loop.
  *Jun 27 14:18:08:372 2008 71_C IVRM/7/VOICE:
  IVR_MPS_Event: Free read control block, CodecType = 0x0, FileName = cfa0:/wav/g729r8/callSecDial_g729.wav.

333
// The read control block was released.
*Jun 27 14:18:08:372 2008 71_C IVRM/7/VOICE:
IVR_MPS_Event: Free resource control block, CodecType = 0x0, FileName = cfa0:/wav/g729r8/callSecDial_g729.wav.
// The resource control block was released.

• Output similar to the following messages is generated when the subscriber (500) hangs up without originating a secondary call:
*Jun 27 14:20:42:677 2008 71_C IVRM/7/VOICE:
IVR_MPS_Event: End playing media, MediaID = 1000 FESCallID = 0.
// The playing of the play resource media 1000 was completed.
*Jun 27 14:20:42:678 2008 71_C IVRM/7/VOICE:
IVR_MPS_Event: Delete PlayCB, PlayID = 1, FESCallID = 0.
// The play control block was deleted.
*Jun 27 14:20:42:678 2008 71_C IVRM/7/VOICE:
IVR_MPS_Timer: Delete timer, TmrId = 223, TmrType = LOOP TIMER, TmrLength = 10.
// The IVR_MPS timer was deleted.

• Output similar to the following messages is generated when the subscriber at 500 dials 600# to originate a secondary call and the subscriber at 600 picks up the ringing phone:
*Jun 27 14:18:12:930 2008 71_C IVRM/7/VOICE:
IVR_MPS_Event: End playing media, MediaID = 1000, FESCallID = 0.
// The playing of the play resource media 1000 was completed.
*Jun 27 14:18:12:930 2008 71_C IVRM/7/VOICE:
IVR_MPS_Event: Delete PlayCB, PlayID = 1, FESCallID = 0.
// The play control block was deleted.
*Jun 27 14:18:12:931 2008 71_C IVRM/7/VOICE:
IVR_MPS_Timer: Delete timer, TmrId = 483, TmrType = LOOP TIMER, TmrLength = 10.
// The IVR_MPS timer was deleted.
The output description tables in this document only contain fields and messages that require an explanation.

**debugging cwmp all**

Use **debugging cwmp all** to enable all types of debugging for CWMP.

Use **undo debugging cwmp all** to disable all CWMP debugging.

**Syntax**

```
debugging cwmp all
undo debugging cwmp all
```

**Default**

CWMP debugging is disabled.

**Views**

**User view**

**Default command level**

1: Monitor level

**Usage guidelines**

This command might cause CWMP to generate a large amount of log messages. To ensure system performance, use this command only when necessary and disable the command when you complete debugging.

**Examples**

# Enable all types of CWMP debugging. When the ACS is power-cycled to send an unsolicited request to the CPE, output similar to the following example is generated:

```
<Sysname> terminal debugging
<Sysname> terminal monitor
<Sysname> debugging cwmp all
```

*Aug 10 17:57:38:219 2007 Sysname CWMP/7/information:
information:Add request to queue. type = 1, value = 0x126bea70.

*Aug 10 17:57:39:203 2007 Sysname CWMP/7/DebugInfo:
information:Connecting to HTTP client succeeded.

// The device connected to the HTTP client successfully.

*Aug 10 17:57:39:203 2007 Sysname CWMP/7/DebugInfo:
information:Manufacturer: H3C.

// The vendor name of the CWMP device was displayed.

*Aug 10 17:57:39:203 2007 Sysname CWMP/7/DebugInfo:
information:The OUI of manufacturer is 000FE2.

// The OUI of the CWMP device was displayed.

*Aug 10 17:57:39:203 2007 Sysname CWMP/7/DebugInfo:
information: Serial number: 210231A73F1034103445.
// The CWMP device serial ID was displayed.
* Aug 10 17:57:39:203 2007 Sysname CWMP/7/DebugInfo:
  information: Construct message. length = 1325, message = <soap:Envelope
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:urn:dslforum-org:cwmp-1-0"
    xmlns:soap="http://schemas.xmlSOAP.org/soap/envelope/"
    xmlns:encodingStyle="http://schemas.xmlSOAP.org/soap/encoding/">
    <soap:Header>
      <cwmp:ID soap:mustUnderstand="1">1</cwmp:ID>
    </soap:Header>
    <soap:Body>
      <cwmp:Inform>
        <DeviceId>
          <Manufacturer>H3C</Manufacturer>
          <OUI>000FE2</OUI>
          <ProductClass>device</ProductClass>
          <SerialNumber>210231A73F1034103445</SerialNumber>
        </DeviceId>
        <Event encodingStyle:arrayType="cwmp:EventStruct[1]">
          <EventStruct>
            <EventCode>0 BOOTSTRAP</EventCode>
            <CommandKey></CommandKey>
          </EventStruct>
          </Event>
        <MaxEnvelopes>1</MaxEnvelopes>
        <CurrentTime>2007-08-20T16:42:26</CurrentTime>
        <RetryCount>0</RetryCount>
      </cwmp:Inform>
    </soap:Body>
  </soap:Envelope>
// CWMP constructed a protocol packet.
* Aug 10 17:57:39:219 2007 Sysname CWMP/7/DebugInfo:
  information: HTTP header sent. return-value = 0.
// CWMP returned a value of 0, and sent the HTTP header successfully.
* Aug 10 17:57:39:438 2007 Sysname CWMP/7/DebugInfo:
  information: Message sent. body-length = 1325.
// CWMP sent the message body with the length of 1325.
* Aug 10 17:57:39:438 2007 Sysname CWMP/7/DebugInfo:
  information: Authentication is requested by HTTP client.
// The HTTP client sent an authentication request.
* Aug 10 17:57:39:438 2007 Sysname CWMP/7/DebugInfo:
  transmit: Resend message. message =
<cwmp:ID soap:mustUnderstand="1">1</cwmp:ID>
</soap:Body>
</cwmp:Inform>
</soap:Body>
</soap:Envelope>

// CWMP sent an inform request again.
*Aug 10 17:57:39:438 2007 Sysname CWMP/7/DebugInfo:
information:HTTP header sent. return-value = 0.

// CWMP sent the inform request successfully.
*Aug 10 17:57:39:438 2007 Sysname CWMP/7/DebugInfo:
information:Message sent. body-length = 1325.

// CWMP sent the message successfully.
*Aug 10 17:57:39:734 2007 Sysname CWMP/7/DebugInfo:
information:HTTP client error or closed.

// CWMP closed HTTP.
*Aug 10 17:57:44:203 2007 Sysname CWMP/7/DebugInfo:
information:Connecting to HTTP client succeeded .

// The device connected to the HTTP client successfully.
*Aug 10 17:57:44:203 2007 Sysname CWMP/7/DebugInfo:
information:HTTP header sent. return-value = 0.

// CWMP sent the HTTP header successfully.
*Aug 10 17:57:44:203 2007 Sysname CWMP/7/DebugInfo:
information:Message sent. body-length = 1325.
*Aug 10 17:57:44:344 2007 Sysname CWMP/7/DebugInfo:
information:Completed to receive HTTP header.
CWMP finished receiving the HTTP header.
*Aug 10 17:57:44:344 2007 Sysname CWMP/7/DebugInfo:
  information:Continue to receive HTTP message.

CWMP continued to receive the HTTP message body.
*Aug 10 17:57:44:344 2007 Sysname CWMP/7/DebugInfo:
  information:Message received. length = 564.

CWMP received a message. The message length is 564.
*Aug 10 17:57:44:344 2007 Sysname CWMP/7/DebugInfo:
  receive:Message received. message =
  <soap:Envelope
    xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/
    xmlns:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/
    xmlns:urn:dslforum-org:cwmp-1-0"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soap:Header>
      <cwmp:ID soap:mustUnderstand="1">1</cwmp:ID>
    </soap:Header>
    <soap:Body>
      <cwmp:InformResponse>
        <MaxEnvelopes>5</MaxEnvelopes>
      </cwmp:InformResponse>
    </soap:Body>
  </soap:Envelope>

CWMP received a message.
*Aug 10 17:57:44:344 2007 Sysname CWMP/7/DebugInfo:
  information:Message received. length = 0.

CWMP received a message. The message length is 0.
*Aug 10 17:57:44:344 2007 Sysname CWMP/7/DebugInfo:
  receive:Receiving HTTP message completed.

CWMP finished receiving the message.
*Aug 10 17:57:44:359 2007 Sysname CWMP/7/DebugInfo:
  information:Pushing label succeeded. full-label = <cwmp:ID>.

CWMP succeeded in pushing the label with the long label <cwmp:ID>.
*Aug 10 17:57:44:359 2007 Sysname CWMP/7/DebugInfo:
  information:Pushing label succeeded. full-label = .

CWMP succeeded in popping the label with the long label empty.
*Aug 10 17:57:44:359 2007 Sysname CWMP/7/DebugInfo:
  information:Parsing RPC method completed.

CWMP completed parsing RPC method.
*Aug 10 17:57:44:359 2007 Sysname CWMP/7/DebugInfo:
  information:Pushing label succeeded. full-label = <cwmp:InformResponse>.

CWMP succeeded in pushing the label with the long label <cwmp:InformResponse>.
*Aug 10 17:57:44:375 2007 Sysname CWMP/7/DebugInfo:
  information:Pushing label succeeded. full-label = <cwmp:InformResponse><MaxE
nvelopes>.
// CWMP succeeded in popping the label with the long label <cwmp:InformResponse><MaxEnvelopes>.
// CWMP succeeded in pushing the label with the long label <cwmp:InformResponse>.
// CWMP succeeded in popping the label with the long label empty.
// CWMP completed parsing RPC method.
// CWMP succeeded in releasing user data.
// CWMP sent the HTTP header successfully and returned a value of 0.

debugging cwmp error
Use debugging cwmp error to enable CWMP error debugging.
Use undo debugging cwmp error to disable CWMP error debugging.

Syntax
debugging cwmp error
undo debugging cwmp error

Default
CWMP error debugging is disabled.

Views
User view

Default command level
1: Monitor level

Examples
# Enable CWMP error debugging and CWMP.
<Sysname> terminal debugging
<Sysname> terminal monitor
<Sysname> debugging cwmp error
<Sysname> system-view
[Sysname] cwmp
[Sysname-cwmp] undo cwmp enable
[Sysname-cwmp] cwmp enable
*Aug 13 16:40:52:719 2007 Sysname CWMP/7/DebugInfo:
error: Failed to create CWMS queue.

// CWMP failed to create CWMS queue when CWMP was enabled.

// CWMP failed to initialize the CWMP task.

// CWMP failed to get queue length.

debugging cwmp information

Use `debugging cwmp information` to enable CWMP information debugging.
Use `undo debugging cwmp information` to disable CWMP information debugging.

Syntax

debugging cwmp information
undo debugging cwmp information

Default

CWMP information debugging is disabled.

Views

User view

Default command level

1: Monitor level

Examples

# Enable CWMP information debugging. When the CPE receives a device status query from the ACS, output similar to the following example is generated:
<Sysname> terminal debugging
<Sysname> terminal monitor
<Sysname> debugging cwmp information
<Sysname> system-view
[Sysname] cwmp
[Sysname~cwmp]
*Aug 13 15:56:56:797 2007 Sysname CWMP/7/DebugInfo: information: Failed to get full name by CMO.

// CWMP failed to get node name according to CMO.

debugging cwmp packet receive

Use `debugging cwmp packet receive` to enable CWMP received packet debugging.
Use `undo debugging cwmp packet receive` to disable CWMP received packet debugging.

Syntax

debugging cwmp packet receive
undo debugging cwmp packet receive
Default

CWMP received packet debugging is disabled.

Views

User view

Default command level

1: Monitor level

Usage guidelines

Table 189 describes output fields and messages for the debugging cwmp packet receive command.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>xmlns:soap=&quot;<a href="http://schemas.xmlsoap.org/soap/envelope/">http://schemas.xmlsoap.org/soap/envelope/</a>&quot;</td>
<td>• xmlns—Naming space in the envelope field. It is a fixed character string defined by the protocol.</td>
</tr>
<tr>
<td>xmlns:encodingStyle=&quot;<a href="http://schemas.xmlsoap.org/soap/encoding/">http://schemas.xmlsoap.org/soap/encoding/</a>&quot;</td>
<td>• cwmp:ID—Identifier in the Header field to uniquely identify an interaction process. A pair of request and response must have the same cwmp:ID.</td>
</tr>
<tr>
<td>xmlns:cwmp=&quot;urn:dslforum-org:cwmp-1-0&quot;</td>
<td>• Body—Contains the packet content of each RPC method supported by CWMP. The Body field in the example is the response of the Inform method.</td>
</tr>
<tr>
<td>xmlns:xsd=&quot;<a href="http://www.w3.org/2001/XMLSchema">http://www.w3.org/2001/XMLSchema</a>&quot;</td>
<td></td>
</tr>
<tr>
<td>xmlns:xsi=&quot;<a href="http://www.w3.org/2001/XMLSchema-instance%22%3E">http://www.w3.org/2001/XMLSchema-instance&quot;&gt;</a></td>
<td></td>
</tr>
<tr>
<td><a href="">soap:Header</a></td>
<td></td>
</tr>
<tr>
<td>&lt;cwmp:ID soap:mustUnderstand=&quot;1&quot;&gt;1&lt;/cwmp:ID&gt;</td>
<td></td>
</tr>
<tr>
<td>&lt;/soap:Header&gt;</td>
<td></td>
</tr>
<tr>
<td><a href="">soap:Body</a></td>
<td></td>
</tr>
<tr>
<td><a href="">cwmp:InformResponse</a></td>
<td></td>
</tr>
<tr>
<td>&lt;MaxEnvelopes&gt;5&lt;/MaxEnvelopes&gt;</td>
<td></td>
</tr>
<tr>
<td>&lt;/cwmp:InformResponse&gt;</td>
<td></td>
</tr>
<tr>
<td>&lt;/soap:Body&gt;</td>
<td></td>
</tr>
<tr>
<td>&lt;/soap:Envelope&gt;</td>
<td></td>
</tr>
</tbody>
</table>

Examples

# Enable CWMP received packet debugging. When the ACS requests the CPE to set the ACS URL, ACS username, and ACS password, output similar to the following example is generated:

<Sysname> terminal debugging
<Sysname> terminal monitor
<Sysname> debugging cwmp packet receive
<Sysname> system-view
[Sysname] cwmp
[Sysname-cwmp] cwmp enable

*Aug 13 16:05:06:562 2007 Sysname CWMP/7/DebugInfo: receive:Message received. message =
<soap:Envelope
debugging cwmp packet send

Use **debugging cwmp packet send** to enable CWMP sent packet debugging.
Use **undo debugging cwmp packet send** to disable CWMP sent packet debugging.

**Syntax**

```
debugging cwmp packet send
undo debugging cwmp packet send
```

**Default**

CWMP sent packet debugging is disabled.

**Views**

User view

**Default command level**

1: Monitor level

**Examples**

```
# Enable CWMP sent packet debugging. When CWMP is enabled, output similar to the following example is generated:
<Sysname> terminal debugging
<Sysname> terminal monitor
<Sysname> debugging cwmp packet send
<Sysname> system-view
[Sysname] cwmp
[Sysname-cwmp] cwmp enable
*Aug 30 14:09:58:997 2007 ar49 CWMP/7/DebugInfo:
send:The packet need not pagination, continue times = 1, get value length = 8.
// CWMP sent an RPC packet.
```
debugging soap all

Use `debugging soap all` to enable all types of SOAP debugging.

Use `undo debugging soap all` to disable all types of SOAP debugging.

Syntax

```
debugging soap all
undo debugging soap all
```

Default

SOAP debugging is disabled.

Views

User view

Default command level

1: Monitor level

Usage guidelines

This command might cause heavy traffic and affect system performance. To ensure system performance, use this command only when necessary, and disable the command when you complete debugging.

Examples

```
# Enable all SOAP debugging.
<Sysname> terminal debugging
<Sysname> terminal monitor
<Sysname> debugging soap all

# Disable CWMP. If CWMP is disabled successfully, output similar to the following example is generated:
<Sysname> system-view
[Sysname] cwmp
[Sysname-cwmp] undo cwmp enable
*Aug 10 17:57:24:188 2007 Sysname CWMP/7/DebugInfo:
    information:CWMP disabled.

# Enable CWMP. If CWMP fails to parse XML, output similar to the following example is generated:
[Sysname-cwmp] cwmp enable
*Aug 30 14:57:33:102 2007 ar49 SOAP/7/DebugInfo:
    error:Failed to parse Xml.
```

debugging soap error

Use `debugging soap error` to enable SOAP error debugging.

Use `undo debugging soap error` to disable SOAP error debugging.

Syntax

```
debugging soap error
undo debugging soap error
```
SOAP error debugging is disabled.

Views
User view

Default command level
1: Monitor level

Examples
# Enable SOAP error debugging. If CWMP failed to create an XML parser for a received HTTP message, output similar to the following example is generated:
<Sysname> terminal debugging
<Sysname> terminal monitor
<Sysname> debugging soap error
*Aug 13 16:26:21:531 2007 Sysname SOAP/7/DebugInfo:
error:Failed to create XML parser.

debugging soap information

Use debugging soap information to enable SOAP information debugging.
Use undo debugging soap information to disable SOAP CWMP information debugging.

Syntax
debugging soap information
undo debugging soap information

Default
SOAP information debugging is disabled.

Views
User view

Default command level
1: Monitor level

Examples
# Enable SOAP information debugging. If CWMP send correct SOAP messages after it is enabled, output similar to the following example is generated:
<Sysname> terminal debugging
<Sysname> terminal monitor
<Sysname> debugging soap information
*Aug 13 16:26:21:531 2007 Sysname SOAP/7/DebugInfo:
information:Parsing SOAP message completed.
// CWMP parsed the SOAP message correctly.
DCC debugging commands

The output description tables in this document only contain fields and messages that require an explanation.

debugging dialer

Use debugging dialer to enable a specific type of DCC (Dial Control Center) debugging. Use undo debugging dialer to disable a specific type of DCC debugging.

Syntax

```plaintext
debugging dialer { all | event | packet }
undo debugging dialer { all | event | packet }
```

Default

All types of DCC debugging are disabled.

Views

User view

Default command level

1: Monitor level

Parameters

all: All types of DCC debugging.
event: Debugging for DCC events.
packet: Debugging for DCC packets.

Usage guidelines

Table 190 describes output fields and messages for the debugging dialer event command.

Table 190 Output from the debugging dialer event command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VLink index</td>
<td>Virtual link index.</td>
</tr>
<tr>
<td>Link layer transfer user &quot; to DCC on interface-type interface-number</td>
<td>Link layer transferred the peer name to DCC. If the peer name is null, a null character is transferred.</td>
</tr>
<tr>
<td>peeraddr matching success on interface interface-type interface-number ,link UP</td>
<td>Succeeded in matching the peer IP address, and the link went up.</td>
</tr>
<tr>
<td>link-layer NCP negotiation is successful</td>
<td>NCP negotiation succeeded.</td>
</tr>
<tr>
<td>received ISDN_CONN_CFM</td>
<td>DCC received ISDN connection confirmation messages.</td>
</tr>
<tr>
<td>userId</td>
<td>ID of the confirmation. 0x9 indicates that the call has been established.</td>
</tr>
<tr>
<td>father interface</td>
<td>Parent interface.</td>
</tr>
</tbody>
</table>
Table 191 describes output fields and messages for the **debugging dialer packet** command.

### Table 191 Output from the debugging dialer packet command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>the packet is interesting</td>
<td>The packet conforms to the dial-up rules and can trigger the device to dial a number.</td>
</tr>
<tr>
<td>try to find routing to ip-address on interface interface-type interface-number</td>
<td>DCC tried to find a route to ip-address on interface interface-type interface-number.</td>
</tr>
<tr>
<td>the NextHop of the packet is not Broadcast</td>
<td>The next hop of the packet is not a broadcast address.</td>
</tr>
<tr>
<td>Vlink index</td>
<td>Virtual link index.</td>
</tr>
<tr>
<td>A Link is connecting by this dialer route, waiting this Link</td>
<td>Link is being established based on the found dialer route.</td>
</tr>
<tr>
<td>Vlink =&gt; interface is interface-type interface-number</td>
<td>The packet traveled to interface-type interface-number over the virtual link.</td>
</tr>
</tbody>
</table>

**Examples**

# Enable debugging for DCC events on Router A. Output similar to the following example is generated when you ping Router B from Router A under the following conditions:

- **Router A and Router B are connected in the form of CE1/PRI.**
- **Router A is configured as follows:**
  ```
  #
  dialer-rule 1 ip permit
  #
  controller E1 2/0
  pri-set
  #
  interface Serial2/0:15
  link-protocol ppp
  dialer enable-circular
  dialer-group 1
  dialer circular-group 0
  isdn protocol-mode network
  #
  interface Dialer0
  link-protocol ppp
  ip address 10.0.0.1 255.255.255.0
  dialer enable-circular
  dialer-group 1
  dialer number 888
  #
  ```

- **Router B is configured as follows:**
  ```
  #
  dialer-rule 1 ip permit
  #
  controller E1 2/0
  pri-set
  ```
interface Serial2/0:15
  link-protocol ppp
dialer enable-circular
dialer-group 1
dialer circular-group 0
#
interface Dialer0
  link-protocol ppp
  ip address 10.0.0.2 255.255.255.0
dialer enable-circular
dialer-group 1
#

<RouterA> debugging dialer event
<RouterA> terminal monitor
<RouterA> ping 10.0.0.2
  PING 10.0.0.2: 56 data bytes, press CTRL_C to break
*0.20464096 RouterA DCC/8/debug:DCC: Try to find a free channel to dial '888' on the interface Dialer0
// DCC tried to find a free channel on Dialer 0 interface to use it to call the number 888.
*0.20464096 RouterA DCC/8/debug:DCC: Dialing 888 on interface Serial2/0:15 of interface Dialer0
// The device called the number 888 on Serial2/0:15 interface.
*0.20464096 RouterA DCC/8/debug:DCC: not set the queue! discard this packet
// Because the first packet is used for establishing a link and no link is established so far, the packet is discarded.
*0.20464116 RouterA DCC/8/debug:DCC: Serial2/0:15 received ISDN_CONN_CFM, with userId 0x9.
// Serial2/0:15 interface received a ISDN Connection confirmation message with the ID 0x9, indicating that the call was successfully placed.
*0.20464116 RouterA DCC/8/debug:DCC: Serial2/0:15 find B channel with timeslot 0xa
// Serial2/0:15 interface (D channel) found a B channel with the time slot 0xa.
*0.20464116 RouterA DCC/8/debug:DCC: Interface Serial2/0:9 connected with peer!
// Serial2/0:9 interface was connected to the peer successfully.
*0.20464116 RouterA DCC/8/debug:DCC: Link layer tries to obtain its network-layer interface for interface Serial2/0:9
// The device tried to obtain a network layer interface for Serial2/0:9 interface.
*0.20467120 RouterA DCC/8/debug:DCC: Link layer transfer user '' to DCC on interface Serial2/0:9
// Peer user name was passed to DCC. (A null string is passed to DCC if the peer user name is null.)
*0.20467130 RouterA DCC/8/debug:DCC: Serial2/0:9's peer address is '10.0.0.2', checking...
// The IP address of Serial2/0:9 interface is 10.0.0.2.
*0.20467130 RouterA DCC/8/debug:DCC: peeraddr matching success on interface Serial2/0:9 ,link UP
// The peer IP address was matched successfully and the link went up.
RouterA DCC/8/debug:DCC: Serial2/0:9's link-layer NCP negotiation is successful!

