

debug tag-switching adjacency

To display changes to Tag Switching entries in the adjacency database, use the **debug tag-switching adjacency** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug tag-switching adjacency

no debug tag-switching adjacency

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

You can use the **debug tag-switching adjacency** command to monitor those instances when entries are updated or added to the adjacency.

Examples

The following is sample output from the **debug tag-switching adjacency** command:

```
Router# debug tag-switching adjacency

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

[Table 236](#) describes the significant fields shown in the display.

Table 236 *debug tag-switching adjacency* Field Description

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

Related Commands

Command	Description
show adjacency	Displays CEF adjacency table information.

debug tag-switching atm-cos

To display ATM label-VC bind or request activity based on the configuration of a CoS map, use the **debug tag-switching atm-cos** ATM command in privileged EXEC mode.

```
debug tag-switching atm-cos [bind | request]
```

Syntax Description	bind	Specifies debug information about bind responses for a VC path.
	request	Specifies debug information about bind requests for a VC path.

Command Modes	Privileged EXEC
---------------	-----------------

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

Examples

The following is sample output from the **debug tag-switching atm-cos** command.

```
Router# show tag forwarding
```

```
Local Outgoing Prefix      Bytes tag Outgoing Next Hop
tag  tag or VC  or Tunnel Id  switched interface
26  28      17.17.17.17/32  0      PO6/0   point2point
27  Pop tag   11.11.11.11/32  1560   PO6/0   point2point
28  27      16.16.16.16/32  0      PO6/0   point2point
29  30      92.0.0.0/8      0      PO6/0   point2point
30  Pop tag   95.0.0.0/8      2600   PO6/0   point2point
31  2/34    10.10.10.10/32  0      AT2/0.1 point2point
32  Pop tag   14.14.14.14/32  0      Fa5/0   91.0.0.1
33  Pop tag   90.0.0.0/8      0      Fa5/0   91.0.0.1
34  Pop tag   96.0.0.0/8      0      Fa5/0   91.0.0.1
    2/36    96.0.0.0/8      0      AT2/0.1 point2point
35  35      93.0.0.0/8      0      PO6/0   point2point
36  36      12.12.12.12/32  0      PO6/0   point2point
37  37      15.15.15.15/32  0      PO6/0   point2point
38  37      18.18.18.18/32  0      Fa5/0   91.0.0.1
39  39      97.0.0.0/8      540    PO6/0   point2point
40  40      98.0.0.0/8      0      PO6/0   point2point
```