// NCP negotiation was successful on Serial2/0:9 interface.

RouterA DCC/8/debug:DCC: Add VLink 9 OK, father interface = Dialer0

// Virtual link 9 was successfully created. The parent interface is Dialer0.

# Enable debugging for DCC packets.

<RouterA> debugging dialer packet
<RouterA> terminal monitor
<RouterA> ping 10.0.0.2

PING 10.0.0.2: 56 data bytes, press CTRL_C to break

RouterA DCC/8/debug:DCC: try to find routing to '10.0.0.2' on interface Dialer0

// The device tried to find a route to 10.0.0.2.

RouterA DCC/8/debug:DCC: the NextHop of the packet is not Broadcast

// The next hop of the packet is not a broadcast address.

RouterA DCC/8/debug:DCC: Vlink index is 0

// The virtual link index is 0.

RouterA DCC/8/debug:DCC: the packet is interesting.

// The packet is an interesting packet, which conforms to the dial-up rules and can trigger the device to dial a number.

RouterA DCC/8/debug:DCC: Found a dialer number

// A dialer number was found.

RouterA DCC/8/debug:DCC: A Link is connecting by this dialer route, waiting this Link

// A link was being established according to the dialer route.

RouterA DCC/8/debug:DCC: Vlink index is 10

// The virtual link index is 10.

RouterA DCC/8/debug:DCC: Vlink => interface is Serial2/0:10

// A packet reached Serial2/0:10 interface with the help of the virtual link.
DDoS protection debugging commands

The output description tables in this document only contain fields and messages that require an explanation.

**debugging ddos packet**

Use `debugging ddos packet` to enable DDoS protection packet debugging.

Use `undo debugging ddos packet` to disable DDoS protection packet debugging.

**Syntax**

```
debugging ddos packet
undo debugging ddos packet
```

**Default**

DDoS protection packet debugging is disabled.

**Views**

User view

**Default command level**

1: Monitor level

**Usage guidelines**

Table 192 describes output fields and messages for the `debugging ddos packet` command.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User-service: user-service-name</td>
<td>Name of the user service.</td>
</tr>
<tr>
<td>Src-ip: src-ip-address</td>
<td>Source IP address of the packet.</td>
</tr>
<tr>
<td>Dst_ip: dst-ip-address</td>
<td>Destination IP address of the packet.</td>
</tr>
<tr>
<td>Src-port: src-port-num</td>
<td>Source port number of the packet.</td>
</tr>
<tr>
<td>Dst-port: dst-port-num</td>
<td>Destination port number of the packet.</td>
</tr>
</tbody>
</table>

**Examples**

# Enable DDoS protection packet debugging on the Guard. Output similar to the following example is generated when IP address 1.1.1.1 is attacked under the following conditions:

1. Traffic filtering devices are added on the SecCenter.
2. On the SecCenter, a user service named example is defined to protect IP address 1.1.1.1 against DDoS attacks.

```
<Sysname> debugging ddos packet
*Feb 26 16:01:28:192 2009 Sysname DDOS/7/PACKET:
```
User-service: example
Src-ip: 114.89.14.154
Dst-ip: 1.1.1.1
Src-port: 1024
Dst-port: 1025
Protocol: 6

// The Guard received a TCP packet for the user service example.

debugging ddos error

Use `debugging ddos error` to enable DDoS protection error debugging.
Use `undo debugging ddos error` to disable DDoS protection error debugging.

**Syntax**

```
debugging ddos error
undo debugging ddos error
```

**Default**

DDoS protection error debugging is disabled.

**Views**

User view

**Default command level**

1: Monitor level

**Usage guidelines**

Table 193 describes output fields and messages for the `debugging ddos error` command.

**Table 193 Output from the debugging ddos error command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Failed to create a drop_stat entry (template-type/template-para) for configuration synchronization. | The DDoS protection module failed to create a statistics entry for discarded packets during configuration synchronization.  
*template-type*: Template type, protocol (other protocol packet template), udp, tcp, dns-udp, dns-tcp, fragments, icmp, and http.  
*template-para*: Template parameter.  
- For a TCP or UDP template, this parameter is used to generate a TCP or UDP template object and is in the range of 1 to 65535 (port number).  
- For a protocol template, this parameter is used to generate a protocol template object and is in the range of 1 to 255 (protocol number).  
- For other templates, this parameter is invalid because each template has the unique object. |
| Not enough resource to send netstream packet | NetStream packets cannot be sent because of insufficient resources. |
Examples

# Enable DDoS protection error debugging on the Guard. Output similar to the following example is
generated when the IP address 1.1.1.1 is attacked under the following conditions:

- On the SecCenter, traffic filtering devices are added, and a user service named **example** is defined
to protect IP address 1.1.1.1 against DDoS attacks.
- A DDoS Guard detection interface is specified on the Guard.
- The NetStream log function is enabled on the Detector.

```plaintext
<Sysname> debugging ddos error
*Feb 26 16:01:58:516 2009 Sysname DDOS/7/ERROR: Failed to create a policy.
  // The DDoS protection module failed to create a policy due to insufficient memory.
*Feb 26 16:01:58:516 2009 Sysname DDOS/7/ERROR: Failed to create an IP statistics entry.
  // The DDoS protection module failed to create an IP statistics entry due to insufficient memory.
*Feb 26 16:01:58:516 2009 Sysname DDOS/7/ERROR: Failed to create an attacker record.
  // The DDoS protection module failed to create an attack source record due to insufficient memory.
*Feb 26 16:01:58:516 2009 Sysname DDOS/7/ERROR: Not enough resource to send netstream packet.
  // The DDoS protection module failed to send NetStream packets due to insufficient memory.
```

**debugging ddos event**

Use **debugging ddos event** to enable DDoS protection event debugging.

Use **undo debugging ddos event** to disable DDoS protection event debugging.

**Syntax**

```plaintext
debugging ddos event
undo debugging ddos event
```

**Default**

DDoS protection event debugging is disabled.

**Views**

User view

**Default command level**

1: Monitor level

**Usage guidelines**

Table 194 describes output fields and messages for the **debugging ddos event** command.
Table 194 Output from the debugging ddos event command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet dropped by the filter of the user-service user-service-name.</td>
<td>The packet was discarded by the filter of the user service named user-service-name.</td>
</tr>
<tr>
<td></td>
<td>• 0—Filtering based on source IP address.</td>
</tr>
<tr>
<td></td>
<td>• 1—Filtering based on destination IP address.</td>
</tr>
<tr>
<td>address: Source or destination IP address of the packet (depends on the filter type).</td>
<td>address: Source or destination IP address of the packet (depends on the filter type).</td>
</tr>
<tr>
<td>Created policy policy-name (policy-key-string).</td>
<td>Policy policy-name was created. The policy name includes template-type, template-para, analysis-type, and analysis-object. The policy key word is policy-key-string.</td>
</tr>
</tbody>
</table>

Examples

# Enable DDoS protection event debugging on the Guard. Output similar to the following example is generated when IP address 1.1.1.1 is attacked under the following conditions:

- Traffic filtering devices are added on the SecCenter.
- On the SecCenter, a user service named example is defined to protect IP address 1.1.1.1 against DDoS attacks.

<Sysname> debugging ddos event
*Feb 26 16:01:58:516 2009 Sysname DDOS/7/EVENT:
Added a protected network (1.1.1.1/32/0) for the user-service example.

// A protected network was added for the user service example.
*Feb 26 16:01:58:516 2009 Sysname DDOS/7/EVENT:
Created an IP statistics entry (1.1.1.1) of the user-service example.
TCP ports: 3, UDP ports: 1, port sequence: 3.

// An IP statistics entry for 1.1.1.1 was created.
*Feb 26 16:01:58:516 2009 Sysname DDOS/7/EVENT:
Created policy tcp/0/*/dst-ip (1.1.1.1).

// A policy was created. The policy processes all TCP packets with destination IP address 1.1.1.1.
*Feb 26 16:01:58:516 2009 Sysname DDOS/7/EVENT:
Refreshed policy tcp/0/*/dst-ip (1.1.1.1) when creating session.
Related sessions: 0, new sessions: 0, current sessions: 0, current sequence: 1.

// The policy was refreshed during session creation.
*Feb 26 16:01:58:516 2009 Sysname DDOS/7/EVENT:
Packet dropped according to policy tcp/0/packet/dst-ip (1.1.1.1).

// A packet was discarded according to the policy. The policy shows the total number of TCP packets with destination IP address 1.1.1.1 per second.
*Feb 26 16:01:58:516 2009 Sysname DDOS/7/EVENT:
Deleted a session.
Initiator: 2.2.2.1(1024), 1.1.1.1(1025), 6.
Responder: 1.1.1.1(1025), 2.2.2.1(1024), 6.

// A session was deleted. It is a TCP session with the initiator 2.2.2.1 at port 1024 and the responder 1.1.1.1 at port 1025.
*Feb 26 16:01:58:516 2009 Sysname DDOS/7/EVENT:
Refreshed policy dst-ip(1.1.1.1) when deleting session.
Related sessions: 887, new sessions: 40, current sessions: 887, current sequence: 1.

// The policy was refreshed during session deletion.
*Feb 26 16:01:58:516 2009 Sysname DDOS/7/EVENT:
Aging an IP statistics entry (1.1.1.1).

// The IP statistics entry of 1.1.1.1 aged out.
*Feb 26 16:01:58:516 2009 Sysname DDOS/7/EVENT:
Aging policy of the user-service example.

// The policy of the user service example aged out.
debugging device

Use `debugging device` to enable device debugging.
Use `undo debugging device` to disable device debugging.

Syntax

```
debugging device
undo debugging device
```

Default

Device debugging is disabled.

Views

User View

Default command level

1: Monitor level

Examples

```
# Enable device debugging.
<Sysname> debugging device
```
DHCP debugging commands

The output description tables in this document only contain fields and messages that require an explanation.

debugging dhcp server

Use `debugging dhcp server` to enable DHCP server debugging.
Use `undo debugging dhcp server` to disable DHCP server debugging.

Syntax

```
debugging dhcp server { all | error | event | packet }
undo debugging dhcp server { all | error | event | packet }
```

Default

DHCP server debugging is disabled.

Views

User view

Default command level

1: Monitor level

Parameters

- `all`: Specifies all types of debugging for DHCP server.
- `error`: Specifies DHCP server error debugging.
- `event`: Specifies DHCP server event debugging.
- `packet`: Specifies DHCP server packet debugging.

Examples

```
# Enable all types of DHCP server debugging. When a DHCP client requests an IP address from the
# DHCP server, output similar to the following example is generated:
<Sysname> terminal debugging
<Sysname> debugging dhcp server all
*0.263828 Sysname DHCPS/7/DHCPS_EVENT:
    Checking for expired lease.
// The DHCP server was checking for expired leases.
*0.278312 Sysname DHCPS/7/DHCPS_EVENT:
    DHCPServer: Receive DHCPCDISCOVER from 00e0.fc14.1601-Vlan-interface2 via 22.0.0.1.
*0.278312 Sysname DHCPS/7/DHCPS_PACKET:
    Rx, interface Vlan-interface1
    Message type: request
        Hardware type: 1, Hardware address length: 6
        Hops: 1, Transaction ID: 4281385283
        Seconds: 0, Broadcast flag: 0
```

355
// The DHCP server received a DHCP-DISCOVER packet.
*0.278312 Sysname DHCPS/7/DHCPS_EVENT:
DHCPServer: Sending ICMP ECHOREQUEST to target IP: 22.0.0.1.

// The DHCP server sent an ICMP echo-request to check whether the IP address 22.0.0.1 was in use before assigning it to the client.
*0.278312 Sysname DHCPS/7/DHCPS_EVENT:
DHCPServer: Assign free lease from global pool.

// The DHCP server sent an ICMP echo-request to check whether the IP address 22.0.0.2 was in use
*0.278406 Sysname DHCPS/7/DHCPS_EVENT:
DHCPServer: Sending ICMP ECHOREQUEST to target IP: 22.0.0.2.

// The DHCP server did not receive an ICMP reply from 22.0.0.2 before the timer expired.
*0.279016 Sysname DHCPS/7/DHCPS_PACKET:
Tx, interface Vlan-interface1
   Message type: reply
   Hardware type: 1, Hardware address length: 6
   Hops: 0, Transaction ID: 4281385283
   Seconds: 0, Broadcast flag: 0
   Client IP address: 0.0.0.0 Your IP address: 22.0.0.2
   Server IP address: 0.0.0.0 Relay agent IP address: 22.0.0.1
   Client hardware address: 00e0-fc14-1601
   Server host name: Not Configured, Boot file name: Not Configured
   DHCP message type: DHCP Offer

// The DHCP server sent a DHCP-OFFER message that contains the IP address 22.0.0.2 to the DHCP client.
*0.279016 Sysname DHCPS/7/DHCPS_EVENT:
DhcpServer: Send DHCPOFFER to 00e0.fc14.1601-Vlan-interface2 Offer IP=> 22.0.0.2 via 22.0.0.1.

*0.279172 Sysname DHCPS/7/DHCPS_EVENT:
DHCPServer: Receive DHCPREQUEST from 00e0.fc14.1601-Vlan-interface2 via 22.0.0.1.
*0.279172 Sysname DHCPS/7/DHCPS_PACKET:
Rx, interface Vlan-interface1
Message type: request
  Hardware type: 1, Hardware address length: 6
  Hops: 1, Transaction ID: 2294688324
  Seconds: 0, Broadcast flag: 0
  Client IP address: 0.0.0.0    Your IP address: 0.0.0.0
  Server IP address: 0.0.0.0    Relay agent IP address: 22.0.0.1
  Client hardware address: 00e0-fc14-1601
  Server host name: Not Configured, Boot file name: Not Configured
DHCP message type: DHCP Request

// The DHCP server received a DHCP-REQUEST message.
*0.279172 Sysname DHCPS/7/DHCPS_EVENT:
  DHCPServer: Acknowledge the DHCPREQUEST message.
*0.279172 Sysname DHCPS/7/DHCPS_PACKET:
Tx, interface Vlan-interface1
Message type: reply
  Hardware type: 1, Hardware address length: 6
  Hops: 0, Transaction ID: 2294688324
  Seconds: 0, Broadcast flag: 0
  Client IP address: 0.0.0.0    Your IP address: 22.0.0.2
  Server IP address: 0.0.0.0    Relay agent IP address: 22.0.0.1
  Client hardware address: 00e0-fc14-1601
  Server host name: Not Configured, Boot file name: Not Configured
DHCP message type: DHCP Ack

*0.279172 Sysname DHCPS/7/DHCPS_EVENT:
  DhcpServer: Send DHCPACK to 00e0.fc14.1601-Vlan-interface2 Offer IP=> 22.0.0.2 via 22.0.0.1.

// The DHCP server sent a DHCP-ACK message.

debugging dhcp relay

Use debugging dhcp relay to enable DHCP relay agent debugging.
Use undo debugging dhcp relay to disable DHCP relay agent debugging.

Syntax

debugging dhcp relay { all | error | event | packet [ client mac mac-address ] }
undo debugging dhcp relay { all | error | event | packet [ client mac mac-address ] }

Default
DHCP relay agent debugging is disabled.

Views
User view

Default command level
1: Monitor level
Parameters

- **all**: Specifies all types of debugging for DHCP relay agent.
- **error**: Specifies DHCP relay agent error debugging.
- **event**: Specifies DHCP relay agent event debugging.
- **packet**: Specifies DHCP relay agent packet debugging.
- **client mac mac-address**: Specifies debugging for packets that the DHCP relay agent forwards for a specific DHCP client. The `mac-address` argument is the MAC address of the DHCP client, in the format of H-H-H.

Examples

# Enable all types of debugging for DHCP relay agent. When a DHCP client requests an IP address from a DHCP server through the DHCP relay agent, output similar to the following example is generated:

```plaintext
<Sysname> terminal debugging
<Sysname> debugging dhcp relay all
<Sysname>
*0.230094 Sysname DHCPR/7/DHCPR_DEBUG_EVENT:
  Begin to deal with DHCP Discover packet.
*0.230094 Sysname DHCPR/7/DHCPR_DEBUG_PKTRXTX:
  Rx, DHCP request packet, interface Vlan-interface2.
*0.230094 Sysname DHCPR/7/DHCPR_DEBUG_PACKET:
  From client to server (Server-group 0):
    Message type: request
    Hardware type: 1, Hardware address length: 6
    Hops: 1, Transaction ID: 4281385283
    Seconds: 0, Broadcast flag: 1
    Client IP address: 0.0.0.0    Your IP address: 0.0.0.0
    Server IP address: 0.0.0.0    Relay agent IP address: 22.0.0.1
    Client hardware address: 00e0-fc14-1601
    Server host name: Not Configured, Boot file name: Not Configured
    DHCP message type: DHCP Discover
*0.230094 Sysname DHCPR/7/DHCPR_DEBUG_RELAYPKT:
  Pkt Sent: send request interface Vlan-interface22, dest IP: 11.0.0.1, chaddr: 00e0.fc14.1601, server-group: 0
// The DHCP relay agent received a DHCP.DISCOVER message from the DHCP client. The DHCP relay agent forwarded it to the DHCP server at 11.0.0.1 in DHCP server group 0.
*0.230891 Sysname DHCPR/7/DHCPR_DEBUG_EVENT:
  Begin to deal with DHCP Offer packet.
*0.230891 Sysname DHCPR/7/DHCPR_DEBUG_PKTRXTX:
  Rx, DHCP reply packet, interface Vlan-interface22.
*0.230891 Sysname DHCPR/7/DHCPR_DEBUG_PACKET:
  From server to client (Server-group 0):
    Message type: reply
    Hardware type: 1, Hardware address length: 6
    Hops: 0, Transaction ID: 2294688324
    Seconds: 0, Broadcast flag: 1
    Client IP address: 0.0.0.0    Your IP address: 22.0.0.2
```
Server IP address: 0.0.0.0    Relay agent IP address: 22.0.0.1
Client hardware address: 00e0-fc14-1601
Server host name: Not Configured, Boot file name: Not Configured
DHCP message type: DHCP Offer

*0.230891 Sysname DHCPR/7/DHCPR_DEBUG_RELAYPKT:
Pkt Sent: send reply interface Vlan-interface22, dest IP: 255.255.255.255,
           chardAddr: 00e0.fc14.1601, server-group: 0

  // The DHCP relay agent received a DHCP-OFFER message and broadcast it.
*0.230969 Sysname DHCPR/7/DHCPR_DEBUG_EVENT:
  Begin to deal with DHCP Request packet.
*0.230969 Sysname DHCPR/7/DHCPR_DEBUG_PKTRXTX:
  Rx, DHCP request packet, interface Vlan-interface22.

*0.230969 Sysname DHCPR/7/DHCPR_DEBUG_PACKET:
  From client to server (Server-group 0):
  Message type: request
  Hardware type: 1, Hardware address length: 6
  Hops: 1, Transaction ID: 2294688324
  Seconds: 0, Broadcast flag: 1
  Client IP address: 0.0.0.0    Your IP address: 0.0.0.0
  Server IP address: 0.0.0.0    Relay agent IP address: 22.0.0.1
  Client hardware address: 00e0-fc14-1601
  Server host name: Not Configured, Boot file name: Not Configured
  DHCP message type: DHCP Request

*0.230969 Sysname DHCPR/7/DHCPR_DEBUG_RELAYPKT:
Pkt Sent: send request interface Vlan-interface22, dest IP: 11.0.0.1,
           chardAddr: 00e0.fc14.1601, server-group: 0

  // The DHCP relay agent received a DHCP-REQUEST message from the DHCP client and forwarded it to the DHCP server.
*0.231063 Sysname DHCPR/7/DHCPR_DEBUG_EVENT:
  Begin to deal with DHCP Ack packet.
*0.231063 Sysname DHCPR/7/DHCPR_DEBUG_PKTRTX:
  Rx, DHCP reply packet, interface Vlan-interface22.

*0.231063 Sysname DHCPR/7/DHCPR_DEBUG_PACKET:
  From server to client (Server-group 0):
  Message type: reply
  Hardware type: 1, Hardware address length: 6
  Hops: 0, Transaction ID: 2294688324
  Seconds: 0, Broadcast flag: 1
  Client IP address: 0.0.0.0    Your IP address: 22.0.0.2
  Server IP address: 0.0.0.0    Relay agent IP address: 22.0.0.1
  Client hardware address: 00e0-fc14-1601
  Server host name: Not Configured, Boot file name: Not Configured
  DHCP message type: DHCP Ack
debugging dhcp client

Use **debugging dhcp client** to enable DHCP/BOOTP client debugging.

Use **undo debugging dhcp client** to disable DHCP/BOOTP client debugging.

**Syntax**

```
debugging dhcp client { all | error | event | packet }
undo debugging dhcp client { all | error | event | packet }
```

**Default**

DHCP/BOOTP client debugging is disabled.

**Views**

User view

**Default command level**

1: Monitor level

**Parameters**

- **all**: Specifies all types of debugging for DHCP/BOOTP clients.
- **error**: Specifies error debugging or unknown packet debugging for DHCP/BOOTP clients.
- **event**: Specifies DHCP/BOOTP client event debugging.
- **packet**: Specifies DHCP/BOOTP client packet debugging.

**Examples**

# Enable all types of debugging for DHCP clients. When a DHCP client requests an IP address from a DHCP server, output similar to the following example is generated:

```
<Sysname> debugging dhcp client all
<Sysname> terminal debugging
<Sysname> system
[Sysname] interface vlan-interface 2
[Sysname-Vlan-interface2] ip address dhcp-alloc
[Sysname-Vlan-interface2]
*0.105343 Sysname DHCPC/7/DHCPC_EVENT:
Vlan-interface2: Succeeded in enabling DHCP.
// DHCP is enabled on VLAN-interface 2 (the DHCP client).
*0.105359 Sysname DHCPC/7/DHCPC_EVENT:
Vlan-interface2: Move to INIT state.
*0.105359 Sysname DHCPC/7/DHCPC_EVENT:
Vlan-interface2: FSM state transfers (HALT-->INIT) successfully.
// The state of the DHCP client changed from HALT to INIT.
```
*0.105359 Sysname DHCPC/7/DHCPC_EVENT:
Vlan-interface2: DHCPDISCOVER will be sent in 5 ms.

*0.110343 Sysname DHCPC/7/DHCPC_PACKET:
Vlan-interface2: Send a DHCP packet...
    Head: op(1); htype(ETHERNET); hlen(6); xid(0x43c130ff);
    ciaddr(0.0.0.0); yiaddr(0.0.0.0); chaddr(00e0-fc14-1601);
    Options:
        63 82 53 63 35 01 01 0C 06 63 6C 69 65 74 37
        04 01 03 06 0F 39 02 04 80 3C 10 00 00 00 00 20
        B8 C4 B1 E4 B5 B1 C7 B0 C2 B7 BE 3D 20 00 30 30
        65 30 2E 66 63 31 34 3E 31 36 30 31 2D 56 6C 61
        6E 2D 69 6E 74 65 72 66 61 63 65 32 32 FF

*0.110343 Sysname DHCPC/7/DHCPC_EVENT:
Vlan-interface2: Sending DHCPDISCOVER packet succeeded.

*0.110343 Sysname DHCPC/7/DHCPC_EVENT:
Vlan-interface2: FSM state transfers (INIT-->SELECTING) successfully.

    // The DHCP client sent a DHCP-DISCOVER message, and the state of the DHCP client changed from
    INIT to SELECTING.

*0.111218 Sysname DHCPC/7/DHCPC_EVENT:
Vlan-interface2: Receive a packet.

*0.111218 Sysname DHCPC/7/DHCPC_PACKET:
Vlan-interface2: Receive a DHCP packet...
    Head: op(BOOTPREPLY); htype(ETHERNET); hlen(6); xid(0x43c130ff);
    ciaddr(0.0.0.0); yiaddr(22.0.0.2); chaddr(00e0-fc14-1601);
    Option: type(DHCPOFFER); mask(255.255.255.0); lease(86400);
    T1(43200); T2(75600); server(11.0.0.1);

*0.111218 Sysname DHCPC/7/DHCPC_EVENT:
Vlan-interface2: Select 11.0.0.1 as the server.

    // The DHCP client received a DHCP-OFFER message from the DHCP server at 11.0.0.1. The assigned IP
    address is 22.0.0.2, and the lease period is 86400 seconds (1 day).

*0.111218 Sysname DHCPC/7/DHCPC_PACKET:
Vlan-interface2: Send a DHCP packet...
    Head: op(1); htype(ETHERNET); hlen(6); xid(0x442ac688);
    ciaddr(0.0.0.0); yiaddr(0.0.0.0); chaddr(00e0-fc14-1601);
    Options:
        63 82 53 63 35 01 03 0C 06 63 6C 69 65 74 32
        04 16 00 00 02 36 04 0B 00 00 01 37 04 01 03 06
        0F 39 02 04 80 3C 10 00 00 00 00 44 65 63 69 65
        4E 6E 76 00 4F 63 74 00 3D 20 00 30 30 65 30 30
        2E 66 63 31 34 2E 31 36 30 31 2D 56 6C 61 6E
        6E 74 65 72 66 61 63 65 32 32 FF

*0.111218 Sysname DHCPC/7/DHCPC_EVENT:
Vlan-interface2: Sending DHCPREQUEST packet succeeded.

*0.111218 Sysname DHCPC/7/DHCPC_EVENT:
Vlan-interface2: FSM state transfers (SELECTING-->REQUESTING) successfully.
// The DHCP client sent a DHCP-REQUEST message, and the state of the DHCP client changed from
SELECTING to REQUESTING.

*0.111421 Sysname DHCPC/7/DHCPC_EVENT:
Vlan-interface2: Receive a packet.

*0.111421 Sysname DHCPC/7/DHCPC_PACKET:
Vlan-interface2: Receive a DHCP packet...
    Head: op(BOOTPREPLY); htype(ETHERNET); hlen(6); xid(0x442ac688);
    ciaddr(0.0.0.0); yiaddr(22.0.0.2); chaddr(00e0-fc14-1601);
    Option: type(DHCPACK); mask(255.255.255.0); lease(86400);
    T1(43200); T2(75600); server(11.0.0.1);

*0.111421 Sysname DHCPC/7/DHCPC_EVENT:
Vlan-interface2: FSM state transfers (REQUESTING-->BOUND) successfully.

// The DHCP client received a DHCP-ACK message, and the state of the DHCP client changed from
REQUESTING to BOUND.

*0.111421 Sysname DHCPC/7/DHCPC_EVENT:
Vlan-interface2: Begin to detect IP address conflict via ARP.

// The DHCP client started address conflict detection by using ARP.

*0.111421 Sysname DHCPC/7/DHCPC_EVENT:
Vlan-interface2: Sending ARP request for address(22.0.0.2) succeeded.

// The DHCP client sent an ARP request.

*0.111421 Sysname DHCPC/7/DHCPC_EVENT:
Vlan-interface2: Move to BOUND state if no ARP reply is received in 1000 milliseconds.

// If the DHCP client does not receive any ARP reply within 1 second, the state of the DHCP client will
change to BOUND.

*0.112375 Sysname DHCPC/7/DHCPC_EVENT:
Vlan-interface2: Receive no ARP reply for 22.0.0.2, so begin to use the address.

// The DHCP client did not receive an ARP reply, the state of the DHCP client changed to BOUND, and
the DHCP client can use the IP address.

debugging dhcp-snooping

Use debugging dhcp-snooping to enable DHCP snooping debugging.
Use undo debugging dhcp-snooping to disable DHCP snooping debugging.

Syntax

debugging dhcp-snooping { all | error | event | information | packet }
undo debugging dhcp-snooping { all | error | event | information | packet }

Default

DHCP snooping debugging is disabled.

Views

User view

Default command level

1: Monitor level
Parameters

- **all**: Specifies all types of DHCP snooping debugging.
- **error**: Specifies DHCP snooping error debugging.
- **event**: Specifies DHCP snooping event debugging.
- **information**: Specifies debugging for DHCP snooping Option 82.
- **packet**: Specifies DHCP snooping packet debugging.

Usage guidelines

Table 195 describes output fields and messages for the **debugging dhcp-snooping error** command.

**Table 195 Output from the debugging dhcp-snooping error command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>valid security items</td>
<td>Valid DHCP snooping entries. The entries are obtained from DHCP ACK messages and need to be synchronized to all cards.</td>
</tr>
<tr>
<td>DHCP snooping security item process:</td>
<td>DHCP snooping processed a DHCP snooping entry according to the information in the received DHCP packet and the local DHCP snooping entries.</td>
</tr>
</tbody>
</table>

Table 196 describes output fields and messages for the **debugging dhcp-snooping event** command.

**Table 196 Output from the debugging dhcp-snooping event command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default router: router-ip</td>
<td>Default gateway.</td>
</tr>
<tr>
<td>Client IP address: client-ip</td>
<td>IP address of the DHCP client.</td>
</tr>
<tr>
<td>Your IP address: your-ip</td>
<td>IP address that the DHCP server assigned to the client.</td>
</tr>
<tr>
<td>Transaction ID: transaction-id</td>
<td>A random number chosen by the DHCP client to uniquely identify an IP address allocation.</td>
</tr>
<tr>
<td>The table is full.</td>
<td>DHCP snooping failed to add a DHCP snooping entry because the DHCP snooping table is full.</td>
</tr>
</tbody>
</table>
### Field

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DHCP snooping processed DHCP snooping entries.</td>
</tr>
<tr>
<td>DHCP snooping entry type:</td>
</tr>
<tr>
<td>• 0—No entry.</td>
</tr>
<tr>
<td>• 1—Static entry.</td>
</tr>
<tr>
<td>• 2—Temporary dynamic entry.</td>
</tr>
<tr>
<td>• 3—Valid dynamic entry.</td>
</tr>
<tr>
<td>• 7—Dynamic entry, including 2 and 3.</td>
</tr>
<tr>
<td>• 8—Valid entry, including 1 and 3.</td>
</tr>
<tr>
<td>• 9—All entries.</td>
</tr>
<tr>
<td>DHCP snooping operation type:</td>
</tr>
<tr>
<td>• 0—Create.</td>
</tr>
<tr>
<td>• 1—Destroy.</td>
</tr>
<tr>
<td>• 2—Add.</td>
</tr>
<tr>
<td>• 3—Delete.</td>
</tr>
<tr>
<td>• 4—Update.</td>
</tr>
<tr>
<td>• 5—Age.</td>
</tr>
<tr>
<td>• 6—Delete based on type.</td>
</tr>
<tr>
<td>• 7—Delete based on interface.</td>
</tr>
<tr>
<td>• 8—Deleted based on IP address.</td>
</tr>
<tr>
<td>• 9—Display.</td>
</tr>
</tbody>
</table>

### Examples

# Enable all types of debugging for DHCP snooping. When a DHCP client requests an IP address through the DHCP snooping device, output similar to the following example is generated:

```
<Sysname> debugging dhcp-snooping all
<Sysname> terminal debugging
<Sysname>
*Sep 10 02:15:54:913 2007 SNOOP DHCPSP/7/PACKET:
  DHCP snooping begins to process packets on the main board.
// DHCP snooping is processing packets.
*Sep 10 02:15:54:915 2007 SNOOP DHCPSP/7/PACKET:
  The VLAN ID is changed for received packet (Old VLAN ID: 1, New VLAN ID: 1, S-VLAN ID: 1).
*Sep 10 02:15:54:965 2007 SNOOP DHCPSP/7/PACKET:
  Succeeded in getting DHCP packet type (1).
// DHCP snooping obtained a type 1 DHCP packet (DHCP DISCOVER packet).
*Sep 10 02:15:55:734 2007 Sysname DHCPSP/7/EVENT:
  Receive a DHCP DISCOVER packet.
// DHCP snooping received a DHCP-DISCOVER message.
*Sep 10 02:02:54:585 2007 SNOOP DHCPSP/7/PACKET:
  Send DHCP snooping packets to ethernet port (length:328).
*Sep 10 02:02:54:725 2007 SNOOP DHCPSP/7/PACKET:
  Succeeded in sending DHCP snooping packet to trusted port Ethernet1/1 in VLAN 1.
// DHCP snooping forwarded the DHCP-DISCOVER message through trusted port Ethernet 1/1.
```
*Sep 10 02:23:17:75 2007 SNOOP DHCPSP/7/PACKET:
Received a DHCP OFFER packet.

// DHCP snooping received a DHCP-OFFER message.
*Sep 10 02:23:17:219 2007 Sysname DHCPSP/7/PACKET:
Send DHCP snooping packet to VLAN (length: 350).
*Sep 10 02:23:17:219 2007 Sysname DHCPSP/7/PACKET:
Succeeded in sending a packet to all ports in vlan 1.

// DHCP snooping forwarded the DHCP-OFFER message to all ports in VLAN 1.
*Sep 10 02:23:22:078 2007 SNOOP DHCPSP/7/PACKET:
Received a DHCP REQUEST packet.
*Sep 10 02:23:22:078 2007 SNOOP DHCPSP/7/PACKET:
Requested IP address: 1.1.1.2
*Sep 10 02:23:22:078 2007 SNOOP DHCPSP/7/PACKET:
Default router: 0.0.0.0

// DHCP snooping received a DHCP-REQUEST message, and part of the packet fields was printed.
*Sep 10 02:23:22:281 2007 Sysname DHCPSP/7/EVENT:
Client hardware address: 00e0-fc00-1601
*Sep 10 02:23:22:281 2007 Sysname DHCPSP/7/EVENT:
Transaction ID: 387764469

// DHCP snooping is processing the DHCP snooping entry.
*Sep 10 02:23:22:297 2007 Sysname DHCPSP/7/EVENT:
Search for the first item with the IP address 1.1.1.2.

// DHCP snooping searched for a DHCP snooping entry that contains IP address 1.1.1.2.
*Sep 10 02:23:22:297 2007 Sysname DHCPSP/7/EVENT:
DHCP snooping security item table is null.

// DHCP snooping found no matching entry.
*Sep 10 02:23:22:297 2007 Sysname DHCPSP/7/EVENT:
Type of the incoming security item is 2.

// DHCP snooping assigned type 2 to the new DHCP snooping entry (temporary dynamic entry).
*Sep 10 02:23:22:312 2007 Sysname DHCPSP/7/EVENT:
DHCP snooping security item process:
A DHCP-Request packet received, the corresponding item does not exist.

// DHCP snooping found no DHCP snooping entry for IP address 1.1.1.2.
*Sep 10 02:23:22:312 2007 Sysname DHCPSP/7/EVENT:
Add item, IP is 1.1.1.2.

// DHCP snooping added the DHCP snooping entry.
*Sep 10 02:23:22:312 2007 Sysname DHCPSP/7/EVENT:
Add a DHCP snooping security item(ip address: 1.1.1.2).
*Sep 10 02:23:22:312 2007 Sysname DHCPSP/7/EVENT:
Updating DHCP snooping security item statistics, type: 2, op: 2.

// DHCP snooping was updating the DHCP snooping entry statistics: the type is 2 (temporary dynamic entry), and the operation type is 2 (add).
*Sep 10 02:23:22:312 2007 Sysname DHCPSP/7/EVENT:
Send DHCP snooping packet to ethernet port (length: 350).

*Sep 10 02:23:22:312 2007 Sysname DHCPSP/7/EVENT:
Succeed in sending DHCP snooping packet to trusted port Ethernet1/1 in VLAN 1

// DHCP snooping forwarded the packet through trusted port Ethernet 1/1.

*Sep 10 02:23:27:625 2007 SNOOP DHCPSP/7/PACKET:
DHCP snooping begins to process packets on the main board.

// DHCP snooping is processing the packet.

*Sep 10 02:23:27:735 2007 SNOOP DHCPSP/7/PACKET:
The VLAN ID is changed for received packet (Old VLAN ID: 1, New VLAN ID: 1, S-VLAN ID: 0).

*Sep 10 02:23:28:334 2007 SNOOP DHCPSP/7/PACKET:
Succeeded in getting DHCP packet type (2).

// DHCP snooping obtained a type 2 DHCP packet (DHCP OFFER packet).

*Sep 10 02:23:28:915 2007 SNOOP DHCPSP/7/PACKET:
Received a DHCP OFFER packet.

*Sep 10 02:23:29:328 2007 Sysname DHCPSP/7/EVENT:
Your IP: 1.1.1.2

*Sep 10 02:23:29:328 2007 Sysname DHCPSP/7/EVENT:
Lease time: 0x3c000000

*Sep 10 02:23:29:328 2007 Sysname DHCPSP/7/EVENT:
Client hardware address: 00e0-fc00-1601

*Sep 10 02:23:29:328 2007 Sysname DHCPSP/7/EVENT:
Transaction ID: 387764469

*Sep 10 02:23:29:328 2007 Sysname DHCPSP/7/EVENT:
Default router: 1.1.1.1

// DHCP snooping received a DHCP-OFFER packet, and part of the packet fields was printed.

*Sep 10 02:23:29:328 2007 Sysname DHCPSP/7/EVENT:
Processing DHCP snooping security item.

// DHCP snooping was processing the DHCP snooping entry.

*Sep 10 02:23:29:328 2007 Sysname DHCPSP/7/EVENT:
Search for the first item with the IP address 1.1.1.2.

*Sep 10 02:23:29:328 2007 Sysname DHCPSP/7/EVENT:
Succeeded in finding the DHCP snooping security item (IP address: 1.1.1.2, type: 2).

// DHCP snooping found the DHCP snooping entry of type 2 (temporary dynamic entry) corresponding
to IP address 1.1.1.2.

*Sep 10 02:23:29:391 2007 Sysname DHCPSP/7/EVENT:
Type of the incoming security item is 3.

*Sep 10 02:23:29:391 2007 Sysname DHCPSP/7/EVENT:
DHCP snooping security item process:
A DHCP-Ack packet received, and the type of the corresponding item is Request.

*Sep 10 02:23:29:391 2007 Sysname DHCPSP/7/EVENT:
Add item, IP is 1.1.1.2.

*Sep 10 02:23:29:391 2007 Sysname DHCPSP/7/EVENT:
The same MAC address is found, so the items will be updated.

// DHCP snooping updated the DHCP snooping entry.
*Sep 10 02:23:29:391 2007 Sysname DHCPSP/7/EVENT:
Updating DHCP snooping security item statistics, type: 3, op: 2.

// DHCP snooping was updating the DHCP snooping entry statistics: the entry type is 3 (valid dynamic entry), and the operation type is 2 (add).

*Sep 10 02:23:29:391 2007 Sysname DHCPSP/7/ERROR:
Failed to request IPCIM to add a DHCP snooping security item (1.1.1.2 00e0-fc00-1601).

// DHCP snooping failed to notify the user information management module to add the DHCP snooping entry.

*Sep 10 02:23:29:391 2007 Sysname DHCPSP/7/EVENT:
Updating DHCP snooping security item statistics, type: 2, op: 3.

// DHCP snooping was updating the DHCP snooping entry statistics: the entry type is 2 (temporary dynamic entry), and the operation type is 3 (delete).

*Sep 10 02:23:29:391 2007 Sysname DHCPSP/7/EVENT:
Send DHCP snooping packet to VLAN (length: 350).

*Sep 10 02:23:29:406 2007 Sysname DHCPSP/7/EVENT:
Succeeded in sending a packet to all ports in vlan 1.

// DHCP snooping forwarded the DHCP packet to all ports in VLAN 1.
DHCPv6 debugging commands

The output description tables in this document only contain fields and messages that require an explanation.

debugging ipv6 dhcp client

Use `debugging ipv6 dhcp client` to enable DHCPv6 client debugging.
Use `undo debugging ipv6 dhcp client` to disable DHCPv6 client debugging.

Syntax

```
debugging ipv6 dhcp client { all | error | event | packet | state }
undo debugging ipv6 dhcp client { all | error | event | packet | state }
```

Default

DHCPv6 client debugging is disabled.

Views

User view

Default command level

1: Monitor level

Parameters

- `all`: Specifies all DHCPv6 client debugging.
- `error`: Specifies DHCPv6 client error debugging.
- `event`: Specifies DHCPv6 client event debugging.
- `packet`: Specifies DHCPv6 client packet debugging.
- `state`: Specifies DHCPv6 client state debugging.

Usage guidelines

Disabling DHCPv6 client debugging globally also disables DHCPv6 client debugging on interfaces.

Examples

# Enable DHCPv6 client state debugging. When DHCPv6 client stateless autoconfiguration is enabled on VLAN-interface 2, output similar to the following example is generated:
```
<Sysname> debugging ipv6 dhcp client state
*Nov 23 16:15:39:118 2006 Sysname DHCPC6/7/DebugState:
Interface Vlan-interface2, stateless client:
    INIT --> IDLE (O-flag received)
```

// The DHCPv6 client received an O-Flag change notification, and its state changed from INIT to IDLE.

# Enable DHCPv6 client packet debugging. When DHCPv6 client stateless autoconfiguration is enabled on VLAN-interface 2, output similar to the following example is generated:
```
<Sysname> debugging ipv6 dhcp client packet
*Dec 12 09:52:12:990 2006 Sysname DHCPC6/7/DebugPacket:
```

368
Vlan-interface2, Packet Sent:
Type Information-request(11)
Transaction-id 0x07e0d3
*Dec 12 09:52:12:990 2006 Sysname DHCPC6/7/PacketOption:
Option               Length  Information
CLIENTID(1)          10      DUID type: Link-layer address(3)
                     Hardware type: 1
                     Link-layer address: 0x000fea23de30
ORO(6)               4       DNS-SERVERS(23)
                     DOMAIN-LIST(24)
ELAPSED-TIME(8)      2       0 centisecond(s)

// The DHCPv6 client sent an Information-request message.
*Dec 12 10:00:45:696 2006 Sysname DHCPC6/7/DebugPacket:
Vlan-interface2, Packet Received:
Type Reply(7)
Transaction-id 0x07e0d3
*Dec 12 10:00:45:712 2006 Sysname DHCPC6/7/PacketOption:
Option               Length  Information
CLIENTID(1)          10      DUID type: Link-layer address(3)
                     Hardware type: 1
                     Link-layer address: 0x000fea23de30
SERVERID(2)          10      DUID type: Link-layer address(3)
                     Hardware type: 1
                     Link-layer address: 0x00137ff6c818
DNS-SERVERS(23)      16      1:2:3::5
DOMAIN-LIST(24)      9       abc.com

// The DHCPv6 client received a Reply message.
# Enable DHCPv6 client stateless autoconfiguration on VLAN-interface 2 and enable DHCPv6 client event debugging.
 SYSname> debugging ipv6 dhcp client event
*Nov 20 17:37:26:502 2006 Sysname DHCPC6/7/DebugEvent:
Timer Event: Timer for sending initial packet is created.

// The timer for sending DHCPv6 client packets was created.
# Enable DHCPv6 client error debugging. When DHCPv6 client stateless autoconfiguration is enabled on VLAN-interface 2, output similar to the following example is generated:
 SYSname> debugging ipv6 dhcp client error
*Nov 25 09:05:19:102 2006 Sysname DHCPC6/7/DebugError:
Packet Error: Wrong packet header length, on Vlan-interface2.