```
Router# debug tag atm-c
```

```
Router# debug tag atm-cos ?
```

```
bind   Bind response for VC path
request Requests for VC binds path
```

```
Router# debug tag atm-cos bind
```

```
ATM TAGCOS Bind response debugging is on
```

```
Router# debug tag atm-cos request
```

```
ATM TAGCOS VC requests debugging is on
```

```

Router# configure terminal

Enter configuration commands, one per line. End with CNTL/Z.
Router(config)# interface a2/0.1
Router(config-subif)# tag atm multi
Router(config-subif)# end
Router#
19:59:14:%SYS-5-CONFIG_I:Configured from console by console
Router#
19:59:24:TAGCOS-REQ:vc request 10.10.10.10/32, available
19:59:24:TAGCOS-REQ:vc request 10.10.10.10/32, standard
19:59:24:TAGCOS-REQ:vc request 10.10.10.10/32, premium
19:59:24:TAGCOS-REQ:vc request 10.10.10.10/32, control
19:59:24:TAGCOS-REQ:vc request 96.0.0.0/8, available
19:59:24:TAGCOS-REQ:vc request 96.0.0.0/8, standard
19:59:24:TAGCOS-REQ:vc request 96.0.0.0/8, premium
19:59:24:TAGCOS-REQ:vc request 96.0.0.0/8, control
TAGCOS-REQ/TCATM:11.11.11.11/32, len=4352, band=1099528405504, class=0x700
TAGCOS-REQ/TCATM:12.12.12.12/32, len=4352, band=2199040033280, class=0x700
TAGCOS-REQ/TCATM:13.13.13.13/32, len=4352, band=3298551661056, class=0x700
TAGCOS-REQ/TCATM:14.14.14.14/32, len=4352, band=4398063288832, class=0x700
TAGCOS-REQ/TCATM:15.15.15.15/32, len=4352, band=5497574916608, class=0x700
TAGCOS-REQ/TCATM:16.16.16.16/32, len=4352, band=6597086544384, class=0x700
TAGCOS-REQ/TCATM:17.17.17.17/32, len=4352, band=7696598172160, class=0x700
TAGCOS-REQ/TCATM:18.18.18.18/32, len=4352, band=8796109799936, class=0x700
TAGCOS-REQ/TCATM:90.0.0.0/8, len=768, band=3940649674539009, class=0x2
TAGCOS-REQ/TCATM:91.0.0.0/8, len=768, band=3940649674604545, class=0x2
TAGCOS-REQ/TCATM:92.0.0.0/8, len=768, band=3940649674670081, class=0x2
TAGCOS-REQ/TCATM:93.0.0.0/8, len=768, band=3940649674735617, class=0x2
TAGCOS-REQ/TCATM:94.0.0.0/8, len=768, band=3940649674801153, class=0x2
TAGCOS-REQ/TCATM:95.0.0.0/8, len=768, band=3940649674866689, class=0x2
TAGCOS-REQ/TCATM:97.0.0.0/8, len=768, band=3940649674932225, class=0x2
TAGCOS-REQ/TCATM:98.0.0.0/8, len=768, band=3940649674997761, class=0x2
TAGCOS-BIND:binding_ok 10.10.10.10/32,VCD=41 - control 41,41,41,41
TAGCOS-BIND:binding_ok 10.10.10.10/32, Inform TFIB pidx=0, in_tag=31, idx=0x80000000
TAGCOS-BIND:binding_ok 96.0.0.0/8,VCD=42 - control 42,42,42,42
TAGCOS-BIND:binding_ok 96.0.0.0/8, Inform TFIB pidx=1, in_tag=34, idx=0x80000001
TAGCOS-BIND:binding_ok 10.10.10.10/32,VCD=43 - premium 43,43,43,41
TAGCOS-BIND:binding_ok 96.0.0.0/8,VCD=44 - premium 44,44,44,42
TAGCOS-BIND:binding_ok 10.10.10.10/32,VCD=45 - standard 45,45,43,41
TAGCOS-BIND:binding_ok 96.0.0.0/8,VCD=46 - standard 46,46,44,42
TAGCOS-BIND:binding_ok 10.10.10.10/32,VCD=47 - available 47,45,43,41
TAGCOS-BIND:binding_ok 96.0.0.0/8,VCD=48 - available 48,46,44,42
72k-41-5#
72k-41-5#

```

Related Commands

Command	Description
debug tag atm-tdp	Debugs label-controlled ATM TDP.
debug tag packets	Debugs tag switching packets.
debug tag tdp	Debugs tag distribution protocol items and information.

debug tag-switching atm-tdp api

To display information about the VCI allocation of tag VCs (TVCs), free, and cross-connect requests, use the **debug tag-switching atm-tdp api** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug tag-switching atm-tdp api

no debug tag-switching atm-tdp api

Syntax Description This command has no arguments or keywords.

Command Modes Privileged EXEC

Usage Guidelines You can use the **debug tag-switching atm-tdp api** command with the **debug tag-switching atm-tdp states** command to display more complete information about a TVC.

Examples The following is sample output from the **debug tag-switching atm-tdp api** command:

```
Router# debug tag-switching atm-tdp api

Tailend Router Free tag Req 167.50.0.0 on ATM0/0.2 VPI/VCI 1/674
TAGATM_API: received tag free request
    interface: ATM0/0.2 dir: in vpi: 1 vci: 674
TAGATM_API: completed tag free
    interface: ATM0/0.2 vpi: 1 vci: 674
    result: TAGATM_OK
```

[Table 237](#) describes the significant fields shown in the display.

Table 237 *debug tag-switching atm-tdp api* Field Descriptions

Field	Description
TAGATM_API	Subsystem that prints the message.
interface	Interface used by the driver to allocate or free VPI/VCI resources.
dir	Direction of the VC: <ul style="list-style-type: none"> In—Input or receive VC Out—Output VC
vpi	Virtual path identifier.
vci	Virtual channel identifier.
result	Return error code from the driver API.

Related Commands

Command	Description
debug tag-switching atm-tdp states	Displays information about TVC state transitions as they occur.

debug tag-switching atm-tdp routes

To display information about the state of the routes for which VCI requests are being made, use the **debug tag-switching atm-tdp routes** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug tag-switching atm-tdp routes

no debug tag-switching atm-tdp routes

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

When there are many routes and system activities (that is, shutting down interfaces, learning of new routes, and so on), the **debug tag-switching atm-tdp routes** command displays a substantial amount of information that may interfere with system timing. Most commonly, this affects the normal operation of the Tag Distribution Protocol (TDP). You should increase the TDP hold-time value by using the **tag-switching tdp holdtime** command.

Examples

The following is sample output from the **debug tag-switching atm-tdp routes** command:

```
Router# debug tag-switching atm-tdp routes

CleanupRoutes,not deleting route of idb ATM0/0.2,rdbIndex 0
tcatmFindRouteTags,153.7.0.0/16,idb=ATM0/0.2,nh=134.111.102.98,index=0
AddNewRoute,153.7.0.0/16,idb=ATM0/0.2
CleanupRoutes,153.7.0.0/16
CleanupRoutes,not deleting route of idb ATM0/0.2,rdbIndex 0
tcatmFindRouteTags,153.8.0.0/16,idb=ATM0/0.2,nh=134.111.102.98,index=0
AddNewRoute,153.8.0.0/16,idb=ATM0/0.2
CleanupRoutes,153.8.0.0/16
CleanupRoutes,not deleting route of idb ATM0/0.2,rdbIndex 0
tcatmFindRouteTags,153.9.0.0/16,idb=ATM0/0.2,nh=134.111.102.98,index=0
AddNewRoute,153.9.0.0/16,idb=ATM0/0.2
CleanupRoutes,153.9.0.0/16
CleanupRoutes,not deleting route of idb ATM0/0.2,rdbIndex 0
tcatmFindRouteTags,153.10.0.0/16,idb=ATM0/0.2,nh=134.111.102.98,index=0
AddNewRoute,153.10.0.0/16,idb=ATM0/0.2
CleanupRoutes,153.10.0.0/16
CleanupRoutes,not deleting route of idb ATM0/0.2,rdbIndex 0
tcatmFindRouteTags,153.11.0.0/16,idb=ATM0/0.2,nh=134.111.102.98,index=0
AddNewRoute,153.11.0.0/16,idb=ATM0/0.2
CleanupRoutes,153.11.0.0/16
```

Table 238 describes the significant fields shown in the display.

Table 238 *debug tag-switching atm-tdp routes Field Descriptions*

Field	Description
CleanupRoutes	Cleans up the routing table after a route has been deleted.
not deleting route of idb ATM0/0.2	Route cleanup event has not removed the specified route.
rdbIndex	Index identifying the route.
tcatmFindRouteTags	Request a VC for the route.
idb	Internal descriptor for an interface.
nh	Next hop for the route.
index	Identifier for the route.
AddNewRoute	Action of adding routes for a prefix or address.

debug tag-switching atm-tdp states

To display information about TVC state transitions as they occur, use the **debug tag-switching atm-tdp states** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug tag-switching atm-tdp states

no debug tag-switching atm-tdp states

Syntax Description This command has no arguments or keywords.

Command Modes Privileged EXEC

Usage Guidelines When there are many routes and system activities (that is, shutting down interfaces, learning of new routes, and so on), the **debug tag-switching atm-tdp states** command outputs a substantial amount of information that may interfere with system timing. Most commonly, this affects the normal operation of the Tag Distribution Protocol (TDP). You should increase the TDP hold-time value by using the **tag-switching tdp holdtime** command.

Examples The following is sample output from the **debug tag-switching atm-tdp states** command:

```
Router# debug tag-switching atm-tdp states

Transit Output 166.35.0.0 VPI/VCI 1/67 Active -> XmitRelease NoPath
Transit Input 166.35.0.0 VPI/VCI 1/466 Active -> ApiWaitParentLoss ParentLoss
Transit Input 166.35.0.0 VPI/VCI 1/466 ApiWaitParentLoss -> ParentWait ApiSuccess
Transit Input 166.35.0.0 VPI/VCI 1/466 ParentWait -> XmitWithdraw NoPath
Transit Input 166.35.0.0 VPI/VCI 1/466 XmitWithdraw -> XmitWithdraw Transmit
Transit Input 166.35.0.0 VPI/VCI 1/466 XmitWithdraw -> NonExistent Release
Transit Input 166.35.0.0 VPI/VCI 1/466 NonExistent -> NonExistent ApiSuccess
```

[Table 239](#) describes the significant fields shown in the display.

Table 239 *debug tag-switching atm-tdp states* Field Descriptions

Field	Description
Transit Output	Output side of a TVC.
VPI/VCI	VC value.
Transit Input	Input side of a TVC.

debug tag-switching packets

To display tagged packets switched by this router, use the **debug tag-switching packets** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug tag-switching packets [*interface*]

no debug tag-switching packets [*interface*]

Syntax Description

interface (Optional) Interface or subinterface name.

Command Modes

Privileged EXEC

Usage Guidelines

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



Caution

Use this command with care, because it generates output for every packet processed. Furthermore, enabling this command causes fast and distributed Tag Switching to be disabled for the selected interfaces. Use this command only when traffic on the network is low, so other activity on the system is not adversely affected.

Examples

The following is sample output from the **debug tag-switching packets** command:

```
Router# debug tag-switching packets

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

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

[Table 240](#) describes the significant fields shown in the display.

Table 240 debug tag-switching packets Field Descriptions

Field	Description
Hs0/0	Identifier for the interface on which the packet was received or sent.
recvd	Packet received.
xmit	Packet sent.
CoS	Class of Service field from the packet tag header.
TTL	Time to live (TTL) field from the packet tag header.
(no tag)	Last tag popped off the packet and sent untagged.
Tag(s)	List of tags on the packet, ordered from top of stack to bottom.

debug tag-switching tdp advertisements

To print information about the advertisement of tags and interface addresses to TDP peer devices, use the **debug tag-switching tdp advertisements** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug tag-switching tdp advertisements

no debug tag-switching tdp advertisements

Syntax Description This command has no arguments or keywords.

Command Modes Privileged EXEC

Examples The following is sample output from the **debug tag-switching tdp advertisements** command:

```
Router# debug tag-switching tdp advertisements

tagcon: adj 210.9.0.9:0 (pp 0x60D8E98C): advertise 99.101.0.8
tagcon: adj 210.9.0.9:0 (pp 0x60D8E98C): advertise 172.27.32.28
tagcon: adj 210.9.0.9:0 (pp 0x60D8E98C): advertise 10.105.0.8
tagcon: adj 210.9.0.9:0 (pp 0x60D8E98C): advertise 10.92.0.8
tagcon: adj 210.9.0.9:0 (pp 0x60D8E98C): advertise 10.205.0.8
tagcon: adj 210.9.0.9:0 (pp 0x60D8E98C): advertise 210.8.0.8
tagcon: adj 210.9.0.9:0 (pp 0x60D8E98C): advertise 10.105.0.0/16, tag 1 (#2)
tagcon: adj 210.9.0.9:0 (pp 0x60D8E98C): advertise 10.102.0.0/16, tag 26 (#4)
tagcon: adj 210.9.0.9:0 (pp 0x60D8E98C): advertise 10.227.0.0/16, tag 27 (#6)
```

Table 241 describes the significant fields shown in the display.

Table 241 debug tag-switching tdp advertisements Field Descriptions

Field	Description
tagcon:	Identifies the source of the message as the tag control subsystem.
adj <a.b.c.d:e>	TDP identifier of the peer device to which the advertisement has been made.
(pp 0xn timer)	Identifier for the data structure used to represent the peer device at the tag distribution level. Useful for correlating debug output.
advertise X	What was advertised to the peer device—either an interface address (“a.b.c.d”) or tag binding (“a.b.c.d/m, tag t (#n”).
(#n)	For a tag binding advertisement, the sequence number of the tag information base (TIB) modification that made it necessary to advertise the tag.

Related Commands	Command	Description
	show tag-switching tdp neighbors	Displays the status of TDP sessions.

debug tag-switching tdp bindings

To print information about changes to the tag information base (TIB) used to keep track of tag bindings learned from TDP peer devices through TDP downstream tag distribution, use the **debug tag-switching tdp bindings** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug tag-switching tdp bindings

no debug tag-switching tdp bindings

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Examples

The following is sample output from the **debug tag-switching tdp bindings** command:

```
Router# debug tag-switching tdp bindings

tagcon: tibent(10.105.0.0/16): created; find route tags request
tagcon: tibent(10.105.0.0/16): lcl tag 1 (#2) assigned
tagcon: tibent(10.102.0.0/16): created; find route tags request
tagcon: tibent(10.102.0.0/16): lcl tag 26 (#4) assigned
tagcon: 210.9.0.9:0: 99.101.0.9 added to addr<->tdp ident map
tagcon: 210.9.0.9:0: 172.27.32.29 added to addr<->tdp ident map
tagcon: 210.9.0.9:0: 10.105.0.9 added to addr<->tdp ident map
tagcon: tibent(172.27.32.0/22): rem tag 1 from 210.9.0.9:0 added
tagcon: tibent(200.26.0.0/16): rem tag 30 from 210.9.0.9:0 added
tagcon: tibent(210.8.0.8/32): created; remote tag learned
tagcon: tibent(210.8.0.8/32): rem tag 31 from 210.9.0.9:0 added
```

[Table 242](#) describes the significant fields shown in the display.

Table 242 debug tag-switching tdp bindings Field Descriptions

Field	Description
tagcon:	Identifies the source of the message as the tag control subsystem.
tibent(<i>network/mask</i>)	Destination that has a tag binding change.
created; <i>reason</i>	TIB entry has been created for the specified destination for the indicated reason.
rem tag ...	Describes a change to the tag bindings for the specified destination. The change is for a tag binding learned from the specified TDP peer device.
lcl tag ...	Describes a change to a locally assigned (incoming) tag for the specified destination.
(#n)	Sequence number of the modification to the TIB corresponding to the local tag change.
<i>a.b.c.d:n: e.f.g.h</i> added to addr<->tdp ident map	Address <i>e.f.g.h</i> has been added to the set of addresses associated with TDP identifier <i>a.b.c.d:n</i> .

■ debug tag-switching tdp bindings

Related Commands	Command	Description
	show tag-switching tdp bindings	Displays the contents of the TIB.

debug tag-switching tdp directed-neighbors

To print information about the directed neighbor mechanism, use the **debug tag-switching tdp directed-neighbors** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug tag-switching tdp directed-neighbors

no debug tag-switching tdp directed-neighbors

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

This mechanism establishes TDP adjacencies to peer devices that are not directly adjacent, such as peer devices at either end of a tunnel.

The directed neighbor mechanism starts TDP discovery between two TSRs that are not necessarily directly adjacent. This mechanism is used, for instance, to support two-level tagging across a TSP tunnel, and to support traffic engineering metric exchange across a TSP tunnel.