// The DHCPv6 client received a packet with a wrong header length.
# Enable all DHCPv6 client debugging. Output similar to the following example is generated when Ethernet 1/1 requests an IP address under the following conditions:
 SYSname> debugging ipv6 dhcp client all
*Aug  9 14:37:40:312 2011 Sysname DHCPC6/7/DebugEvent:
Timer Event: Timer for sending packets is destroyed.
The timer for sending packets was removed.

*Aug  9 14:37:40:312 2011 Sysname DHCPC6/7/DebugEvent:
Try to add link-local address on interface Ethernet1/1.

The DHCPv6 client tried to add a link-local address on the interface Ethernet 1/1.

*Aug  9 14:37:40:312 2011 Sysname DHCPC6/7/DebugEvent:
Timer Event: Timer for sending the initial packet expired.

The timer for sending the initial packets expired.

*Aug  9 14:37:40:312 2011 Sysname DHCPC6/7/DebugEvent:
Timer Event: Timer for sending packets is created, try after 1076 ms.

The timer for sending packets was created, and packets were sent in 1076 milliseconds.

*Aug  9 14:37:40:312 2011 Sysname DHCPC6/7/DebugEvent:
Sent Solicit(1) packet from interface Ethernet1/1.

A Solicit message was sent.

*Aug  9 14:37:40:312 2011 Sysname DHCPC6/7/DebugPacket:
Ethernet1/1, Packet Sent:
Type Solicit(1)
Transaction-ID 0xd60e00

*Aug  9 14:37:40:312 2011 Sysname DHCPC6/7/PacketOption:
Option               Length    Information
RAPID_COMMIT(14)     0
CLIENTID(1)          10        DUID type: Link-layer address(3)
                        Hardware type: 1
                        Link-layer address: 0x000fe20a4d00
IA_NA(3)             40        IAID: 0xf0019
                        T1: 0
                        T2: 0
IAADDR(5)           24        Address: ::
                        Preferred lifetime: 0
                        Valid lifetime: 0
ORO(6)               4
                        DNS_SERVERS(23)
                        DOMAIN_LIST(24)
ELAPSED_TIME(8)      2        0 centisecond(s)

Information about the Solicit message was displayed.

*Aug  9 14:37:40:312 2011 Sysname DHCPC6/7/DebugState:
Interface Ethernet1/1, stateful client requested for address:
    INIT --> SOLICIT (Enable client function)

The client state changed from INIT to SOLICIT.

*Aug  9 14:37:40:468 2011 Sysname DHCPC6/7/DebugPacket:
Ethernet1/1, Packet Received:
Type Reply(7)
Transaction-ID 0xd60e00

*Aug  9 14:37:40:468 2011 Sysname DHCPC6/7/PacketOption:
Option               Length    Information
RAPID_COMMIT(14)     0
CLIENTID(1)          10        DUID type: Link-layer address(3)
                        Hardware type: 1
SERVERID(2) 14 DUID type: Link-layer address plus time(1)
Hardware type: 6
Time: 328711988 seconds
Link-layer address: 0x001143bbb78c

IA_NA(3) 74 IAID: 0xf0019
T1: 300
T2: 400

IAADDR(5) 24 Address: 100::9DD8:D090:A1A6:7858
Preferred lifetime: 500
Valid lifetime: 600

STATUS_CODE(13) 30 status-code: Success(0)
All addresses have been assigned.

DNS_SERVERS(23) 32 2000::FF
2000::FE

DOMAIN_LIST(24) 32 example.com
example2.test.com

// A reply packet was received. Information about the reply packet was displayed.
*Aug  9 14:37:40:468 2011 Sysname DHCPC6/7/DebugEvent:
  Received Reply(7) packet from interface Ethernet1/1.

// The received packet was a Reply message.
*Aug  9 14:37:40:468 2011 Sysname DHCPC6/7/DebugEvent:
  Deleted network parameters on interface Ethernet1/1.

// Network parameters on interface Ethernet 1/1 were deleted.
*Aug  9 14:37:40:468 2011 Sysname DHCPC6/7/DebugEvent:
  Added address 100::9DD8:D090:A1A6:7858 on interface Ethernet1/1.

// The DHCPv6 client added an address on interface Ethernet 1/1.
*Aug  9 14:37:40:468 2011 Sysname DHCPC6/7/DebugEvent:
  Timer Event: Timer for sending packets is destroyed.

// The timer for sending packets was removed.
*Aug  9 14:37:40:468 2011 Sysname DHCPC6/7/DebugEvent:
  Timer Event: Timer for sending packets is created, try after 300000 ms.

// The timer for sending packets was created. The aging time for the timer was T1.
*Aug  9 14:37:40:468 2011 Sysname DHCPC6/7/DebugState:
  Interface Ethernet1/1, stateful client requested for address:
    SOLICIT --> OPEN (Received a reply with success status)

// The DHCPv6 client obtained an IP address, and its state changed from SOLICIT to OPEN.
  Sent DAD event: The state of 100::9DD8:D090:A1A6:7858 on interface Ethernet1/1 is OK.

// DAD was performed on the obtained address. The address is usable.
  Received DAD event: The state of 100::9DD8:D090:A1A6:7858 on interface Ethernet1/1 is OK.

// The DHCPv6 client received the DAD result. No DAD conflict occurred.

371
debugging ipv6 dhcp relay

Use **debugging ipv6 dhcp relay** to enable DHCPv6 relay agent debugging.
Use **undo debugging ipv6 dhcp relay** to disable DHCPv6 relay agent debugging.

**Syntax**

```
debugging ipv6 dhcp relay { all | error | event | packet | timer }
undo debugging ipv6 dhcp relay { all | error | event | packet | timer }
```

**Default**

DHCPv6 relay agent debugging is disabled.

**Views**

User view

**Default command level**

1: Monitor level

**Parameters**

- **all**: Specifies all DHCPv6 relay agent debugging.
- **error**: Specifies DHCPv6 relay agent error debugging.
- **event**: Specifies DHCPv6 relay agent event debugging.
- **packet**: Specifies DHCPv6 relay agent packet debugging.
- **timer**: Specifies DHCPv6 relay agent timer debugging.

**Usage guidelines**

Table 197 describes output fields and messages for the **debugging ipv6 dhcp relay error** command.

**Table 197 Output from the debugging ipv6 dhcp relay error command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Drop a relay-reply packet for link address information error. | The Relay-reply packet is discarded because of a link layer address error in the following situations:  
- The interface ID option does not exist, and the IPv6 address in the link-address option is one of the following addresses:  
  - Unspecified address (::).
  - Loopback address (::1).
  - Multicast address of the interface-local scope (FF01::/16)
  - Multicast address of the link-local scope (FF02::/16)
- The interface ID option does not exist, and the valid interface index cannot be obtained from the link-address option. |
| The hop count in the msg-type message has reached the maximum. | The hop count in the Relay-forward or Relay-reply packet exceeds the maximum of 31. |
### Examples

# Enable DHCPv6 relay agent packet debugging. When a DHCPv6 client requests an IPv6 address through a DHCPv6 relay agent, output similar to the following example is generated:

```
<Sysname> debugging ipv6 dhcp relay packet
*Jan  8 09:04:22:985 2008 Sysname DHCPR6/7/Packet:
Packet received on interface Vlan-interface1.
Message type: INFORMATION-REQUEST (11)
Transaction ID: 0x833c98
// The DHCPv6 client received an Information-request message on VLAN-interface 1.
```

```
*Jan  8 09:04:23:985 2008 Sysname DHCPR6/7/Packet:
Sent a DHCPv6 packet.
Message type: RELAY-FORW (12)
Hop count: 0
Link address: 2001:1::2
Peer address: FE80::200:5EFF:FE28:3503
// The DHCPv6 client sent a Relay-forward message.
```

```
*Jan  8 09:04:23:94 2008 Sysname DHCPR6/7/Packet:
Packet received on interface Vlan-interface2.
Message type: RELAY-REPL (13)
Hop count: 0
Link address: 2001:1::2
Peer address: FE80::200:5EFF:FE28:3503
// The DHCPv6 client received a Relay-reply message on VLAN-interface 2.
```

```
*Jan  8 09:04:23:94 2008 Sysname DHCPR6/7/Packet:
Packet sent from interface Vlan-interface1.
Message type: REPLY (7)
Transaction ID: 0x833c98
// The DHCPv6 client sent a Reply message out of VLAN-interface 1.
```

### debugging ipv6 dhcp server

Use **debugging ipv6 dhcp server** to enable DHCPv6 server debugging.

Use **undo debugging ipv6 dhcp server** to disable DHCPv6 server debugging.

#### Syntax

- `debugging ipv6 dhcp server { all | error | event | packet [ verbose ] }`
- `undo debugging ipv6 dhcp server { all | error | event | packet }`

#### Default

DHCPv6 server debugging is disabled.
Views

User view

Default command level

1: Monitor level

Parameters

- **all**: Specifies all DHCPv6 server debugging.
- **error**: Specifies DHCPv6 server error debugging.
- **event**: Specifies DHCPv6 server event debugging.
- **packet**: Specifies DHCPv6 server packet debugging.
- **verbose**: Specifies Detailed DHCPv6 packet debugging information.

Usage guidelines

Table 198 describes output fields and messages for the **debugging ipv6 dhcp server error** command.

### Table 198 Output from the debugging ipv6 dhcp server error command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Invalid Confirm message,</td>
<td>Option values:</td>
</tr>
<tr>
<td>option status:</td>
<td>• 1—The option is available.</td>
</tr>
<tr>
<td>Auth = Auth, ClientID = ClientID, ServerID = ServerID.</td>
<td>• 0—The option is not available.</td>
</tr>
<tr>
<td>Invalid Decline message,</td>
<td>Option values:</td>
</tr>
<tr>
<td>option status:</td>
<td>• 1—The option is available.</td>
</tr>
<tr>
<td>Auth = Auth, ClientID = ClientID, ServerID = ServerID.</td>
<td>• 0—The option is not available.</td>
</tr>
</tbody>
</table>

Table 199 describes output fields and messages for the **debugging ipv6 dhcp server event** command.

### Table 199 Output from the debugging ipv6 dhcp server event command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step n: step-description</td>
<td>Steps for assigning IPv6 addresses to clients:</td>
</tr>
<tr>
<td>Interface name: Interface-name</td>
<td>1. Search in statically bound addresses.</td>
</tr>
<tr>
<td>Result: result-description</td>
<td>2. Search with hint in allocated addresses.</td>
</tr>
<tr>
<td>Current IP: ipv6-address</td>
<td>3. Search without hint in allocated addresses.</td>
</tr>
<tr>
<td></td>
<td>4. Search with hint in free addresses.</td>
</tr>
<tr>
<td></td>
<td>5. Search in timed out addresses.</td>
</tr>
<tr>
<td></td>
<td>Interface name specifies the interface of receiving Solicit messages.</td>
</tr>
<tr>
<td></td>
<td>Result indicates the IPv6 address search result:</td>
</tr>
<tr>
<td></td>
<td>• Found.</td>
</tr>
<tr>
<td></td>
<td>• Not found.</td>
</tr>
<tr>
<td></td>
<td>• Error occurred.</td>
</tr>
<tr>
<td></td>
<td>Current IP indicates the found IP address. If no IP address is found, this</td>
</tr>
<tr>
<td></td>
<td>field is not displayed.</td>
</tr>
</tbody>
</table>
Examples

# Enable all DHCPv6 server debugging on the DHCPv6 server. When a DHCPv6 client requests an IPv6
prefix, output similar to the following example is generated:

<Sysname> debugging ipv6 dhcp server all
*Jun 29 17:02:18:608 2009 Sysname DHCPS6/7/Packet: Received SOLICIT from FE80::C800:EFF:FE30:0
Verbose packet contents:
  src FE80::C800:EFF:FE30:0
dst FF02::1:2
type SOLICIT (1), xid 114D76
option ELAPSED_TIME (8), len 2
  0
option CLIENTID (1), len 10
  00030001CA000E300000
option ORO (6), len 6
  IA_PD (25)
  DNS_SERVERS (23)
  DOMAIN_LIST (24)
option IA_PD (25), len 12
  IAID 0x00040001, T1 0, T2 0

// The DHCPv6 server received a Solicit message containing an IAPD option from the client.


// The DHCPv6 server successfully created a prefix binding in address pool 1. The prefix to be assigned to the client is 2001:410::/48.

*Jun 29 17:02:18:609 2009 Sysname DHCPS6/7/Event: Back up the PD information in real time successfully.

// The DHCPv6 server successfully backed up the prefix binding in real time.

*Jun 29 17:02:18:609 2009 Sysname DHCPS6/7/Packet: Sending ADVERTISE to FE80::C800:EFF:FE30:0
Verbose packet contents:
  dst FE80::C800:EFF:FE30:0
type ADVERTISE (2), xid 114D76
option CLIENTID (1), len 10
  00030001CA000E300000
option SERVERID (2), len 10
  00030001000FE26A58ED
option PREFERENCE (7), len 1
  200
option IA_PD (25), len 41
  IAID 0x00040001, T1 43200, T2 69120
option IA_PREFIX (26), len 25
  preferred 86400, valid 259200, prefix 2001:410::/48
option DNS_SERVERS (23), len 16
  2:2:3
option DOMAIN_LIST (24), len 9
// The DHCPv6 server sent to the client an Advertise message containing the prefix 2001:410::/48.

// The DHCPv6 server received a Request message from the client.

// The DHCPv6 server successfully backed up the prefix binding in real time.
debugging ipv6 dhcp snooping

Use **debugging ipv6 dhcp snooping** to enable DHCPv6 snooping debugging. Use **undo debugging ipv6 dhcp snooping** to disable DHCPv6 snooping debugging.

**Syntax**

```
debugging ipv6 dhcp snooping { all | error | event | packet }
undo debugging ipv6 dhcp snooping { all | error | event | packet }
```

**Default**

DHCPv6 snooping debugging is disabled.

**Views**

User view

**Default command level**

1: Monitor level

**Parameters**

- **all**: Specifies all DHCPv6 snooping debugging.
- **error**: Specifies DHCPv6 snooping error debugging.
- **event**: Specifies DHCPv6 snooping event debugging.
- **packet**: Specifies DHCPv6 snooping packet debugging.

**Examples**

# Enable DHCPv6 snooping packet debugging on a DHCPv6 snooping device. When the device receives a DHCPv6 reply packet, output similar to the following example is generated:

```
<Sysname> terminal debugging
<Sysname> debugging ipv6 dhcp snooping packet
// DHCPv6 snooping pre-processed the packet.
// DHCPv6 snooping started to process packets.
// The DHCPv6 reply packet was received.
// DHCPv6 snooping started to forward packets.
```

377
Sending the DHCPv6 packet to VLAN.

// The DHCPv6 packet was forwarded to the VLAN.

*Aug 19 16:53:46:265 2009 Sysname DHCPSP6/7/PACKET:
Sending the packet to all ports in VLAN 1.

// The DHCPv6 packet was forwarded through all ports in VLAN 1.
Dial plan debugging commands

The output description tables in this document only contain fields and messages that require an explanation.

debugging voice dpl

Use **debugging voice dpl** to enable dial plan debugging.

Use **undo debugging voice dpl** to disable dial plan debugging.

**Syntax**

```plaintext
debugging voice dpl { all | error | general }
undo debugging voice dpl { all | error | general }
```

**Default**

Dial plan debugging is disabled.

**Views**

User view

**Default command level**

2: System level

**Parameters**

- **all**: Specifies all types of debugging for dial plan.
- **error**: Specifies error debugging.
- **general**: Specifies general debugging.

**Usage guidelines**

Table 200 describes output fields and messages for the **debugging voice dpl error** command.

**Table 200 Output from the debugging voice dpl error command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DPL_INTF: [GetCallingNumByIfIndex] Get null calling number By VpIfIndex!</td>
<td>The calling number corresponding to the interface index was not found.</td>
</tr>
<tr>
<td>DPL_INTF: [DPL_GetCurrentEntityFromTab]The call information table index is invalid!</td>
<td>The call information table index was invalid.</td>
</tr>
<tr>
<td>DPL_SUBST: [MultTabMainProc]No suitable substitute rule found! The user number 01 have not been substituted.</td>
<td>The user number 01 was not substituted because there was no matching number substitution rule in the global number substitution rule list.</td>
</tr>
<tr>
<td>DPL_SUBST: [SingleTabMainProc]No suitable substitute rule found! The user number 01 has not been substituted.</td>
<td>The user number 01 was not substituted because there was no matching number substitution rule in the number substitution rule list applied to the voice entity or voice subscriber line.</td>
</tr>
<tr>
<td>DPL_SRCH: [GetEntity]Cannot find suitable entity!</td>
<td>No voice entity was matched.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>DPL_INTF: GetCallInfoTab Cannot get CallInfoTab! Perhaps, the table have been use up!</td>
<td>The call information table was unavailable. It was possible that the table had been used up.</td>
</tr>
<tr>
<td>DPL_INTF: UpdateEntityInfoForCall Update failed! Cannot find the entity 100!</td>
<td>Updating failed, and voice entity 100 was not found.</td>
</tr>
<tr>
<td>DPL_INTF: NextEntityForCall Entity 100 update failed! The other process will be continued!</td>
<td>Updating voice entity 100 failed, and the other processing continued.</td>
</tr>
</tbody>
</table>

**Table 201** describes output fields and messages for the **debugging voice dpl general** command.

**Table 201 Output from the debugging voice dpl general command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DPL_INTF: Get entity 100 successfully!</td>
<td>The dial plan module succeeded in obtaining voice entity 100.</td>
</tr>
<tr>
<td>DPL_SRCH: GetEntity Found access service number is 301</td>
<td>The found access service number was 301.</td>
</tr>
<tr>
<td>DPL_INTF: UpdateEntityInfoForCall The current connect number is 0. Perhaps, the configuration of entity 100 have been modified.</td>
<td>The number of current connections was 0, and the configuration of voice entity 100 might have been modified.</td>
</tr>
<tr>
<td>DPL_INTF: UpdateEntityInfoForCall The current connect number of the group is 0. Perhaps, the configuration of max-con group have been modified. Entity is 100.</td>
<td>The number of current connections of voice entity 100 in the set was 0, and the configuration of the maximum-call-connection set might have been modified.</td>
</tr>
</tbody>
</table>
| DPL_SUBST: MultTabMainProc Found suitable substitute rule: szInputFormat: ^010100$ >> szOutputFormat: 100 Input Number Type: any(0x05) >> Output Number Type: international(0x01) Input Numbering Plan: any(0x05) >> Output Numbering Plan: telex(0x04) | The following global number substitution rule was found:  
- Input number: 010100.  
- Input number type: Any.  
- Input numbering plan: Any.  
- Output number: 100.  
- Output number type: International.  
- Output numbering plan: Telex. |
| DPL_SUBST: MultTabMainProc The user number has been substituted successfully:  
- Original Number is 010100, Substituted Number is 100;  
- Original Number Type is unknown(0x00), Substituted Number Type is international(0x01);  
- Original Numbering Plan is unknown(0x00), Substituted Numbering Plan is telex(0x04). | The global number substitution succeeded:  
- Input number: 010100.  
- Input number type: Unknown.  
- Input numbering plan: Unknown.  
- Output number: 100.  
- Output number type: International.  
- Output numbering plan: Telex. |
### Field

**DPL_SUBST: [SingleTabMainProc]**

- **Found suitable substitute rule:**

  - szInputFormat: `^010100$` >> szOutputFormat: 100
  - Input Number Type: any(0x05) >> Output Number Type: international(0x01)
  - Input Numbering Plan: any(0x05) >> Output Numbering Plan: telex(0x04)

  The following number substitution rule was found for a voice subscriber line or entity:
  - Input number: 010100.
  - Input number type: Any.
  - Input numbering plan: Any.
  - Output number: 100.
  - Output number type: International.
  - Output numbering plan: Telex.

- **The user number has been substituted successfully.**
  - Original Number is 010100, Substituted Number is 100;
  - Original Number Type is unknown(0x00), Substituted Number Type is international(0x01);
  - Original Numbering Plan is unknown(0x00), Substituted Numbering Plan is telex(0x04).

**DPL_INTF: [EntityMaxConn]**

- The current number of connection has reached the max number of its group!
  - Entity is 100, Current con-num is 2
  - The group current con-num is 2, Max con-num is 2

  The number of current connections reached the maximum.
  - The number of current connections for voice entity 100 was 2. The number of current connections in the maximum-call-connection set bound to voice entity 100 was 2, and the maximum number of call connections in the set was 2, too.

**DPL_CP: [GetSelPRIIndexByEntityType]**

- Entity reach MAX amount 5! The rest cannot be selected!
  - The number of qualified voice entities found reached the maximum number 5, and other entities could not be selected any longer.

**DPL_BiT: [CompNode]**

- The number only matched the front of the Regular Expression

  The user number 20 only matched the first half part of the regular expression 20[1-9]T.

### Examples

Enable all types of debugging for dial plan. Output similar to the following example is generated when the subscriber at 0101001 attempts to dial 1001 under the following conditions:

- A number substitution rule is configured by using the rule `0 ^010....$ .... number-type any international` command.
- The called number is 1001, and the calling number is 0101001.

```plaintext
<Sysname> debugging voice dpl all
*Feb 27 12:02:11:288 2007 Sysname DPL/7/VOICE:  
  DPL_SUBST: [MultTabMainProc]No suitable substitute rule found! The user number 0 have not been substituted.
*Feb 27 12:02:11:289 2007 Sysname DPL/7/VOICE:  
  DPL_SRCH: [GetEntity]Cannot find suitable entity!
// After the subscriber dialed 0, no entity was matched.
*Feb 27 12:02:11:498 2007 Sysname DPL/7/VOICE:  
  DPL_SUBST: [MultTabMainProc]No suitable substitute rule found! The user number 01 have not been substituted.
```
After the subscriber dialed 01, no entity was matched.

After the subscriber dialed 010, no entity was matched.

After the subscriber dialed 0101, no entity was matched.

After the subscriber dialed 01010, no entity was matched.

After the subscriber dialed 010100, no entity was matched.

After the subscriber dialed 0101001, a number substitution rule was found.

szInputFormat: ^010....$  >>  szOutputFormat: ....

Input Number Type: any(0x05) >> Output Number Type: international(0x01)

The user number has been substituted successfully.

Number substitution succeeded.