The mechanism is based on an IP address, such as the IP address of the last hop of a TSP tunnel. A TSR wanting to establish a TDP adjacency to some other TSR with a given IP address is the active TSR for that directed neighbor discovery. A TSR willing to respond to that discovery is the passive TSR for that discovery.

As with TDP discovery between adjacent TSRs, it is possible to have multiple directed neighbor discovery sessions can run between two TSRs, all supporting a single TDP adjacency.

The debug messages track discovery changes, such as discovery or loss of a directed neighbor. As a detail reflected in the debug prints, discovery of a directed neighbor with IP address X is complete when a TDP adjacency comes up and the far end announces that IP address X is one of its IP addresses.

Examples

The following is sample output from the **debug tag-switching tdp directed-neighbors** command:

```
Router# debug tag-switching tdp directed-neighbors

tdp_directednbr: TDPDirAdj 10.11.10.11 received address addition notification
tdp_directednbr: TDPDirAdj 10.11.10.11 TDP peer set
tdp_directednbr: TDPDirAdj 10.11.10.11 received address deletion notification
tdp_directednbr: TDPDirAdj 10.11.10.11 peer cleared
```

[Table 243](#) describes the significant fields shown in the display.

Table 243 *debug tag-switching tdp directed-neighbors Field Descriptions*

Field	Description
tdp_directednbr:	Identifies this as a TDP directed neighbor debug statement.
TDPDirAdj <address>	Identifies the IP address to which a TDP adjacency is desired.

■ `debug tag-switching tdp directed-neighbors`

Related Commandss	Command	Description
	<code>show tag-switching tdp neighbors</code>	Displays the status of TDP sessions.

debug tag-switching tdp peer state-machine

To print information about state transitions at the tag distribution level, use the **debug tag-switching tdp peer state-machine** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

```
debug tag-switching tdp peer state-machine
```

```
no debug tag-switching tdp peer state-machine
```

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

TDP sessions are supported by data structures and state machines at three levels:

- Transport—The transport level establishes and maintains TCP connections used to support TDP sessions.
- Protocol—The protocol level implements the TDP session setup protocol, and constructs and parses TDP PDUs and PIEs.
- Tag distribution—The tag distribution level uses TDP sessions to exchange tags with TDP peer devices.

The **debug tag-switching tdp transport** command provides visibility of activity at the transport level, the **debug tag-switching tdp session** command at the protocol level, and the **debug tag-switching tdp peer** command at the tag distribution level.

Examples

The following is sample output from the **debug tag-switching tdp peer state-machine** command:

```
Router# debug tag-switching tdp peer state-machine

tagcon: start TDP TCP timers for 202.0.0.1:1 (pp 0x60D8ABC8)
tagcon: adj 202.0.0.1:1-1 (pp 0x60D8ABC8): Event unsol open
        unsol op pdg -> estab
tagcon: start TDP TCP timers for 210.9.0.9:0 (pp 0x60D93608)
tagcon: adj 210.9.0.9:0 (pp 0x60D93608): Event unsol open
        unsol op pdg -> estab
tagcon: adj 210.9.0.9:0 (pp 0x60D93608): Event down
        estab -> ddestroy
tagcon: adj 202.0.0.1:1 (pp 0x60D8ABC8): Event down
        estab -> ddestroy
tagcon: start TDP TCP timers for 202.0.0.1:1 (pp 0x60DAC678)
tagcon: adj 202.0.0.1:1-1 (pp 0x60DAC678): Event unsol open
        unsol op pdg -> defrd
tagcon: start TDP TCP timers for 210.9.0.9:0 (pp 0x60D895C4)
tagcon: adj 210.9.0.9:0 (pp 0x60D895C4): Event unsol open
        unsol op pdg -> defrd
tagcon: adj 210.9.0.9:0 (pp 0x60D93608): Event cleanup done
        ddestroy -> non-ex
tagcon: adj 210.9.0.9:0 (pp 0x60D895C4): Event undefer
        defrd -> estab
```

```
tagcon: adj 202.0.0.1:1 (pp 0x60D8ABC8): Event cleanup done
      dstroy -> non-ex
tagcon: adj 202.0.0.1:1-1 (pp 0x60DAC678): Event undefer
      defrd -> estab
```

Table 244 describes the significant fields shown in the display.

Table 244 *debug tag-switching tdp peer state-machine Field Descriptions*

Field	Description
tagcon:	Identifies the source of the message as the tag control subsystem.
adj <i>a.b.c.d:e</i>	TDP identifier of the peer device for the session with the state change.
(pp 0xnnnnnnnn)	Address of the data structure used to represent the peer device at the tag distribution level. It is useful for correlating debug output.
Event E	Event causing the state change.
S1 -> S2	State of the TDP session has changed from state S1 to state S2.

debug tag-switching tdp pies received

To print information about TDP protocol information elements (PIEs) received from TDP peer devices, use the **debug tag-switching tdp pies received** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug tag-switching tdp pies received [all]

no debug tag-switching tdp pies received [all]

Syntax Description

all (Optional) TDP received PIEs, including periodic keepalive PIEs.