Numbering Plan:
- Original Number is 0101001, Substituted Number is 1001;
- Original Number Type is unknown(0x00), Substituted Number Type is international(0x01);
- Original Numbering Plan is unknown(0x00), Substituted Numbering Plan is unknown(0x00).

The input number (dialed number) is 0101001, the output number is 1001, the input number type is unknown, the output number type was international, and the numbering plan is unchanged.

Entity 1001 was matched.

# Enable error debugging for dial plan. Output similar to the following example is generated when the subscriber at 0101001 attempts to dial 1001 under the following conditions:
• No number substitution rule is configured.
• The calling party hears busy tones.

<Sysname> debugging voice dpl error
*Feb 27 13:40:17:330 2007 Sysname DPL/7/VOICE:
    DPL_SRCH: [GetEntity]Cannot find suitable entity!

// No dial plan is configured, and any dialed number is unable to match an entity.
**DLDP debugging commands**

The output description tables in this document only contain fields and messages that require an explanation.

**debugging dldp**

Use `debugging dldp` to enable DLDP debugging.

Use `undo debugging dldp` to disable DLDP debugging.

**Syntax**

```
debugging dldp { all | error | neighbor | packet | state }
```

```
undo debugging dldp { all | error | neighbor | packet | state }
```

**Default**

DLDP debugging is disabled.

**Views**

User view

**Default command level**

1: Monitor level

**Parameters**

- **all**: Specifies all types of debugging for DLDP.
- **error**: Specifies debugging for error DLDP packets.
- **neighbor**: Specifies debugging for DLDP neighbors.
- **packets**: Specifies debugging for DLDP packets.
- **state**: Specifies debugging for DLDP state.

**Usage guidelines**

Table 202 describes output fields and messages for the `debugging dldp error` command.
Table 202 Output from the debugging dldp error command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Reason types of error packet | Types of error DLDP packets:  
- **DLDP NOT ENABLE** — DLDP is not enabled.  
- **CURRENT STATE CANN'T RECEIVE PACKET** — A port in the current DLDP state cannot receive packets.  
- **PROTOCOL ID ERROR** — The protocol ID of the packet is invalid.  
- **VERSION ID ERROR** — The version ID of the packet is invalid.  
- **INTERVAL ERROR** — The interval for sending Advertisement packets carried in the packet is invalid.  
- **AUTHTYPE ERROR** — The authentication type of the packet is invalid.  
- **PASSWORD ERROR** — The authentication password of the packet is invalid.  
- **PACKET TYPE ERROR** — The packet type is invalid. |

Table 203 describes output fields and messages for the `debugging dldp neighbor` command.

Table 203 Output from the debugging dldp neighbor command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Neighbor State | Neighbor state:  
- **UNIDIRECTIONAL** — The neighbor is in the unidirectional state.  
- **BIDIRECTIONAL** — The neighbor is in the bidirectional state.  
- **UNKNOWN** — The state of the neighbor is unknown. |
| RSY Flag | RSY flag of the neighbor, including:  
- **NO_RSY** — The RSY flag is 0.  
- **RSY** — The RSY flag is 1. |
| AgedTime | Aging time of a neighbor entry. |
| EchoTime | Echo timeout time of a neighbor. |
| Neighbor State Transition | A neighbor transited from state1 to state2. (A neighbor can be in unidirectional state, bidirectional state, and unknown state.) |

Table 204 describes output fields and messages for the `debugging dldp packet` command.

Table 204 Output from the debugging dldp packet command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DLDP ID</td>
<td>Protocol ID carried in packets.</td>
</tr>
<tr>
<td>DLDP Version ID</td>
<td>DLDP version ID carried in the packet.</td>
</tr>
</tbody>
</table>
**Table 205** describes output fields and messages for the `debugging dldp state` command.

**Table 205 Output from the debugging dldp state command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port Name</td>
<td>Port name transited from state1 to state2. A port can be in one of the following states: INITIAL, ADVERTISEMENT, PROBE, DISABLE, DELAYDOWN, ACTIVE, and INACTIVE.</td>
</tr>
<tr>
<td>Transition</td>
<td></td>
</tr>
<tr>
<td>State1</td>
<td></td>
</tr>
<tr>
<td>State2</td>
<td></td>
</tr>
</tbody>
</table>

**Examples**

# Enable debugging for error DLDP packets on Device B. When Device B sends Advertisement packets at the interval of 10 seconds, output similar to the following example is generated:

```
The output in this example was created when the following conditions exist:
• DLDP is enabled on Device A and Device B globally and on Ethernet 1/1 of both devices.
• Device A sends Advertisement packets at the interval of 5 seconds.
<DeviceB> debugging dldp error
*Apr 26 12:05:54:962 2009 DeviceB DLDP/7/ErrorPacket:
  Port Ethernet1/1 receives an error packet. Reason types of error packet: INTERVAL ERROR
  // Ethernet 1/1 of Device B received an error packet. The packet indicates that the interval of sending Advertisement packets is invalid.
```

# Enable debugging for DLDP neighbors on Device B. The output in this example was created when DLDP is enabled on Device A and Device B globally and on Ethernet 1/1 of both devices.

```
<DeviceB> debugging dldp neighbor
*Apr 26 12:07:07:731 2009 DeviceB DLDP/7/NeighborAdd:
```
Add a neighbor to the neighbor table of port Ethernet1/1.

Neighbor BridgeMac : 00e0-fc00-3333, Neighbor PortIndex : 35
Neighbor State : UNKNOWN, RSY Flag: RSY
AgedTime: 15 seconds, EchoTime : 10 seconds.

// A neighbor entry was added on Ethernet 1/1 of Device B.

# Enable debugging for DLDP packets on Device B. The output in this example was created when DLDP is enabled on Device A and Device B globally and on Ethernet 1/1 of both devices.

<DeviceB> debugging dldp packet
*Apr 26 12:10:18:523 2009 DeviceB DLDP/7/PktSndAndRcv:
  Port Ethernet1/1 receives a DLDP packet. Following is the content of the packet:
  DLDP ID : 1, DLDP Version ID : 1
  DLDP Packet Type : ADVERTISEMENT, Flags : NO_RSY
  Authentication mode : NONE, Authentication Password :
  HostBridgeMac : 00e0-fc00-3333, HostPortIndex : 9
  Interval of sending Advertisement packet : 5 seconds.

// Ethernet 1/1 of Device B sent a DLDP packet.

# Enable debugging for DLDP state machines on Device B. The output in this example was created when DLDP is enabled on Device B globally and on Ethernet 1/1. When you disable DLDP globally on Device B, output similar to the following example is generated:

<DeviceB> debugging dldp state
*Apr 26 12:12:22:653 2009 DeviceB DLDP/7/StateTransition:
  Port Ethernet1/1 State Transition : INITIAL --> INACTIVE

// The DLDP state of Ethernet 1/1 of Device B transited from INITIAL to INACTIVE.
DLSw debugging commands

The output description tables in this document only contain fields and messages that require an explanation.

DLSw debugging commands

debugging dlsw

Use `debugging dlsw` to enable DLSw debugging.

Use `undo debugging dlsw` to disable DLSw debugging.

Syntax

```
debugging dlsw { circuit [ circuit-id ] | dlc { error | packet [ downstream | upstream ] } |
ethernet-backup | filter | reachable-cache | ssp { error | packet [ receive [ ip-address ] | send [ ip-address ] ] |
tcp { error | remote [ ip-address ] } | udp }
undo debugging dlsw { circuit [ circuit-id ] | dlc { error | packet [ downstream | upstream ] } |
ethernet-backup | filter | reachable-cache | ssp { error | packet [ receive [ ip-address ] | send [ ip-address ] ] |
tcp { error | remote [ ip-address ] } | udp }
```

Default

DLSw debugging is disabled.

Views

User view

Default command level

1: Monitor level

Parameters

- **circuit**: Debugging for DLSw virtual circuit status.
- **circuit-id**: Identifier of the DLSw virtual circuit, in the range of 0 to 0xFFFFFFFF.
- **dlc**: Debugging for the DLC sub-module.
- **ethernet-backup**: Debugging for Ethernet redundancy.
- **filter**: Debugging for SAP address-based filtering.
- **reachable-cache**: Debugging for reachability information.
- **ssp**: Debugging for the SSP sub-module.
- **tcp**: Debugging for the TCP sub-module.
- **udp**: Debugging for the UDP sub-module.
- **error**: Debugging for error information of all sub-modules.
- **packet**: Debugging for packet contents of all sub-modules.
- **downstream**: Specifies packets sent to the local SNA device.
**upstream**: Specifies packets sent from the local SNA device.

**receive**: Specifies packets received from the specified remote DLSw router or all remote DLSw routers.

**send**: Specifies packets sent to the specified remote DLSw router or all remote DLSw routers.

**remote**: Debugging for the specified remote peer or all remote peers.

**ip-address**: IP address of the remote peer.

**Usage guidelines**

Table 206 describes output fields and messages for the `debugging dlsw circuit` command.

**Table 206 Output from the debugging dlsw circuit command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Circuit</td>
<td>circuit-id Identifier of the DLSw virtual circuit.</td>
</tr>
<tr>
<td>Event</td>
<td>Event State machine event of the DLSw virtual circuit.</td>
</tr>
<tr>
<td>State</td>
<td>State State transition of the state machine of the DLSw virtual circuit.</td>
</tr>
<tr>
<td>Action</td>
<td>Action Action taken by the DLSw virtual circuit state machine for the event.</td>
</tr>
</tbody>
</table>

Table 207 describes output fields and messages for the `debugging dlsw reachable-cache` command.

**Table 207 Output from the debugging dlsw reachable-cache command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Event</td>
<td>Event DLSw reachability event.</td>
</tr>
<tr>
<td>State</td>
<td>State State transition of the DLSw reachability information entry.</td>
</tr>
<tr>
<td>Action</td>
<td>Action Action taken, such as sending an explorer frame or sending an acknowledgment.</td>
</tr>
</tbody>
</table>

Table 208 describes output fields and messages for the `debugging dlsw dlc error` command.

**Table 208 Output from the debugging dlsw dlc error command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Error</td>
<td>Error An error that occurred in the DLC sub-module. This error can be a module-level error, such as failure of finding the DLSw virtual circuit for a packet, or a system-level error, such as memory request failure.</td>
</tr>
</tbody>
</table>

Table 209 describes output fields and messages for the `debugging dlsw dlc packet` command.

**Table 209 Output from the debugging dlsw dlc packet command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet</td>
<td>Upstream/Downstream Packet direction.</td>
</tr>
<tr>
<td>Packet Len</td>
<td>Packet Len = length Packet length.</td>
</tr>
<tr>
<td>Partial data</td>
<td>Partial data Part of the packet data.</td>
</tr>
</tbody>
</table>
Table 210 describes output fields and messages for the `debugging dlsw ethernet-backup` command.

**Table 210 Output from the debugging dlsw ethernet-backup command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DLSW_EB: Hold timeout</td>
<td>The IWANTIT timer expired. The UGOTIT or CKT_TKN was sent.</td>
</tr>
<tr>
<td>DLSW_EB: Start hold timer interval = %d.</td>
<td>Started the timer waiting for the IWANTIT frames.</td>
</tr>
<tr>
<td>DLSW_EB: There is no neighbor, master get the circuit.</td>
<td>There is no neighbor. The primary router established the circuit.</td>
</tr>
<tr>
<td>DLSW_EB: Master get the circuit.</td>
<td>The primary router created the circuit through calculation.</td>
</tr>
<tr>
<td>DLSW_EB: Master giveup the circuit.</td>
<td>The primary router gave up establishing the circuit. The secondary router will establish the circuit.</td>
</tr>
<tr>
<td>DLSW_EB: Circuit exist on master, master get the circuit.</td>
<td>The circuit already exists, and was created by the primary router.</td>
</tr>
<tr>
<td>DLSW_EB: MASTER -&gt; SLAVE</td>
<td>State switchover.</td>
</tr>
<tr>
<td>DLSW_EB: Destination address map xxxxxxxxxx-&gt;xxxxx</td>
<td>Destination address mapping.</td>
</tr>
<tr>
<td>DLSW_EB: Source address map xxxxxxxxxx-&gt;xxxxx</td>
<td>Source address mapping.</td>
</tr>
</tbody>
</table>

Table 211 describes output fields and messages for the `debugging dlsw filter` command.

**Table 211 Output from the debugging dlsw filter command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>lsap-output-acl xxxx</td>
<td>Number of the ACL for packet filtering.</td>
</tr>
<tr>
<td>rule xxx</td>
<td>Rule ID in the ACL.</td>
</tr>
<tr>
<td>The packet to x.x.x.x was discarded</td>
<td>The packet sent to the remote peer x.x.x.x was discarded.</td>
</tr>
<tr>
<td>The head of the packet is below</td>
<td>Packet header information.</td>
</tr>
</tbody>
</table>

Table 212 describes output fields and messages for the `debugging dlsw ssp error` command.

**Table 212 Output from the debugging dlsw ssp error command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Error</td>
<td>An error that occurred in the SSP sub-module. This error can be a module-level error, such as failure of finding the DLSw virtual circuit for a packet, or a system-level error, such as memory request failure.</td>
</tr>
</tbody>
</table>

Table 213 describes output fields and messages for the `debugging dlsw ssp packet` command.

**Table 213 Output from the debugging dlsw ssp packet command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DLSw PACKET ip-address</td>
<td>IP address of the remote peer for the packet.</td>
</tr>
</tbody>
</table>
Field | Description
---|---
TCP/UDP | Packet sent/received via TCP/UDP.
Output/Input | Packet direction, where "output" or "input" corresponds to the send or receive keyword respectively.
Partial data | Part of the packet data.

Table 214 describes output fields and messages for the **debugging dlsw tcp error** command.

### Table 214 Output from the debugging dlsw tcp error command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Error</td>
<td>An error that occurred in the TCP sub-module. This error can be a module-level error, such as failure of establishing a TCP connection with the remote peer, or a system-level error, such as memory request failure.</td>
</tr>
</tbody>
</table>

Table 215 describes output fields and messages for the **debugging dlsw tcp remote** command.

### Table 215 Output from the debugging dlsw tcp remote command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DLSw_TCP ip-address</td>
<td>IP address of the remote peer.</td>
</tr>
<tr>
<td>Action</td>
<td>Action taken by the remote peer, such as receiving or sending packet.</td>
</tr>
</tbody>
</table>

Table 216 describes output fields and messages for the **debugging dlsw udp** command.

### Table 216 Output from the debugging dlsw udp command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Error</td>
<td>An error that occurred in the UDP sub-module. This error can be a module-level error, such as failure of creating a UDP socket, or a system-level error, such as memory request failure.</td>
</tr>
</tbody>
</table>

**Examples**

**Figure 4 LLC2-to-LLC2 network diagram**

![LLC2-to-LLC2 network diagram](image)

# Enable DLSw debugging on the router connecting with the IBM host in Figure 4. The output shows the complete process of DLSw virtual circuit setup, including:

- TCP connection establishment.
- Capabilities exchange.
- Reachability information learning.
• The state transition process of the virtual circuit from DISCONNECTED state to CONNECTED state.

```plaintext
<Sysname> display debugging
DLSw TCP remote debugging switch is on
DLSw TCP error debugging switch is on
DLSw circuit debugging switch is on
DLSw reachable-cache debugging switch is on
DLSw DLC error debugging switch is on
DLSw DLC packet upstream debugging switch is on
DLSw DLC packet downstream debugging switch is on
DLSw UDP debugging switch is on
DLSw SSP error debugging switch is on
DLSw SSP packet receive debugging switch is on
DLSw SSP packet send debuging switch is on

// All types of DLSw debugging have been enabled.
*0.2404191 Sysname DLSW/7/DLSw_TCP:
DLSw_TCP 1.1.1.2: only one tcp session is created, try to connect remote
*0.2404195 Sysname DLSW/7/DLSw_TCP:
DLSw_TCP 1.1.1.2: send capability exchange frame
*0.2404195 Sysname DLSW/7/DLSw_TCP:
DLSw_PACKET 1.1.1.2 ( TCP Output)
*0.2404195 Sysname DLSW/7/DLSw_SSP:
085d0a63 31 48 00 a9 00 00 00 00 00 00 00 00 00 00 00 20 00
085d0a73 42 01 00 00 00 00 00 20 00 00 00 00 00 00 00 00
085d0a83 00 00 00 00 00 00 01 00 00 00 00 00 00 00 00 00
085d0a93 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
085d0aa3 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
085d0ab3 fc 04 82 02 00 04 83 00 28 12 86 ff 00 00 00 00
085d0ac3 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
085d0ad3 20 73 6f 66 74 77 61 72 65 2c 20 56 65 72 73 69
085d0ae3 6f 6e 20 35 2e 32 00 2c 20 42 65 74 61 20 31 31
085d0af3 0d 0a 20 20 43 6f 70 79 72 69 67 68 74 20 (63 29 20
085d0b03 28 63 29 20 31 39 39 38 2d 32 30 36 20 48 75
085d0b13 61 77 65 69 20 54 65 63 68 6f 6c 6f 67 69 65
085d0b23 73 20 6c 61 6e 67 74 7e 63 20 4c 6f 77 73 20 72
085d0b33 72 66 66 6f 6e 65 64 2e 20 38 85 00 03 87 01
085d0b43 39 20 20 43 6f 70 79 72 69 67 68 74 20 20 48 75
085d0b53 01

*0.2404195 Sysname DLSW/7/DLSw_TCP:
DLSw_TCP 1.1.1.2: send whole packet, len=241
*0.2404199 Sysname DLSW/7/DLSw_TCP:
DLSw_TCP 1.1.1.2: recv a whole packet, len=241
*0.2404199 Sysname DLSW/7/DLSw_TCP:
DLSw_PACKET 1.1.1.2 ( TCP Input)
*0.2404199 Sysname DLSW/7/DLSw_SSP:
085be0e3 31 48 00 a9 00 00 00 00 00 00 00 00 00 00 00 20 00
085bef03 42 01 00 00 00 00 00 20 00 00 00 00 00 00 00 00
085be203 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
085be213 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

392
*0.2404200 Sysname DLSW/7/DLSw_TCP:
DLSw_TCP 1.1.1.2: receive a capability exchange frame
*0.2404400 Sysname DLSW/7/DLSw_TCP:
DLSw_TCP 1.1.1.2: send capability exchange response frame
*0.2404400 Sysname DLSW/7/DLSw_TCP:
DLSw_PACKET 1.1.1.2 ( TCP Output)
*0.2404600 Sysname DLSW/7/DLSw_SSP:
085d0a63 31 48 00 04 00 00 00 00 00 00 00 00 00 00 00 20 00
085d0a73 42 01 00 00 00 00 00 20 00 00 00 00 00 00 00 00 00
085d0a83 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
085d0a93 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
085d0aa3 00 00 00 00 00 00 00 00 00 04 15 21
*0.2405600 Sysname DLSW/7/DLSw_TCP:
DLSw_TCP 1.1.1.2: send whole packet, len=76
*0.2405600 Sysname DLSW/7/DLSw_TCP:
DLSw_TCP 1.1.1.2: recv a whole packet, len=76
*0.2405800 Sysname DLSW/7/DLSw_TCP:
DLSw_PACKET 1.1.1.2 ( TCP Input)
*0.2405810 Sysname DLSW/7/DLSw_SSP:
083356e3 31 48 00 04 00 00 00 00 00 00 00 00 00 00 00 20 00
083356f3 42 01 00 00 00 00 00 20 00 00 00 00 00 00 00 00 00
08335703 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
08335713 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
08335723 00 00 00 00 00 00 00 00 00 04 15 21
*0.2406810 Sysname DLSW/7/DLSw_TCP:
DLSw_TCP 1.1.1.2: receive a capability exchange response frame
*0.2407010 Sysname DLSW/7/DLSw_TCP:
DLSw_TCP: ASYN_PEERCLOSE
*0.2407010 Sysname DLSW/7/DLSw_TCP:
DLSw_TCP: ASYN_CLOSE

The output shows the process of TCP connection establishment and capabilities exchange between the DLSw routers. After this, all packet exchanges will take place on this TCP connection.
The device received a CANUREACH_ex packet from the remote peer, and learned a new remote reachability entry. The remote reachability entry has the MAC address of the remote SNA device and the origin of the entry (0000-1738-6dfd, 1.1.2).

The CANUREACH_ex packet queried whether the local reachability entry that matches the MAC address 0020-357b-e065. The query result is NOT_FOUND. A TEST_CMD packet (with the dsap value of 0) is sent to all local interfaces associated with DLSw.

The device received a TEST_RSP packet from Ethernet 1/0, and learned a new local reachability entry. The local reachability entry has the source MAC address and the origin of the packet (0000-357-6dfd, Ethernet 1/0).

The device received a CANUREACHf_ex packet from the remote peer. Because the reachability entry already exists and the state is FOUND, the device did not learn it again.
The CANUREACH_ex packet queried whether the local reachability entry that matches the MAC address 0020-357b-e065. The query result is FOUND. The device responds to the remote peer with an ICANREACH_ex packet.

The device received a CANUREACH_cs packet, and triggered the establishment of a DLSw virtual circuit with the circuit ID of ce000a.

The state of circuit ce000a changed from DISCONNECTED to RESOLVE_PENDING. The action is sending a TEST_CMD packet (with a non-0 dsap value) to the SNA device with a native MAC address of 0020-357b-e065.
Event DLC_DL_STARTED
State RESOLVE_PENDING --> CIRCUIT_PENDING.
Action SSP_TYPE_ICANREACH_CS.
*0.2427260 Sysname DLSW/7/DLSw_TCP:
DLSw_PACKET 1.1.1.2 (TCP Output)
*0.2427460 Sysname DLSW/7/DLSw_SSP:
 08041109 31 48 00 00 00 ce 00 0a 00 0f 00 00 00 00 04 00
 08041119 42 01 00 00 88 00 00 04 00 20 35 7b e0 65 00 00
 08041129 17 38 6d fd 04 04 02 00 00 00 00 00 00 0f 00 00
 08041139 00 ce 00 0a ff ff ff ff 0f 00 00 00 ce 00 0a
 08041149 00 00 00 00 00 00 00 00
*0.2427460 Sysname DLSW/7/DLSw_TCP:
DLSw_TCP 1.1.1.2: send whole packet, len=72
// The device received a response from Ethernet 1/0 and the state of circuit ce000a transited from RESOLVE_PENDING to CIRCUIT_PENDING. The action is sending an ICANREACH_cs packet to the peer router.
*0.2427660 Sysname DLSW/7/DLSw_TCP:
DLSw_TCP 1.1.1.2: recv a whole packet, len=72
*0.2427860 Sysname DLSW/7/DLSw_TCP:
DLSw_PACKET 1.1.1.2 (TCP Input)
*0.2428060 Sysname DLSW/7/DLSw_SSP:
 08041103 31 48 00 00 00 ce 00 0a 00 0f 00 00 00 00 05 00
 08041113 42 01 00 00 88 00 00 05 00 20 35 7b e0 65 00 00
 08041123 17 38 6d fd 04 04 01 00 00 00 00 00 00 0f 00 00
 08041133 00 ce 00 0a 00 00 00 00 00 00 0f 00 00 00 ce 00 0a
 08041143 00 00 00 00 00 00 00 00
*0.2428260 Sysname DLSW/7/DLSw_CIR:
DLSW FSM, Circuit ce000a:
  Event RCV_REACH_ACK
  State CIRCUIT_PENDING --> CIRCUIT_ESTABLISHED.
  Action Drop frame.
// The device received a REACH_ACK packet from the peer, the state of circuit ce000a transited from CIRCUIT_PENDING to CIRCUIT_ESTABLISHED. The action is dropping the packet and waiting for the subsequent XID (exchange identification) packet exchange.
*0.2428670 Sysname DLSW/7/DLSw_TCP:
DLSw_TCP 1.1.1.2: recv a whole packet, len=72
*0.2428870 Sysname DLSW/7/DLSw_TCP:
DLSw_PACKET 1.1.1.2 (TCP Input)
*0.2428870 Sysname DLSW/7/DLSw_SSP:
 0833b243 31 48 00 00 00 ce 00 0a 00 0f 00 00 00 00 07 80
 0833b253 42 01 00 00 88 00 00 07 00 20 35 7b e0 65 00 00
 0833b263 17 38 6d fd 04 04 01 00 00 00 00 00 00 0f 00 00
 0833b273 00 ce 00 0a 00 00 00 00 00 00 0f 00 00 00 ce 00 0a
 0833b283 00 00 00 00 00 00 00 00
*0.2429270 Sysname DLSW/7/DLSw_CIR:
Circuit ce000a Receive a Rpt FCI
*0.2429270 Sysname DLSW/7/DLSw_DLC:
DLSw_DLC Packet Downstream, Packet Len = 17, Partial data as follows:
Sysname DLSW/7/DLSw_CIR:
DLSW FSM, Circuit ce000a:
  Event RCV_XIDFRAME
  State CIRCUIT_ESTABLISHED --> CIRCUIT_ESTABLISHED.
  Action LLC2_CTRL_XID_CMD.
*0.2429470 Sysname DLSW/7/DLSw_DLC:
DLSw_DLC Packet Upstream, Packet Len = 60, Partial data as follows:
08041960 00 00 17 38 6d fd 00 20 35 7b e0 65 00 03 04 05
08041970 bf 01 00 01 00 00 00 00 00 00 00 00 00 00 00 00
08041980 00 00 00 00 00 00 00 00 00 00 00 00 00 00 01 e5 f5 c0
08041990 00 00 00 2c e1 e5 f0 01 01 e5 f6 04
*0.2429870 Sysname DLSW/7/DLSw_CIR:
Circuit ce000a repeat the current receive window.
*0.2430070 Sysname DLSW/7/DLSw_CIR:
DLSW FSM, Circuit ce000a:
  Event DLC_XID
  State CIRCUIT_ESTABLISHED --> CIRCUIT_ESTABLISHED.
  Action SSP_TYPE_XIDFRAME.
*0.2430070 Sysname DLSW/7/DLSw_TCP:
DLSw_TCP 1.1.1.2 ( TCP Output)
*0.2430280 Sysname DLSW/7/DLSw_SSP:
08041929 31 48 00 00 00 ce 00 0a 00 0f 00 00 00 00 07 c0
08041939 42 01 00 00 88 00 00 07 00 20 35 7b e0 65 00 00
08041949 17 38 6d fd 04 04 02 00 00 00 00 00 00 0f 00 00
08041959 00 ce 00 0a 00 00 00 00 00 0f 00 00 00 ce 00 0a
08041969 00 00 00 00 00 00 00 00
*0.2430480 Sysname DLSW/7/DLSw_TCP:
DLSw_TCP 1.1.1.2: send whole packet, len=72
*0.2430680 Sysname DLSW/7/DLSw_DLC:
DLSw_DLC Packet Upstream, Packet Len = 87, Partial data as follows:
08042180 00 00 17 38 6d fd 00 20 35 7b e0 65 00 49 04 04
08042190 bf 32 46 05 62 e5 50 00 00 00 08 00 00 00 00 00
080421a0 00 15 01 0b 70 00 05 d8 00 00 00 00 07 00 0e 0e
080421b0 f4 c1 d7 d7 d5 4b e2 f6 f5 c3 f2 c6 c2 c1 10 17
080421c0 f1 16 11 01 13 00 11 f9 f4 f0 f6 f1 f7 f0 f6 f5
080421d0 c3 f2 c6 c2 c1 40 40
*0.2430880 Sysname DLSW/7/DLSw_CIR:
Circuit ce000a repeat the current receive window.
*0.2431080 Sysname DLSW/7/DLSw_CIR:
DLSW FSM, Circuit ce000a:
  Event DLC_XID
  State CIRCUIT_ESTABLISHED --> CIRCUIT_ESTABLISHED.
  Action SSP_TYPE_XIDFRAME.
*0.2431480 Sysname DLSW/7/DLSw_TCP:
DLSw_PACKET 1.1.1.2 ( TCP Output)
*0.2431480 Sysname DLSW/7/DLSw_SSP:
08042149 31 48 00 46 00 ce 00 0a 00 0f 00 00 00 00 07 80
08042159 42 01 00 00 88 00 00 07 00 20 35 7b e0 65 00 00
08042169 17 38 6d fd 04 04 02 00 00 00 00 00 00 32 46 05 62
08042179 00 ce 00 0a 00 00 00 00 00 0f 00 00 00 ce 00 0a
08042189 00 00 00 00 00 00 00 00 00 00 00 00 15 01 0b 70 00 05 0d 80
08042199 00 08 00 00 00 00 00 00 32 46 05 62 e5 50 00 00
080421a9 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
080421b9 c3 f2 c6 c2 c1 10 17 f1 16 11 01 13 00 11 f4 080421c9 f0 f6 f1 f7 f0 f6 f5 c3 f2 c6 c2 c1 40 40

*0.2431680 Sysname DLSW/7/DLSw_TCP:
DLSw_TCP 1.1.1.2: send whole packet, len=142
*0.2431880 Sysname DLSW/7/DLSw_TCP:
DLSw_TCP 1.1.1.2: recv a whole packet, len=146
*0.2432080 Sysname DLSW/7/DLSw_TCP:
DLSw_PACKET 1.1.1.2 ( TCP Input)
*0.2432290 Sysname DLSW/7/DLSw_SSP:
0833d2c3 31 48 00 4a 00 ce 00 0a 00 0f 00 00 00 00 07 0c
0833d2d3 42 01 00 00 88 00 00 07 00 20 35 7b e0 65 00 00
0833d2e3 17 38 6d fd 04 04 01 00 00 00 00 00 00 0d 00 00
0833d2f3 00 ce 00 0a 00 00 00 00 00 00 0f 00 00 00 ce 00 0a
0833d303 00 00 00 00 00 00 00 00 00 00 00 00 3d 4a 05 60 00 01 00 00
0833d313 00 bb d1 00 00 00 00 80 00 01 0b 71 00 01 09 00
0833d323 00 00 00 07 00 0e 0b f4 c1 d7 d7 d5 4b e2 d5 c1
0833d333 f0 f1 10 1e 00 1d 11 0c 08 04 f0 f5 f0 f1 f0 f1
0833d343 09 06 e6 c1 d5 e6 c1 d9 c5 09 08 e4 e2 c5 d9 f0
0833d353 40 40

*0.2432690 Sysname DLSW/7/DLSw_CIR:
Circuit ce000a Receive a Rpt FCI
*0.2432890 Sysname DLSW/7/DLSw_DLC:
DLSw_DLC Packet Downstream, Packet Len = 91, Partial data as follows:
0833d2fa 00 20 35 7b e0 65 00 00 17 38 6d fd 00 4d 04 04
0833d30a bf 32 4a 05 60 00 01 00 00 00 bb d1 00 00 00 00
0833d31a 80 00 01 0b 71 00 01 09 00 00 00 00 00 07 00 0e 0b
0833d32a f4 c1 d7 d7 d5 4b e2 d5 c1 f0 f1 10 1e 00 1d 11
0833d33a 0c 08 04 f0 f5 f0 f1 f0 f1 09 0e e6 c1 d5 e6 c1
0833d34a d9 c5 08 e4 e2 c5 d9 f0 40 40

*0.2433090 Sysname DLSW/7/DLSw_CIR:
DLSw FSM, Circuit ce000a:
  Event RCV_XIDFRAME
  State CIRCUIT_ESTABLISHED --> CIRCUIT_ESTABLISHED.
  Action LLC2_CTRL_XID_CMD.
*0.24333090 Sysname DLSW/7/DLSw_TCP:
DLSw_TCP 1.1.1.2: send whole packet, len=146
*0.24333690 Sysname DLSW/7/DLSw_TCP:
DLSw_PACKET 1.1.1.2 ( TCP Input)
*0.24333890 Sysname DLSW/7/DLSw_SSP:
08041923 31 48 00 4a 00 ce 00 0a 00 0f 00 00 00 00 07 80
08041933 42 01 00 00 88 00 00 07 00 20 35 7b e0 65 00 00
*0.2433890 Sysname DLSW/7/DLSw_CIR:
Circuit ce000a Receive a Rpt FCI

*0.2434090 Sysname DLSW/7/DLSw_DLC:
DLSw_DLC Packet Downstream, Packet Len = 91, Partial data as follows:

DLSW FSM, Circuit ce000a:
  Event RCV_XIDFRAME
  State CIRCUIT_ESTABLISHED --> CIRCUIT_ESTABLISHED.
  Action LLC2_CTRL_XID_CMD.

*0.2434700 Quidway DLSW/7/DLSw_DLC:
DLSw_DLC Sysname Upstream, Packet Len = 87, Partial data as follows:

*0.2435100 Sysname DLSW/7/DLSw_CIR:
Circuit ce000a repeat the current receive window.

*0.2435500 Sysname DLSW/7/DLSw_CIR:
DLSW FSM, Circuit ce000a:
  Event DLC_XID
  State CIRCUIT_ESTABLISHED --> CIRCUIT_ESTABLISHED.
  Action SSP_TYPE_XIDFRAME.

*0.2435500 Sysname DLSW/7/DLSw_TCP:
DLSw_PACKET 1.1.1.2 ( TCP Output)

*0.2435700 Sysname DLSW/7/DLSw_SSP:
*0.2436300 Sysname DLSW/7/DLSw_TCP:
DLSw_TCP 1.1.1.2: send whole packet, len=142

*0.2436300 Sysname DLSW/7/DLSw_DLC:
DLSw_DLC Packet Upstream, Packet Len = 87, Partial data as follows:

*0.2436700 Sysname DLSW/7/DLSw_CIR:
Circuit ce000a repeat the current receive window.

*0.2437100 Sysname DLSW/7/DLSw_CIR:
DLSW FSM, Circuit ce000a:
  Event DLC_XID
  State CIRCUIT_ESTABLISHED --> CIRCUIT_ESTABLISHED.
  Action SSP_TYPE_XIDFRAME.

*0.2437500 Sysname DLSW/7/DLSw_TCP:
DLSw_PACKET 1.1.1.2 ( TCP Output)

*0.2437710 Sysname DLSW/7/DLSw_SSP:
DLSw_PACKET 1.1.1.2 ( TCP Input)
Circuit ce000a Receive a Rpt FCI

DLSw_DLC Packet Downstream, Packet Len = 91, Partial data as follows:
0833e43a 00 20 35 7b e0 65 00 00 17 38 6d fd 00 4d 04 04
0833e44a bf 32 4a 05 60 00 01 00 00 00 b7 d1 00 00 00 00
0833e45a 80 15 01 0b 41 00 01 09 00 00 00 00 07 00 0e 0b
0833e46a f4 c1 d7 d7 d5 4b e2 d5 c1 f0 f1 10 0e 00 1d 11
0833e47a 0c 08 04 f0 f5 f0 f1 f0 f1 09 06 e6 c1 d5 e6 c1
0833e48a d9 c5 09 08 e4 e2 c5 d9 f0 40 40

Event RCV_XIDFRAME
State CIRCUIT_ESTABLISHED --> CIRCUIT_ESTABLISHED.
Action LLC2_CTRL_XID_CMD.

DLSW FSM, Circuit ce000a:

Event RCV_XIDFRAME
State CIRCUIT_ESTABLISHED --> CIRCUIT_ESTABLISHED.
Action LLC2_CTRL_XID_CMD.

The output shows that a large number of XID packets are exchanged. This is a parameter negotiation process before a connection is finally established between the SNA host and the terminal. During this process, the DLSw virtual circuit state remains in the CIRCUIT_ESTABLISHED state.
The XID packets exchange process is complete. The SNA host initiated a SAMBE packet to request for connection setup. The state of circuit ce000a transited from CIRCUIT_ESTABLISHED to CONNECT_PENDING. The SAMBE packet was encapsulated in an SSP frame, which was sent to the remote DLSw router via the TCP connection.
transited from CONNECT_PENDING to CONNECTED, and the UA packet was sent to the local SNA host. The DLSw virtual circuit is successfully established.

# Enable debugging for the DLSw SAP address-based filtering. Output similar to the following example is generated when you configure DLSw SAP address-based filtering on the router connected to the terminal.

```plaintext
<Sysname> debugging dlsw filter
*Mar  9 16:28:38:088 2011 Sysname DLSW/7/DLSw filter: DLSw filter: The packet to 1.1.1.1 was discarded by lsap-output-acl 4000 rule 0
   The head of the packet is below
       0A 00 00 00 00 00 08 14 00 00 00 00 00 08 00 03 04 04 F3
```

**Figure 5 Network diagram**

# Enable debugging for Ethernet redundancy on Router A and Router B, which are connected to the IBM host. The output shows the whole primary/secondary router election procedure of the DLSw Ethernet redundancy function.

```plaintext
<Sysname> debugging dlsw ethernet-backup
*Mar  9 16:04:48:677 2011 Sysname DLSW/7/DLSw_EB: DLSW_EB: Neighbor 00f0.47b8.a5df is created
*Mar  9 16:04:48:678 2011 Sysname DLSW/7/DLSw_EB: DLSW_EB: Received MP frame from GigabitEthernet0/0
*Mar  9 16:04:48:678 2011 Sysname DLSW/7/DLSw_EB: DLSW_EB: Sending SABME frame from 00f0.47b8.a0dc to 00f0.47b8.a5df
*Mar  9 16:04:48:678 2011 Sysname DLSW/7/DLSw_EB: DLSW_EB: LLC circuit timer with neighbor 000f.e21d.a5fb creates
```

403
**LLC2 debugging commands**

**debugging llc2**

Use **debugging llc2** to enable LLC2 debugging.

Use the **undo debugging llc2** command to disable LLC2 debugging.

**Syntax**

```
debugging llc2 { circuit [ circuit-index ] | error | packet }
```

```
undo debugging llc2 { circuit [ circuit-index ] | error | packet }
```

**Default**

LLC2 debugging is disabled.

**Views**

User view

**Default command level**

1: Monitor level

**Parameters**

- **circuit**: Debugging for LLC2 virtual circuit status.
- **circuit-index**: Identifier of the LLC2 virtual circuit, in the range of 0 to 4294967295.
- **error**: Debugging for error information of the LLC2 module.
- **packet**: Debugging for packet contents of the LLC2 module (for U frames only).

**Usage guidelines**

Table 217 describes output fields and messages for the **debugging llc2 circuit** command.

**Table 217 Output from the debugging llc2 circuit command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>receive from: source</td>
<td>Origin of the LLC2 packet, which can be a remote peer or a local interface.</td>
</tr>
<tr>
<td>I/S/U frame</td>
<td>Packet type.</td>
</tr>
<tr>
<td>Ns =Ns, Nr =Nr</td>
<td>Sequence number of packet received/sent.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td>LLC2 FSM event: event</td>
<td>State machine event of the LLC2 virtual circuit.</td>
</tr>
<tr>
<td>LLC2 FSM action: action</td>
<td>Action taken by the DLSw virtual circuit state machine for the event.</td>
</tr>
</tbody>
</table>

**Table 218** describes output fields and messages for the `debugging llc2 error` command.

**Table 218 Output from the debugging llc2 error command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Error</td>
<td>An error that occurred in the LLCP sub-module. This error can be a module-level error, such as failure of finding the LLC2 virtual circuit for a packet, or a system-level error, such as memory request failure.</td>
</tr>
</tbody>
</table>

**Table 219** describe output fields and messages for the `debugging llc2 packet` command.

**Table 219 Output from the debugging llc2 packet command**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>interface-type interface-number (Output / Input)</td>
<td>The local interface on which the packet was sent out/received and the packet direction.</td>
</tr>
<tr>
<td>U frame</td>
<td>Packet type. Only the information of U frames can be displayed with the LLC2 debugging command.</td>
</tr>
<tr>
<td>Partial data</td>
<td>Part of the packet data.</td>
</tr>
</tbody>
</table>

**Examples**

**Figure 6 LLC2-to-LLC2 network diagram**

![Figure 6 LLC2-to-LLC2 network diagram](image)

# Enable LLC2 debugging on the router connecting to the IBM host in Figure 6 to view the debugging information of the LLC2 sub-module in the process of DLSw virtual circuit establishment.

```bash
<Sysname>display debugging
DLSw llc2 circuit debugging switch is on
DLSw llc2 packet debugging switch is on
DLSw llc2 error debugging switch is on

// All types of debugging for the LLC2 sub-module have been enabled.
*0.191645 Sysname LLC2/7/LLC2_PKT:
  LLC2_PACKET Ethernet1/0 (Input) U frame
*0.191646 Sysname LLC2/7/LLC2_PKT:
  07fd4460 00 00 e8 1c b6 bf 00 04 ac de 07 a6 00 03 04 05
  07fd4470 f3 00 04 ac de 07 a6 00 00 e8 1c b6 00 04 ac de
  07fd4480 07 a6 00 00 e8 1c b6 bf 00 03 04 04 f3 2a 20 63
  07fd4490 6f 6d 6d 61 00 00 00 00 00 00 00 00

// TEST_RSP packet.
```

405
Sysname LLC2/7/LLC2_PKT:
LLC2_PACKET Ethernet1/0 (Output) U frame

Sysname LLC2/7/LLC2_PKT:
082f9ada 00 04 ac de 07 a6 00 00 e8 1c b6 bf 00 03 04 04
082f9aea bf

Sysname LLC2/7/LLC2_PKT:
LLC2_PACKET Ethernet0/0 (Input) U frame

Sysname LLC2/7/LLC2_PKT:
07fd4c80 00 00 e8 1c b6 bf 00 04 ac de 07 a6 00 03 04 05
07fd4c90 bf 01 3e 14 01 c8 4f a4 f2 f8 f3 f8 f0 f0 f0 f1
07fd4ca0 a1 70 0a 20 00 00 00 00 00 02 01 60 01 e7 a9 90
07fd4cb0 00 00 00 2c e1 e7 a0 01 01 e7 a9 d4

Sysname LLC2/7/LLC2_PKT:
LLC2_PACKET Ethernet1/0 (Output) U frame

Sysname LLC2/7/LLC2_PKT:
082fd4ba 00 04 ac de 07 a6 00 00 e8 1c b6 bf 00 04 04 04
082fd4ca bf 32 4a 05 60 00 01 00 00 00 bb d1 00 00 00 00
082fd4da 80 00 01 0b 71 00 01 09 00 00 00 00 07 00 0e 0b
082fd4ea f4 c1 d7 d7 d5 4b e2 d5 c1 f0 f1 10 0e 00 1d 11
082fd4fa 0c 08 04 f0 f5 f0 f1 f0 f1 09 06 e6 c1 d5 e6 c1
082fd50a d9 c5 09 08 e4 e2 c5 d9 f0 40 40

Sysname LLC2/7/LLC2_PKT:
LLC2_PACKET Ethernet1/0 (Input) U frame

Sysname LLC2/7/LLC2_PKT:
07fd54a0 00 00 e8 1c b6 bf 00 04 ac de 07 a6 00 04 04 04
07fd54b0 bf 32 46 05 62 e5 50 00 00 00 08 00 00 00 00
07fd54c0 00 00 01 0b 70 00 05 08 00 00 00 00 07 00 0e 0e
07fd54d0 f4 c1 d7 d7 d5 4b e2 f6 f5 c3 f2 c6 c2 c1 10 17
07fd54e0 f1 16 11 01 13 00 11 f9 f4 f0 f6 f1 f7 f0 f6 f5
07fd54f0 c3 f2 c6 c2 c1 40 40

Sysname LLC2/7/LLC2_PKT:
LLC2_PACKET Ethernet1/0 (Output) U frame

Sysname LLC2/7/LLC2_PKT:
082fddca 00 04 ac de 07 a6 00 00 e8 1c b6 bf 00 04 04 04
082fdce0 bf 32 4a 05 60 00 01 00 00 00 b7 d1 00 00 00 00
082fdce1 80 00 01 0b 71 00 01 09 00 00 00 00 07 00 0e 0b
082fdd0a f4 c1 d7 d7 d5 4b e2 d5 c1 f0 f1 10 0e 00 1d 11
082fdd1a 0c 08 04 f0 f5 f0 f1 f0 f1 09 06 e6 c1 d5 e6 c1
082fdd2a d9 c5 09 08 e4 e2 c5 d9 f0 40 40

Sysname LLC2/7/LLC2_PKT:
LLC2_PACKET Ethernet1/0 (Input) U frame

Sysname LLC2/7/LLC2_PKT:
07fd5cc0 00 00 e8 1c b6 bf 00 04 ac de 07 a6 00 04 04 05
07fd5cd0 bf 32 46 05 62 e5 50 00 00 00 b4 c0 00 00 00 00
07fd5ce0 80 15 01 0b 70 00 05 08 00 00 00 00 07 00 0e 0e
07fd5cf0 f4 c1 d7 d7 d5 4b e2 f6 f5 c3 f2 c6 c2 c1 10 17
07fd5d00 f1 16 11 01 13 00 11 f9 f4 f0 f6 f1 f7 f0 f6 f5
07fd5d10 c3 f2 c6 c2 c1 40 40
// XID packets exchange.

// SABME packet.

// UA packet, which shows that a DLSw virtual circuit has been successfully established.
Sysname LLC2/7/LLC2_CIR:
Nr = 0

Sysname LLC2/7/LLC2_CIR:
LLC2 FSM event: Receive an RR PDU(CMD&POLL)
LLC2 FSM action: SEND_RR_RSP(F=1)

// The output is the LLC2 debugging information output after the DLSw virtual circuit is successfully established.

SDLC debugging commands
debugging sdlc

Use `debugging sdlc` to enable SDLC debugging.
Use `undo debugging sdlc` to disable SDLC debugging.

Syntax
```
debugging sdlc { all | event | packet } [interface interface-type interface-number ]
undo debugging sdlc { all | event | packet } [interface interface-type interface-number ]
```

Default
SDLC debugging is disabled.

Views
User view

Default command level
1: Monitor level

Parameters
- `all`: All types of debugging for the SDLC module, including event and packet debugging.
- `event`: Debugging for events of the SDLC module.
- `packet`: Debugging for packet contents of the SDLC module.
- `interface interface-type interface-number`: Specifies an interface by its type and number.

Usage guidelines
Table 220 describes output fields and messages for the `debugging sdlc event` command.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>interface-type interface-number (Output / Input)</td>
<td>The local interface on which the packet is sent out/received and the packet direction.</td>
</tr>
<tr>
<td>SDLC PDU: frame type</td>
<td>Packet type.</td>
</tr>
<tr>
<td>Frame head</td>
<td>The first two bytes of the packet.</td>
</tr>
<tr>
<td>Event: event</td>
<td>State machine event of the SDLC virtual circuit.</td>
</tr>
<tr>
<td>Event: action</td>
<td>Action taken by the SDLC virtual circuit state machine for the event.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>State: state</td>
<td>State transition of the state machine of the SDLC virtual circuit.</td>
</tr>
</tbody>
</table>

Table 221 describes output fields and messages for the `debugging sdlc packet` command.

### Table 221: Output from the debugging sdlc packet command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>interface-type interface-number (Output / Input)</td>
<td>The local interface on which the packet is sent out/received and the packet direction.</td>
</tr>
<tr>
<td>SDLC PDU: frame type</td>
<td>Packet type.</td>
</tr>
<tr>
<td>Frame head</td>
<td>The first two bytes of the packet.</td>
</tr>
<tr>
<td>Partial data</td>
<td>Part of the packet data (this field is not printed if the packet length is 2 bytes).</td>
</tr>
</tbody>
</table>

### Examples

**Figure 7 SDLC-to-LLC2 network diagram**

![SDLC-to-LLC2 network diagram](image)

# Enable SDLC debugging on the router connecting with the IBM host in Figure 7 to view the debugging information of the SDLC module in the process of DLSw virtual circuit establishment.

```plaintext
<Sysname> debugging sdlc all
*0.4066531 Sysname SDLC/7/debug2:
    SDLC PDU: SDLC_XID Send to Serial2/0 c1 bf
*0.4071031 Sysname SDLC/7/debug2:
    SDLC PDU: SDLC_XID Send to Serial2/0 c1 bf
// The SDLC module sent an XID_CMD packet to the terminal.
*0.4071050 Sysname SDLC/7/debug2:
    SDLC PDU: SDLC_XID Receive from Serial2/0 c1 bf
*0.4071050 Sysname SDLC/7/debug2:
    08393d02 32 4a 05 60 00 01 00 00 00 b9 d1 00 00 00 00 80 ....J............
*0.4071050 Sysname SDLC/7/debug2:
    08393d12 00 01 0b 30 00 01 09 00 20 00 00 07 00 0e 0b f4 ...O............
*0.4071051 Sysname SDLC/7/debug2:
    08393d22 c1 d7 d7 d5 4b e2 d5 c1 f0 f1 10 1e 00 1d 11 0c ...K............
*0.4071051 Sysname SDLC/7/debug2:
    08393d32 08 04 f0 f5 f0 f1 f0 f1 09 06 e6 c1 d5 e6 c1 d9 ..............
*0.4071051 Sysname SDLC/7/debug2:
    08393d42 ...........
*0.4071052 Sysname SDLC/7/debug2:
    SDLC: Event:SDLC_RECEIVE_FULL_XID
    Action:SDLC_SEND_FULL_XID_REMOTE
    SDLC_DISCONNECT-->SDLC_XIDSTOP
```

409
The device received an XID packet from the terminal. The state of the SDLC virtual circuit transited from SDLC_DISCONNECT to SDLC_XIDSTOP. The packet was sent to the DLSw module, which then passed the packet to the remote endpoint.

*0.4071066 Sysname SDLC/7/debug2:
  SDLC: Event:SDLC_SEND_NULL_XID_REQUEST
  Action:SDLC_SEND_NULL_XID
  SDLC_XIDSTOP->SDLC_XIDSENT

*0.4071270 Sysname SDLC/7/debug2:
  SDLC PDU: SDLC_XID Send to Serial2/0 c1 bf
  SDLC: Event:SDLC_SEND_FULL_XID_REQUEST
  Action:SDLC_SEND_FULL_XID
  SDLC_XIDSENT->SDLC_XIDSENT

*0.4071470 Sysname SDLC/7/debug2:
  SDLC PDU: SDLC_XID Receive from Serial2/0 c1 bf

*0.4071470 Sysname SDLC/7/debug2:
  SDLC PDU: SDLC_XID Receive from Serial2/0 c1 bf

*0.4071470 Sysname SDLC/7/debug2:
  0839a082 32 4a 05 60 00 01 00 00 00 b9 d1 00 00 00 00 80 2J............

*0.4071680 Sysname SDLC/7/debug2:
  0839a092 00 01 0b 30 00 01 09 00 20 00 00 07 00 0e 0b f4 ...0............

*0.4071880 Sysname SDLC/7/debug2:
  0839a0a2 c1 d7 d7 d5 4b e2 d5 c1 f0 f1 10 01 0d 11 0c ....K............

*0.4071880 Sysname SDLC/7/debug2:
  0839a0b2 08 04 f0 f5 f0 f1 f0 f1 09 06 e6 c1 d5 e6 c1 d9 .......... ......K............

*0.4072080 Sysname SDLC/7/debug2:
  0839a0c2 

*0.4072280 Sysname SDLC/7/debug2:
  SDLC: Event:SDLC_RECEIVE_FULL_XID
  Action:SDLC_SEND_FULL_XID_REMOTE
  SDLC_XIDSENT->SDLC_XIDSTOP

*0.4072290 Sysname SDLC/7/debug2:
  SDLC PDU: SDLC_XID Send to Serial2/0 c1 bf

*0.4072490 Sysname SDLC/7/debug2:
  086b938b 32 46 05 62 e5 50 00 00 00 08 00 00 00 00 00 00 2F.b.P............

*0.4072490 Sysname SDLC/7/debug2:
  086b939b 16 01 0b 10 00 05 d8 00 00 00 00 07 00 0e 0e f4 ...............K............

*0.4072690 Sysname SDLC/7/debug2:
  086b93ab c1 d7 d7 d5 4b e2 f6 f5 c3 f2 c6 c2 c1 10 17 f1 ....K............

*0.4072691 Sysname SDLC/7/debug2:
  086b93bb 16 11 01 13 00 11 f9 f4 f0 f6 f1 f7 f0 f6 f5 c3 ...............K............

*0.4072890 Sysname SDLC/7/debug2:
  086b93cb 

*0.4073090 Sysname SDLC/7/debug2:
  SDLC: Event:SDLC_SEND_FULL_XID_REQUEST
  Action:SDLC_SEND_FULL_XID
  SDLC_XIDSTOP->SDLC_XIDSENT

*0.4073090 Sysname SDLC/7/debug2:
  SDLC PDU: SDLC_XID Receive from Serial2/0 c1 bf

*0.4073290 Sysname SDLC/7/debug2:

410
086b28e2 32 4a 05 60 00 01 00 00 00 b5 d1 00 00 00 80 2J.
*0.4073290 Sysname SDLC/7/debug2:
086b28f2 00 01 0b 30 00 01 09 00 20 00 00 07 00 0e 0b f4 ...0
*0.4073500 Sysname SDLC/7/debug2:
086b2902 c1 d7 d5 4b e2 d5 c1 f0 f1 10 0e 00 1d 11 0c ...K
*0.4073700 Sysname SDLC/7/debug2:
086b2912 08 04 f0 f5 f0 f1 f0 09 06 e6 c1 d5 e6 c1 d9 ............
*0.4073700 Sysname SDLC/7/debug2:
086b2922 ........
*0.4073900 Sysname SDLC/7/debug2:
SDLC: Event:SDLC_RECEIVE_FULL_XID
Action:SDLC_SEND_FULL_XID_REMOTE
SDLC_XIDSENT->SDLC_XIDSTOP
*0.4073900 Sysname SDLC/7/debug2:
SDLC PDU: SDLC_XID Send to Serial2/0 c1 bf
*0.4074100 Sysname SDLC/7/debug2:
083a0b2b 32 46 05 62 e5 50 00 00 00 b4 c0 00 00 00 80 2F.b.P
*0.4074300 Sysname SDLC/7/debug2:
083a0b3b 16 01 0b 10 00 05 d8 00 00 00 00 07 00 0e 0e f4 ............
*0.4074300 Sysname SDLC/7/debug2:
083a0b4b c1 d7 d5 4b e2 f6 0f f5 c3 f2 c6 c2 c1 10 17 f1 ...K
*0.4074500 Sysname SDLC/7/debug2:
083a0b5b 16 11 01 13 00 11 f9 f4 f0 f6 f1 f7 f0 f6 f5 c3 ............
*0.4074500 Sysname SDLC/7/debug2:
083a0b6b ........
*0.4074710 Sysname SDLC/7/debug2:
SDLC: Event:SDLC_SEND_FULL_XID_REQUEST
Action:SDLC_SEND_FULL_XID
SDLC_XIDSTOP->SDLC_XIDSENT
*0.4074710 Sysname SDLC/7/debug2:
SDLC PDU: SDLC_XID Send to Serial2/0 c1 bf
*0.4074910 Sysname SDLC/7/debug2:
07ff43ab 32 46 05 62 e5 50 00 00 00 b4 c0 00 00 00 80 2F.b.P
*0.4075110 Sysname SDLC/7/debug2:
07ff43bb 16 01 0b 10 00 05 d8 00 00 00 00 07 00 0e 0e f4 ............
*0.4075110 Sysname SDLC/7/debug2:
07ff43cb c1 d7 d5 4b e2 f6 0f f5 c3 f2 c6 c2 c1 10 17 f1 ...K
*0.4075310 Sysname SDLC/7/debug2:
07ff43db 16 11 01 13 00 11 f9 f4 f0 f6 f1 f7 f0 f6 f5 c3 ............
*0.4075310 Sysname SDLC/7/debug2:
07ff43eb ........
*0.4075510 Sysname SDLC/7/debug2:
SDLC PDU: SDLC_XID Receive from Serial2/0 c1 bf
*0.4075710 Sysname SDLC/7/debug2:
0839a8a2 32 4a 05 60 00 01 00 00 00 b5 d1 00 00 00 80 2J.
*0.4075710 Sysname SDLC/7/debug2:
0839a8b2 16 01 0b 00 00 01 09 00 20 00 00 07 00 0e 0b f4 ............
*0.4075920 Sysname SDLC/7/debug2:
The output shows the XID packets exchange process between the host and the terminal. The remote host initiated a SAMBE packet to request connection setup. The SDLC module converted the packet into an SDLC format SNRM packet, which was then sent to the local SNA terminal. At the same time, the state of SDLC virtual circuit transited from SDLC_XIDSTOP to SDLC_SNRMSENT.

The device received a UA packet from the local SNA terminal. The state of the SDLC virtual circuit transited from SDLC_SNRMSENT to SDLC_CONNECT. The packet was sent to the DLSw module, which then passed the packet to the remote endpoint. Then an RR packet (S frame) was sent to the local SNA terminal.

The output shows the S frame exchange process between the router and the local SNA terminal when the SDLC virtual circuit is in SDLC_CONNECT state.
DNS debugging commands

The output description tables in this document only contain fields and messages that require an explanation.

down debugging dns

Use `debugging dns` to enable DNS debugging.

Use `undo debugging dns` to disable DNS debugging.

Syntax

```
d debugging dns
undo debugging dns
```

Default

DNS debugging is disabled.

Views

User view

Default command level

1: Monitor level

Usage guidelines

Table 222 describes output fields and messages for the `debugging dns` command.

Table 222 Output from the debugging dns command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DNS name resolution started.</td>
<td>DNS name resolution started.</td>
</tr>
<tr>
<td>• domain-name: Domain name to be resolved.</td>
<td>Domain name to be resolved.</td>
</tr>
<tr>
<td>• type: Query type.</td>
<td>Query type.</td>
</tr>
<tr>
<td>o A—Resolves a domain name to an IPv4 address.</td>
<td>IPv4 address.</td>
</tr>
<tr>
<td>o AAAA—Resolves a domain name to an IPv6 address.</td>
<td>IPv6 address.</td>
</tr>
<tr>
<td>o NAPTR—Offers the replacement rule of a character string to convert the character string to a domain name.</td>
<td>Domain name.</td>
</tr>
<tr>
<td>o SRV—Offers the domain name of a certain service site.</td>
<td>Service site.</td>
</tr>
</tbody>
</table>

Resolve results for query name `domain-name` are found in the local dynamic cache (type: type, TTL: num seconds). Resolve results: `result-desc`. Name resolution result information.

The query type can be NAPTR or SRV.

Resolve results for query name `domain-name` are found in the local dynamic cache (type: type). Resolve results: `result-desc`. Name resolution result information.

The query type can be A or AAAA.

The TTL is not recorded for type A and AAAA queries.
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>The answers for query name <code>domain-name</code> (type: <code>type</code>, TTL: <code>num</code> seconds): <code>result-desc</code>.</td>
<td>The name resolution result for domain name <code>domain-name</code> has been obtained from the DNS reply packet. The query type can be NAPTR or SRV.</td>
</tr>
<tr>
<td>The answers for query name <code>domain-name</code> (type: <code>type</code>): <code>result-desc</code>.</td>
<td>The name resolution result for domain name <code>domain-name</code> has been obtained from the DNS reply packet. The query type can be A or AAAA. The TTL is not recorded for type A and AAAA queries.</td>
</tr>
<tr>
<td>Record number: order: <code>order</code> preference: <code>preference</code> flags: <code>flag</code> service: <code>service-type</code> regexp: <code>regexp</code> replacement: <code>replacement</code></td>
<td>Result for an NAPTR query: • Sequence number. • Precedence. • Flags. • Service type. • Replacement rule of the character string. • Replacement string.</td>
</tr>
<tr>
<td>Record number: priority: <code>priority</code> weight: <code>weight</code> port: <code>port</code> target: <code>targe-name</code></td>
<td>Result for an SRV query: • Priority. • Weight. • Port. • Domain name of the service site.</td>
</tr>
<tr>
<td>The class of DNS reply is not supported.</td>
<td>The protocol type of the DNS reply is not supported. Only Internet protocols are supported.</td>
</tr>
<tr>
<td>Delete the resolve results for query name <code>domain-name</code> from the local dynamic cache (type: <code>type</code>, TTL: <code>num</code> seconds).</td>
<td>The record of the domain name <code>domain-name</code> has been aged out from the dynamic cache. The query type can be NAPTR or SRV.</td>
</tr>
<tr>
<td>Delete the resolve results for query name <code>domain-name</code> from the local dynamic cache (type: <code>type</code>).</td>
<td>The record of the domain name <code>domain-name</code> has been aged out from the dynamic cache. The query type can be A or AAAA. The TTL is not recorded for type A and AAAA queries.</td>
</tr>
</tbody>
</table>

**Examples**

# On the DNS client, configure a DNS server 192.168.10.1 and a domain name suffix com.
```bash
<Sysname> system-view
<Sysname> [Sysname] dns domain com
<Sysname> [Sysname] dns server 192.168.10.1
<Sysname> [Sysname] quit
```

# Enable DNS debugging.
```bash
<Sysname> [Sysname] debugging dns
<Sysname> [Sysname] terminal debugging
<Sysname> [Sysname] terminal monitor
```

# Execute the ping sample command.
```bash
<Sysname> ping sample
```

Output similar to the following messages is generated:
```
Trying DNS resolve, press CTRL_C to break
```
Trying DNS server (192.168.10.1)

*Feb 9 17:21:40:719 2010 H3C DNS/7/dns: There is no resolve result matching the query name sample.com in the local dynamic cache.

// DNS did not find a resolution result for domain name sample.com in the local dynamic cache.


// DNS constructed a DNS query packet for resolving domain name sample.com.

*Feb 9 17:21:40:735 2010 H3C DNS/7/dns: Send the packet to DNS server 192.168.10.1 for 1 times.

// DNS sent the DNS query packet to the DNS server for the first time.


// DNS received a DNS reply packet from the DNS server. The IP address for the domain name is 1.1.1.1.

PING sample.com (1.1.1.1):
56 data bytes, press CTRL_C to break
   Reply from 1.1.1.1: bytes=56 Sequence=1 ttl=128 time=4 ms
   Reply from 1.1.1.1: bytes=56 Sequence=2 ttl=128 time=1 ms
   Reply from 1.1.1.1: bytes=56 Sequence=3 ttl=128 time=1 ms
   Reply from 1.1.1.1: bytes=56 Sequence=4 ttl=128 time=1 ms
   Reply from 1.1.1.1: bytes=56 Sequence=5 ttl=128 time=1 ms
--- sample.com ping statistics ---
  5 packet(s) transmitted
  5 packet(s) received
  0.00% packet loss
  round-trip min/avg/max = 1/1/4 ms

debugging dns proxy

Use debugging dns proxy to enable DNS proxy debugging.

Use undo debugging dns proxy to disable DNS proxy debugging.

Syntax

debugging dns proxy { all | event | packet | error }
undo debugging dns proxy { all | event | packet | error }

Default

DNS proxy debugging is disabled.

Views

User view

Default command level

1: Monitor level

Parameters

all: Specifies all types of debugging for DNS proxy.

event: Specifies DNS proxy event debugging.
**packet**: Specifies DNS proxy packet debugging.

**error**: Specifies DNS proxy error debugging.

**Usage guidelines**

Table 223 describes output fields and messages for the `debugging dns proxy event` command.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Begin to find query information locally.</td>
<td>DNS proxy starts searching information locally according to the client’s query.</td>
</tr>
<tr>
<td>Opcode: operation-code</td>
<td>Operation code of the query:</td>
</tr>
<tr>
<td>Query type: query-type</td>
<td>Query type:</td>
</tr>
<tr>
<td>Question: query-question</td>
<td>Query type:</td>
</tr>
<tr>
<td>• 0—Standard query.</td>
<td></td>
</tr>
<tr>
<td>• 1—Inverse query.</td>
<td></td>
</tr>
<tr>
<td>• 2—Server status query.</td>
<td></td>
</tr>
<tr>
<td>• 1—Type A query.</td>
<td></td>
</tr>
<tr>
<td>• 12—Type PTR query.</td>
<td></td>
</tr>
<tr>
<td>• 28—Type AAAA query.</td>
<td></td>
</tr>
<tr>
<td>Requested information is found locally.</td>
<td>DNS proxy locates the requested host name or IP address locally.</td>
</tr>
<tr>
<td>Query type: query-type</td>
<td>Query type:</td>
</tr>
<tr>
<td>Question: query-question</td>
<td>Query type:</td>
</tr>
<tr>
<td>Answer: answer</td>
<td>Query type:</td>
</tr>
<tr>
<td>• 1—Type A query.</td>
<td></td>
</tr>
<tr>
<td>• 12—Type PTR query.</td>
<td></td>
</tr>
<tr>
<td>• 28—Type AAAA query.</td>
<td></td>
</tr>
</tbody>
</table>

Table 224 describes output fields and messages for the `debugging dns proxy error` command.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Failed to set socket ioctl option, socket ID: sock-id, error: error code.</td>
<td>DNS failed to set input output control option for the socket.</td>
</tr>
<tr>
<td>Failed to add an item for too many items.</td>
<td>DNS failed to add an item because the number of items reached the maximum.</td>
</tr>
</tbody>
</table>

**Examples**

# Enable DNS proxy.

```bash
<Sysname> system-view
[Sysname] dns proxy enable
```

# Specify the IP address of the domain name server as 192.168.0.244 on the DNS proxy.

```bash
[Sysname] dns server 192.168.0.244
```

# Configure a static entry with host name www.aabbcc.com and IP address 11.11.11.11 on the DNS proxy. The DNS proxy is specified as the domain name server on the DNS client.