Command Modes

Privileged EXEC

Usage Guidelines

TDP requires periodic transmission of keepalive PIEs. If you do not specify the **all** option, periodic keepalive PIEs are not displayed.

Examples

The following is sample output from the **debug tag-switching tdp pies received** command:

```
Router# debug tag-switching tdp pies received all

tdp: Rcvd open PIE from 202.0.0.1 (pp 0x0)
tdp: Rcvd keep_alive PIE from 202.0.0.1:1 (pp 0x0)
tdp: Rcvd request_bind PIE from 202.0.0.1:1 (pp 0x60DAC678)
tdp: Rcvd request_bind PIE from 202.0.0.1:1 (pp 0x60DAC678)
tdp: Rcvd open PIE from 210.9.0.9 (pp 0x0)
tdp: Rcvd keep_alive PIE from 210.9.0.9:0 (pp 0x0)
tdp: Rcvd bind PIE from 202.0.0.1:1 (pp 0x60DAC678)
tdp: Rcvd bind PIE from 202.0.0.1:1 (pp 0x60DAC678)
```

[Table 245](#) describes the significant fields shown in the display.

Table 245 debug tag-switching tdp pies received all Field Descriptions

Field	Description
tdp:	Identifies the source of the message as TDP.
Rcvd xxx PIE	Type of PIE received.
from a.b.c.d	Host that sent the PIE. Used in the early stages of the opening of a TDP session, when the TDP identifier is not yet known.
from a.b.c.d:e	TDP identifier of the peer device that sent the PIE.
(pp 0xnnnnnnnn)	Identifies the data structure used to represent the peer device at the tag distribution level. Useful for correlating debug output.

Related Commands

Command	Description
debug tag-switching tdp pies sent	Prints information about state transitions at the tag distribution level.

debug tag-switching tdp pies sent

To print information about state transitions at the tag distribution level, use the **debug tag-switching tdp pies sent** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug tag-switching tdp pies sent [all]

no debug tag-switching tdp pies sent [all]

Syntax Description	all (Optional) TDP sent PIEs, including periodic keepalive PIEs.
---------------------------	---

Command Modes	Privileged EXEC
----------------------	-----------------

Usage Guidelines	TDP requires periodic transmission of keepalive PIEs. If you do not specify the all option, periodic keepalive PIEs are not displayed.
-------------------------	---

Examples	The following is sample output from the debug tag-switching tdp pies sent all command:
-----------------	---

```
Router# debug tag-switching tdp pies sent all

tdp: Queued open PIE to 210.222.0.222:1 (pp 0x0)
tdp: Sent open PIE to 210.222.0.222:1 (pp 0x0)
tdp: Queued keep_alive PIE to 210.222.0.222:1 (pp 0x0)
tdp: Sent keep_alive PIE to 210.222.0.222:1 (pp 0x0)
tdp: Queued request_bind PIE to 210.222.0.222:1 (pp 0x60F264C8)
tdp: Sent request_bind PIE to 210.222.0.222:1 (pp 0x60F264C8)
tdp: Queued request_bind PIE to 210.222.0.222:1 (pp 0x60F264C8)
tdp: Sent request_bind PIE to 210.222.0.222:1 (pp 0x60F264C8)
tdp: Queued open PIE to 210.8.0.8 (pp 0x0)
tdp: Queued bind PIE to 210.222.0.222:1 (pp 0x60F264C8)
tdp: Sent bind PIE to 210.222.0.222:1 (pp 0x60F264C8)
tdp: Queued bind PIE to 210.222.0.222:1 (pp 0x60F264C8)
tdp: Sent bind PIE to 210.222.0.222:1 (pp 0x60F264C8)
tdp: Queued bind PIE to 210.222.0.222:1 (pp 0x60F264C8)
tdp: Queued open PIE to 210.8.0.8 (pp 0x0)
tdp: Sent open PIE to 210.8.0.8 (pp 0x0)
tdp: Queued keep_alive PIE to 210.8.0.8:0 (pp 0x0)
tdp: Sent keep_alive PIE to 210.8.0.8:0 (pp 0x0)
tdp: Queued address PIE to 210.8.0.8:0 (pp 0x60F161AC)
tdp: Sent address PIE to 210.8.0.8:0 (pp 0x60F161AC)
tdp: Queued bind PIE to 210.8.0.8:0 (pp 0x60F161AC)
tdp: Queued bind PIE to 210.8.0.8:0 (pp 0x60F161AC)
tdp: Queued bind PIE to 210.8.0.8:0 (pp 0x60F161AC)
tdp: Queued bind PIE to 210.8.0.8:0 (pp 0x60F161AC)
tdp: Queued bind PIE to 210.8.0.8:0 (pp 0x60F161AC)
tdp: Sent bind PIE to 210.8.0.8:0 (pp 0x60F161AC)
tdp: Sent bind PIE to 210.8.0.8:0 (pp 0x60F161AC)
tdp: Sent bind PIE to 210.8.0.8:0 (pp 0x60F161AC)
tdp: Sent bind PIE to 210.8.0.8:0 (pp 0x60F161AC)
tdp: Sent bind PIE to 210.8.0.8:0 (pp 0x60F161AC)
```

Table 246 describes the significant fields shown in the display.

Table 246 *debug tag-switching tdp sent all Field Descriptions*

Field	Description
tdp:	Identifies the source of the message as TDP.
Queued <i>xxx</i> PIE	Indicates that a PIE of the specified type has been queued for transmission.
Sent <i>xxx</i> PIE	Indicates that a PIE of the specified type has been sent on the TDP session TCP connection.
to <i>a.b.c.d</i>	Host to which the PIE has been sent or for which it has been queued. Used in the early stages of opening a TDP session when the TDP identifier is not yet known.
to <i>a.b.c.d:e</i>	TDP identifier of the peer device to which the PIE has been sent or for which it has been queued.
(pp 0xnnnnnnnn)	Identifies the data structure used to represent the peer device at the tag distribution level. Useful for correlating debug output.

Related Commands

Command	Description
debug tag-switching tdp pies received	Prints information about TDP PIEs received from TDP peer devices.
debug tag-switching tdp session io	Prints the contents of TDP PIEs sent to and received from TDP peer devices.

debug tag-switching tdp session io

To print the contents of TDP PIEs sent to and received from TDP peer devices, use the **debug tag-switching tdp session io** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug tag-switching tdp session io [all]

no debug tag-switching tdp session io [all]

Syntax Description

all (Optional) TDP session I/O activity, including I/O for periodic keepalives.

Command Modes

Privileged EXEC

Usage Guidelines

TDP sessions are supported by data structures and state machines at three levels:

- Transport—The transport level establishes and maintains TCP connections used to support TDP sessions.
- Protocol—The protocol level implements the TDP session setup protocol, and constructs and parses TDP PDUs and PIEs.
- Tag distribution—The tag distribution level uses TDP sessions to exchange tags with TDP peer devices.

The **debug tag-switching tdp transport** command provides visibility of activity at the transport level, the **debug tag-switching tdp session** command at the protocol level, and the **debug tag-switching tdp peer** command at the tag distribution level.

TDP requires periodic transmission of keepalive PIEs. If you do not specify the **all** option, periodic keepalive PIEs are not displayed.

Examples

The following is sample output from the **debug tag-switching tdp session io all** command:

```
Router# debug tag-switching tdp session io all

tdp: Rcvd open PIE from 210.9.0.9 (pp 0x0)
tdp: TDP open PIE: PDU hdr: TDP Id: 210.9.0.9:0; PIE Contents:
 0x00 0x01 0x00 0x10 0xD2 0x09 0x00 0x09 0x00 0x00 0x00 0x00 0x01 0x00 0x00 0x04
 0x01 0x00 0x00 0x1E
tdp: Sent open PIE to 210.9.0.9:0 (pp 0x0)
tdp: TDP open PIE: PDU hdr: TDP Id: 172.27.32.28:0; PIE Contents:
 0x00 0x01 0x00 0x10 0xAC 0x1B 0x20 0x1C 0x00 0x00 0x00 0x00 0x01 0x00 0x00 0x04
 0x01 0x00 0x00 0x0F
tdp: Sent keep_alive PIE to 210.9.0.9:0 (pp 0x0)
tdp: TDP keep_alive PIE: PDU hdr: TDP Id: 172.27.32.28:0; PIE Contents:
 0x00 0x01 0x00 0x0C 0xAC 0x1B 0x20 0x1C 0x00 0x00 0x00 0x00 0x05 0x00 0x00 0x00
tdp: Rcvd keep_alive PIE from 210.9.0.9:0 (pp 0x0)
tdp: TDP keep_alive PIE: PDU hdr: TDP Id: 210.9.0.9:0; PIE Contents:
 0x00 0x01 0x00 0x0C 0xD2 0x09 0x00 0x09 0x00 0x00 0x00 0x00 0x05 0x00 0x00 0x00
tdp: Rcvd address PIE from 210.9.0.9:0 (pp 0x60E109F0)
tdp: TDP address PIE: PDU hdr: TDP Id: 210.9.0.9:0; PIE Contents:
 0x00 0x01 0x00 0x35 0xD2 0x09 0x00 0x09 0x00 0x00 0x00 0x00 0x00 0x08 0x00 0x00 0x29
```

```

0x00 0x01 0x00 0x03 0x00 0x23 0x20 0x63 0x65 0x00 0x09 0x20 0xAC 0x1B 0x20 0x1D
0x20 0x0A 0x69 0x00 0x09 0x20 0x0A 0x5C 0x00 0x09 0x20 0x0A 0x6F 0x00 0x09 0x20
0x0A 0xCD 0x00 0x09 0x20 0xD2 0x09 0x00 0x09
tdp: Rcvd bind PIE from 210.9.0.9:0 (pp 0x60E109F0)
tdp: TDP bind PIE: PDU_hdr: TDP Id: 210.9.0.9:0; PIE Contents:
0x00 0x01 0x00 0xFC 0xD2 0x09 0x00 0x09 0x00 0x00 0x00 0x00 0x02 0x00 0x00 0xF0
0x00 0x00 0x00 0x00 0x00 0x01 0x00 0x02 0x00 0xE6 0x00 0x00 0x00 0x01 0x10
0x0A 0x6F 0x00 0x00 0x00 0x00 0x01 0x16 0xAC 0x1B 0x20 0x00 0x00 0x00 0x01
0x10 0xD2 0x09 0x00 0x00 0x00 0x00 0x1A 0x20 0x0A 0x0B 0x00 0x0B 0x00 0x00 0x01