```bash
[Sysname] ip host www.aabbcc.com 11.11.11.11
[Sysname] quit
```

# Enable all types of debugging for DNS proxy.
Output similar to the following messages is generated on the DNS proxy when the client pings www.aabbcc.com:


// DNS received a DNS request from the client at 192.168.5.212.


Opcode: 0
Query type: 1
Question: www.aabbcc.com

// DNS started searching information locally according to the request.

*Apr 19 14:07:29:554 2007 Sysname DNS/7/DNS_Proxy: Requested information is found locally.

Query type: 1
Question: www.aabbcc.com
Answer: 11.11.11.11

// DNS found a matching entry.


// DNS sent a reply to the client at 192.168.5.212.

Output similar to the following messages is generated on the DNS proxy when the client pings www.abc.com:


// DNS received a DNS request from the client at 192.168.5.212.


Opcode: 0
Query type: 1
Question: www.abc.com

// DNS started searching information locally according to the request.


// DNS sent a request to the domain name server 192.168.0.244.


// DNS received a reply from the domain name server at 192.168.0.244.


// DNS sent a reply to the client at 192.168.5.212.
DDNS debugging commands

The output description tables in this document only contain fields and messages that require an explanation.

debugging ddns

Use debugging ddns to enable DDNS debugging.

Use undo debugging ddns to disable DDNS debugging.

Syntax

depending on ddns

undo debugging ddns

Default

DDNS debugging is disabled.

Views

User view

Default command level

1: Monitor level

Usage guidelines

Table 225 describes output fields and messages for the debugging ddns command.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DDNS event (Interface: if-name, policy: policy-name)</td>
<td>A DDNS event occurs when interface if-name uses DDNS policy policy-name for DDNS update.</td>
</tr>
<tr>
<td>Get server IP address ip-addr.</td>
<td>The system obtains the IP address ip-addr of the DDNS server and is ready to initiate the DDNS update.</td>
</tr>
<tr>
<td>Authentication mode was sent.</td>
<td>The system has sent the authentication mode to the PeanutHull server for the update.</td>
</tr>
<tr>
<td>Authentication request was sent.</td>
<td>The system has sent the authentication request to the PeanutHull server for the update.</td>
</tr>
<tr>
<td>Domain name was sent.</td>
<td>The system has sent the domain name to the PeanutHull server for the update.</td>
</tr>
<tr>
<td>DDNS server reply (Interface: if-name, policy: policy-name): packet-content</td>
<td>Reply (Interface: if-name, policy: policy-name) is received from the DDNS server. The content of the reply packet is packet-content.</td>
</tr>
<tr>
<td>DDNS error (Interface: if-name, policy: policy-name)</td>
<td>A DDNS error occurs on interface if-name using DDNS policy policy-name for DDNS update.</td>
</tr>
<tr>
<td>Parameters incomplete.</td>
<td>The DDNS policy lacks required parameters.</td>
</tr>
</tbody>
</table>
### Field
**Interface is not ready.**
The interface is not ready to initiate the DDNS update. The interface is either down or not configured with an IP address.

**Invalid server name.**
The host name of the DDNS server is invalid. The system does not start domain name translation.

**Failed to send connection request, with error code `error-code`.**
The system failed to establish the update request connection to the DDNS server. The error code is `error-code`.

**Failed to connect server.**
The system cannot connect to the DDNS server.

**Last update not finished, the DDNS update skipped.**
Current update is cancelled because there is an ongoing DDNS update. The current state is `current-state`.

**DDNS state:**
- **waiting**—The system is waiting for the interface state to be stable.
- **resolving**—The domain name resolution is ongoing.
- **updating**—The DDNS update is ongoing.
- **sleeping**—The system is waiting for next DDNS update.
- **invalid**—The DDNS policy is invalid.

**Failed to get server IP address.**
The system cannot obtain the IP address of the DDNS server.

**Failed to create SSL context.**
The system fails to create the SSL context for the HTTPS-based DDNS update.

**Failed to create SSL.**
The system fails to create SSL for the HTTPS-based DDNS update.

**Failed to close SSL.**
The system fails to close the SSL for the HTTPS-based DDNS update.

**Failed to get source IP address.**
The system fails to obtain the source IP address when contacting the PeanutHull server for the update.

**Failed to bind IP address to socket, with error code `error-code`.**
The system fails to bind the source IP address to the socket when contacting the PeanutHull server for the update.

**Failed to negotiate with server.**
The system fails to negotiate with the PeanutHull server for the update.

**Domain name not found.**
The system cannot find a match for the domain name when contacting the PeanutHull server for the update.

---

### Examples

# Enable DDNS debugging.

```bash
<Sysname> terminal monitor
<Sysname> terminal debugging
<Sysname> debugging ddns
```

# Configure a DDNS policy named 3322.org, and apply the policy to Ethernet 1/1.

```bash
```
<Sysname> system-view
[Sysname] ddns policy 3322.org
[Sysname-ddns-policy-3322.org] url

Output similar to the following messages is generated:
http://steven:nevets@members.3322.org/dyndns/update?system=dyndns&hostname=<h>&myip=<a>
[Sysname-ddns-policy-3322.org] interval 0 0 15
[Sysname-ddns-policy-3322.org] quit
[Sysname] interface ethernet 1/1
[Sysname-Ethernet1/1] ddns apply policy 3322.org fqdn whatever.3322.org

// The device translated the domain name of the DDNS server into IP address 222.185.245.239.
*Jun 12 12:38:45:453 2008 Sysname DDNS/7/Debug:
  DDNS event (Interface: Ethernet1/1, policy: 3322.org): Waiting for reply from DNS server.
*Jun 12 12:38:45:672 2008 Sysname DDNS/7/Debug:
  DDNS event (Interface: Ethernet1/1, policy: 3322.org): Get server IP address 222.185.245.239.

// The device connected to the DDNS server to initiate the first DDNS update.
*Jun 12 12:38:45:672 2008 Sysname DDNS/7/Debug:
  DDNS event (Interface: Ethernet1/1, policy: 3322.org): Starting DDNS update for the 1st time.

// The device send the DDNS update request.
*Jun 12 12:38:47:906 2008 Sysname DDNS/7/Debug:
  DDNS server reply (Interface: Ethernet1/1, policy: 3322.org): good 60.192.123.32

// The device received a reply from the DDNS server and updated the DNS entry.
*Jun 12 12:38:47:906 2008 Sysname DDNS/7/Debug:
  DDNS event (Interface: Ethernet1/1, policy: 3322.org): DDNS update stopped.

// The DDNS update completed.

# Configure a DDNS policy named oray.cn and apply the policy to Ethernet 1/1.
<Sysname> system-view
[Sysname] ddns policy oray.cn
[Sysname-ddns-policy-oray.cn] url http://steven:nevets@phservice2.oray.net
[Sysname-ddns-policy-oray.cn] interval 0 0 12
[Sysname-ddns-policy-oray.cn] quit
[Sysname] interface ethernet 1/1
[Sysname-Ethernet1/1] ddns apply policy oray.cn fqdn whatever.gicp.cn

Output similar to the following messages is generated:
*Jun 12 17:52:04:390 2008 Sysname DDNS/7/Debug:
  DDNS event (Interface: Ethernet1/1, policy: oray.cn): Waiting for reply from DNS server.
*Jun 12 17:52:04:690 2008 Sysname DDNS/7/Debug:
  DDNS event (Interface: Ethernet1/1, policy: oray.cn): Get server IP address 61.152.96.118.

// The device translated the domain name of the PeanutHull server into IP address 61.152.96.118.
*Jun 12 17:52:04:690 2008 Sysname DDNS/7/Debug:
DDNS event (Interface: Ethernet1/1, policy: oray.cn): Starting DDNS update for the 1st time.

// The device was trying to connect to the DDNS server to initiate the first DDNS update.
*Jun 12 17:52:04:733 2008 Sysname DDNS/7/Debug:
  DDNS server reply (Interface: Ethernet1/1, policy: oray.cn):
  220 Oray.cn DHRP server ready

// The device connected to the DDNS server and received the welcome information.
*Jun 12 17:52:04:733 2008 Sysname DDNS/7/Debug:
  DDNS event (Interface: Ethernet1/1, policy: oray.cn): Authentication mode was sent.
*Jun 12 17:52:04:812 2008 Sysname DDNS/7/Debug:
  DDNS server reply (Interface: Ethernet1/1, policy: oray.cn):
  334 3doxTtwQ0EDBAAh1R3WMGEQ==

// The authentication mode was successfully negotiated. The DDNS server replies with the BASE64 authentication code string.
*Jun 12 17:52:04:812 2008 Sysname DDNS/7/Debug:
  DDNS event (Interface: Ethernet1/1, policy: oray.cn): Authentication request was sent.
*Jun 12 17:52:04:886 2008 Sysname DDNS/7/Debug:
  DDNS server reply (Interface: Ethernet1/1, policy: oray.cn):
  250 Authentication passed
  whenever.gicp.net
  whatever.gicp.net

// User authentication succeeded. The DDNS server replies with the registered domain name list.
*Jun 12 17:52:04:886 2008 Sysname DDNS/7/Debug:
  DDNS event (Interface: Ethernet1/1, policy: oray.cn): Domain name was sent.
*Jun 12 17:52:04:939 2008 Sysname DDNS/7/Debug:
  DDNS server reply (Interface: Ethernet1/1, policy: oray.cn):
  250 Register successfully

// The DDNS server accepted the update request.
*Jun 12 17:52:04:939 2008 Sysname DDNS/7/Debug:
  DDNS event (Interface: Ethernet1/1, policy: oray.cn): DDNS update stopped.

// The DDNS update completed.
debugging dvpn

Use debugging dvpn to enable DVPN debugging.
Use undo debugging dvpn to disable DVPN debugging.

Syntax

dehugging dvpn { all | error | event | packet }
undo debugging dvpn { all | error | event | packet }

Default
DVPN debugging is disabled.

Views
User view

Default command level
1: Monitor level

Parameters

all: All types of DVPN debugging.
error: DVPN error debugging.
event: DVPN event debugging.
packet: DVPN packet debugging.

Examples

# Enable DVPN error debugging. When the device is enabled with DVPN and OSPF but has no tunnels,
output similar to the following example is generated:
<MainServer> debugging dvpn error
*Jan 6 14:18:03:246 2007 MainServer DVPN/7/PACKET:
Failed to get DVPN session.

# Enable DVPN event debugging on a Spoke. When the reset dvpn session all command is executed,
output similar to the following example is generated:
<Spoke1> debugging dvpn event
*Jul 28 16:44:07:969 2010 Spoke1 DVPN/7/EVENT:
Sent message (type 3) to VAM.

  // DVPN sent a message of type 3 to the VAM.
*Jul 28 16:44:07:969 2010 Spoke1 DVPN/7/EVENT:
Received message 4 from VAM.

  // DVPN received a message of type 4 from the VAM.
*Jul 28 16:44:07:985 2010 Spoke1 DVPN/7/EVENT:
Created session to peer 10.0.1.1.

  // DVPN created a session to peer 10.0.1.1.

423
Initialized session 10.0.1.1.

// DVPN initialized the session to peer 10.0.1.1.

Created KEEPALIVE timer for session 10.0.1.1.

// DVPN started a keepalive timer for session 10.0.1.1.

# Enable DVPN packet debugging. When a DVPN UDP tunnel is established, output similar to the following example is generated:
<Spoke1> debugging dvpn packet

After encapsulation: from 202.10.19.74 to 202.10.19.73.

// DVPN sent a packet from 202.10.19.74 to 202.10.19.73.

// DVPN sent a keepalive packet to the device whose private address is 10.0.1.1.

Received a DVPN UDP packet.

// DVPN received a DVPN UDP packet.

Received a packet (type 3) from peer 10.0.1.1.

// DVPN received a packet of type 3 from 10.0.1.1.

# Enable DVPN packet debugging. When a DVPN GRE tunnel is established, output similar to the following example is generated:
<Hub-1> debugging dvpn packet


// The DVPN GRE packet before de-encapsulation is from 202.10.19.73 to 202.10.19.71.

Received a packet (type 0) from peer 10.0.1.3.

// DVPN received a packet of type 0 from 10.0.1.3.

After encapsulation: from 202.10.19.71 to 202.10.19.73.

// DVPN sent a packet from 202.10.19.71 to 202.10.19.73.

Sent KEEPALIVE response to peer 10.0.1.3.

// DVPN sent a keepalive response to the private address 10.0.1.3.

Sending a DVPN packet.

// DVPN sent a DVPN packet.
Before encapsulation: from 192.168.1.1 to 192.168.3.1.

// The data packet before DVPN encapsulation is from 192.168.1.1 to 192.168.3.1.

*Jul 29 15:29:53:547 2010 Hub-1 DVPN/7/PACKET:

After encapsulation: from 202.10.19.71 to 202.10.19.73.

// The data packet after DVPN encapsulation is from 202.10.19.71 to 202.10.19.73.

*Jul 29 15:29:53:547 2010 Hub-1 DVPN/7/PACKET:

Received a DVPN GRE packet.

// DVPN received a DVPN packet encapsulated in GRE.

*Jul 29 15:29:53:547 2010 Hub-1 DVPN/7/PACKET:


// The data packet before de-encapsulation is from 202.10.19.73 to 202.10.19.71.

*Jul 29 15:29:53:563 2010 Hub-1 DVPN/7/PACKET:

After decapsulation: from 192.168.3.1 to 192.168.1.1.

// The data packet after de-encapsulation is from 192.168.3.1 to 192.168.1.1.

debugging vam client

Use **debugging vam client** to enable VAM client debugging.

Use **undo debugging vam client** to disable VAM client debugging.

Syntax

```
debugging vam client { all | error | event | packet }
undo debugging vam client { all | error | event | packet }
```

Default

VAM client debugging is disabled.

Views

User view

Default command level

1: Monitor level

Parameters

- **all**: All types of VAM client debugging.
- **error**: VAM client error debugging.
- **event**: VAM client event debugging.
- **packet**: VAM client control packet debugging.

Usage guidelines

Table 226 describes output fields and messages for the **debugging vam client packet** command.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Received a packet (length bytes) from IP ip-address port port VRF Index vrf-index.</td>
<td>The vrf-index argument specifies the index of a VPN instance. A value of 0 for the vrf-index indicates the public network.</td>
</tr>
</tbody>
</table>
Sent a packet (length bytes) to IP ip-address port port VRF Index vrf-index successfully.  

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>vrf-index</td>
<td>The vrf-index argument indicates the index of a VPN instance. A value of 0 for the vrf-index indicates the public network.</td>
</tr>
</tbody>
</table>

**Examples**

# Enable VAM client error debugging on the VAM client. When the `shutdown` command is executed on a tunnel interface, output similar to the following example is generated:

```
<Client> debugging vam client error
<Client> system-view
[Client] interface tunnel 1
[Client-Tunnel1] shutdown
*Jan  6 11:55:48:342 2007 Client VAM/7/ERROR:VAM_Client:  
The interface state is not ready.
```

# Enable VAM event debugging for the VAM client. After the client registers with the VAM server, output similar to the following example is generated:

```
*Jan  6 11:46:11:824 2007 Client VAM/7/EVENT:VAM_Client:  
Sent KEEPALIVE request successfully.
*Jan  6 11:46:11:834 2007 Client VAM/7/EVENT:VAM_Client:  
Received a packet of type 13.
*Jan  6 11:46:11:834 2007 Client VAM/7/EVENT:VAM_Client:  
Received KEEPALIVE response.
```

# Enable VAM packet debugging for the VAM client. After the client registers with the VAM server, output similar to the following example is generated:

```
*Jan  6 11:40:29:830 2007 Client VAM/7/PACKET:VAM_Client:  
Sent a packet (38 bytes) to IP 28.1.1.21 port 40959 VRF Index 0 successfully.
```

**debugging vam server**

Use `debugging vam server` to enable VAM server debugging.

Use `undo debugging vam server` to disable VAM server debugging.

**Syntax**

```
depugging vam server { all | error | event | packet }
```

**Default**

VAM server debugging is disabled.
Views

User view

Default command level

1: Monitor level

Parameters

all: All types of VAM server debugging.
error: VAM server error debugging.
event: VAM server event debugging.
packet: VAM server packet debugging.

Usage guidelines

Table 227 describes output fields and messages for the \texttt{debugging vam server event} command.

Table 227 Output from the debugging vam server event command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Client address1 is replaced: VRF vrf-index1 address1:port1 --> VRF vrf-index2 address2:port2 | The client whose VPN instance index is:  
  - vrf-index1.  
  - IP address is address1.  
  - Port number is port1  
  is replaced by the client whose VPN instance index is:  
  - vrf-index2.  
  - IP address is address2.  
  - Port number is port2.  
  A value of 0 for vrf-index indicates the public network. |

Table 228 describes output fields and messages for the \texttt{debugging vam server packet} command.

Table 228 Output from the debugging vam server packet command

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Received a packet (length bytes) from IP ip-address port port VRF Index vrf-index.</td>
<td>The vrf-index argument specifies the index of a VPN instance. A value of 0 for vrf-index indicates the public network.</td>
</tr>
<tr>
<td>Sent a packet (length bytes) to IP ip-address port port VRF Index vrf-index successfully.</td>
<td>The vrf-index argument specifies the index of a VPN instance. A value of 0 for vrf-index indicates the public network.</td>
</tr>
</tbody>
</table>

Examples

# Enable VAM server error debugging. When a client that has a different key registers with the server, output similar to the following example is generated:

```
<MainServer> debugging vam server error
*Jul 28 16:13:27:375 2010 Mainserver VAM/7/ERROR: VAM_Server:  
  Failed to authenticate the packet.
```

# Enable VAM server event debugging. When a client registers with the server, output similar to the following example is generated:
<Mainserver> debugging vam server event
*Jul 28 15:34:22:312 2010 Mainserver VAM/7/EVENT: VAM_Server:
  Created client aging timer successfully.
*Jul 28 15:34:22:312 2010 Mainserver VAM/7/EVENT: VAM_Server:
  Received a packet of type 1.
*Jul 28 15:34:22:312 2010 Mainserver VAM/7/EVENT: VAM_Server:
  Received connect request successfully.
*Jul 28 15:34:22:312 2010 Mainserver VAM/7/EVENT: VAM_Server:
  Received server FSM event of type 1.
*Jul 28 15:34:22:312 2010 Mainserver VAM/7/EVENT: VAM_Server:
  State changed on Server.
*Jul 28 15:34:22:312 2010 Mainserver VAM/7/EVENT: VAM_Server:
  Sent connect response successfully.
*Jul 28 15:34:22:344 2010 Mainserver VAM/7/EVENT: VAM_Server:
  Received a packet of type 3.
*Jul 28 15:34:22:344 2010 Mainserver VAM/7/EVENT: VAM_Server:
  Received connect finish request successfully.
*Jul 28 15:34:22:359 2010 Mainserver VAM/7/EVENT: VAM_Server:
  Received server FSM event of type 2.
*Jul 28 15:34:22:359 2010 Mainserver VAM/7/EVENT: VAM_Server:
  State changed on Server.
*Jul 28 15:34:22:375 2010 Mainserver VAM/7/EVENT: VAM_Server:
  Received a packet of type 5.
*Jul 28 15:34:22:391 2010 Mainserver VAM/7/EVENT: VAM_Server:
  Received register request successfully.
*Jul 28 15:34:22:406 2010 Mainserver VAM/7/EVENT: VAM_Server:
  Received server FSM event of type 3.
*Jul 28 15:34:22:406 2010 Mainserver VAM/7/EVENT: VAM_Server:
  The state of client 202.10.19.73 in VPN 1 changed from INIT to REG.
*Jul 28 15:34:22:422 2010 Mainserver VAM/7/EVENT: VAM_Server:
  State changed on Server.
*Jul 28 15:34:22:422 2010 Mainserver VAM/7/EVENT: VAM_Server:
  Sent authentication request successfully.
*Jul 28 15:34:22:437 2010 Mainserver VAM/7/EVENT: VAM_Server:
  Received a packet of type 8.
*Jul 28 15:34:22:437 2010 Mainserver VAM/7/EVENT: VAM_Server:
  Received authentication information request successfully.
*Jul 28 15:34:22:453 2010 Mainserver VAM/7/EVENT: VAM_Server:
  Received server FSM event of type 4.
*Jul 28 15:34:22:453 2010 Mainserver VAM/7/EVENT: VAM_Server:
  State changed on Server.
*Jul 28 15:34:22:469 2010 Mainserver VAM/7/EVENT: VAM_Server:
  Sent authentication request to AAA module successfully.
*Jul 28 15:34:22:469 2010 Mainserver VAM/7/EVENT: VAM_Server:
  Created AAA timer successfully.
*Jul 28 15:34:22:484 2010 Mainserver VAM/7/EVENT: VAM_Server:
Received authentication success message from AAA.
*Jul 28 15:34:22:484 2010 Mainserver VAM/7/EVENT: VAM_Server: 
  Received server FSM event of type 9.
*Jul 28 15:34:22:484 2010 Mainserver VAM/7/EVENT: VAM_Server: 
  State changed on Server.
*Jul 28 15:34:22:500 2010 Mainserver VAM/7/EVENT: VAM_Server: 
  Sent account request to AAA module successfully.
*Jul 28 15:34:22:500 2010 Mainserver VAM/7/EVENT: VAM_Server: 
  Received accounting success message from AAA.
*Jul 28 15:34:22:516 2010 Mainserver VAM/7/EVENT: VAM_Server: 
  Received server FSM event of type 11.
*Jul 28 15:34:22:516 2010 Mainserver VAM/7/EVENT: VAM_Server: 
  State changed on Server.
*Jul 28 15:34:22:531 2010 Mainserver VAM/7/EVENT: VAM_Server: 
  Sent register response successfully.
*Jul 28 15:34:22:531 2010 Mainserver VAM/7/EVENT: VAM_Server: 
  Received a packet of type 13.
*Jul 28 15:34:22:547 2010 Mainserver VAM/7/EVENT: VAM_Server: 
  Received server FSM event of type 7.
*Jul 28 15:34:22:547 2010 Mainserver VAM/7/EVENT: VAM_Server: 
  Refreshed register aging timer successfully.

// Set the register aging timer.
*Jul 28 15:34:22:562 2010 Mainserver VAM/7/EVENT: VAM_Server: 
  Added address map successfully.
*Jul 28 15:34:22:562 2010 Mainserver VAM/7/EVENT: VAM_Server: 
  The state of client 202.10.19.73 in VPN 1 changed from REG to ONLINE.
*Jul 28 15:34:22:578 2010 Mainserver VAM/7/EVENT: VAM_Server: 
  Received first KEEPALIVE request.
*Jul 28 15:34:22:578 2010 Mainserver VAM/7/EVENT: VAM_Server: 
  State changed on Server.
*Jul 28 15:34:22:594 2010 Mainserver VAM/7/EVENT: VAM_Server: 
  Sent KEEPALIVE response successfully.

# Enable VAM server packet debugging. When a client registers with the server, output similar to the following example is generated:
</MainServer> debugging vam server packet
*Jan 6 10:59:28:612 2007 MainServer VAM/7/PACKET:VAM_Server: 
  Received a packet (76 bytes) from IP 28.1.1.21 port 1027 VRF Index 0.

// The VAM server received a 76-byte packet from port 1027 of the device whose IP is 28.1.1.21 in the public network.
*Jan 6 10:59:28:612 2007 MainServer VAM/7/PACKET:VAM_Server: 
  Sent a packet (38 bytes) to IP 28.1.1.21 port 1027 VRF Index 0 successfully.

// The VAM server sent a 38-byte packet to port 1027 of the device whose IP is 28.1.1.21 in the public network.