```

Table 247 describes the significant fields shown in the display.

Table 247 debug tag-switching tdp session io Field Descriptions

Field	Description
tdp:	Identifies the source of the message as TDP.
Rcvd <i>xxx</i> PIE	Indicates that a PIE of the specified type has been received.
from <i>a.b.c.d</i>	Host to which the PIE has been sent. Used in the early stages of the opening of a TDP session when the TDP identifier is not yet known.
Sent <i>xxx</i> PIE	Indicates that a PIE of the specified type has been sent.
to <i>a.b.c.d</i>	Host to which the PIE has been sent. Used in the early stages of opening a TDP session when the TDP identifier is not yet known.
to <i>a.b.c.d:e</i>	TDP identifier of the peer device to which the PIE has been sent.
(pp 0xnnnnnnnn)	Identifies the data structure used to represent the peer device at the tag distribution level. Useful for correlating debug output.
--TDP <i>xxx</i> PIE	Type of PIE that has been sent.
PDU_hdr: TDP Id: <i>a.b.c.d:e</i>	TDP identifier of the sender included in the TDP PDU header.
PIE contents: 0xnn ... 0xnn	Contents of the PIE represented as a sequence of bytes.

Related Commands

Command	Description
debug tag-switching tdp pies received	Prints information about TDP PIEs received from TDP peer devices.
debug tag-switching tdp pies sent	Prints information about state transitions at the tag distribution level.

debug tag-switching tdp session state-machine

To print information about state transitions at the protocol level, use the **debug tag-switching tdp session state-machine** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

```
debug tag-switching tdp session state-machine
```

```
no debug tag-switching tdp session state-machine
```

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

TDP sessions are supported by data structures and state machines at three levels:

- Transport—The transport level establishes and maintains TCP connections used to support TDP sessions.
- Protocol—The protocol level implements the TDP session setup protocol, and constructs and parses TDP PDUs and PIEs.
- Tag distribution—The tag distribution level uses TDP sessions to exchange tags with TDP peer devices.

The **debug tag-switching tdp transport** command provides visibility of activity at the transport level, the **debug tag-switching tdp session** command at the protocol level, and the **debug tag-switching tdp peer** command at the tag distribution level.

Examples

The following is sample output from the **debug tag-switching tdp session state-machine** command:

```
Router# debug tag-switching tdp session state-machine

tdp: adj:210.9.0.9(0x60DDBB4C): Event: Xport opened;
      Non-existent -> Init pasv
tdp: tdp_create_ptcl_adj: tp = 0x60DDBB4C, ipaddr = 210.9.0.9
tdp: adj:210.9.0.9(0x60DDBB4C): Event: Xport opened;
      Init pasv -> Init pasv
tdp: adj:10.105.0.9(0x60DDBB4C): Event: Rcv TDP Open;
      Init pasv -> Open rcvd pasv
tdp: adj:10.105.0.9(0x60DDBB4C): Event: Rcv TDP KA;
      Open rcvd pasv -> Oper
tdp: adj:unknown(0x60DDBB4C): Event: Xport closed;
      Oper -> Non-existent
```

Table 248 describes the significant fields shown in the display.

Table 248 *debug tag-switching tdp session state-machine Field Descriptions*

Field	Description
tdp:	Identifies the source of the message as TDP.
adj:a.b.c.d	Identifies the network address of the TDP peer device.
(0xnnnnnnnn)	Identifies the data structure used to represent the peer device at the protocol level. Useful for correlating debug output.
Event: E	Event that caused the state transition.
S1 -> S2	State of the TDP session has changed from state S1 to state S2.

debug tag-switching tdp transport connections

To print information about the TCP connections used to support TDP sessions, use the **debug tag-switching tdp transport connections** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

```
debug tag-switching tdp transport connections
```

```
no debug tag-switching tdp transport connections
```

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

TDP sessions are supported by data structures and state machines at three levels:

- Transport—The transport level establishes and maintains TCP connections used to support TDP sessions.
- Protocol—The protocol level implements the TDP session setup protocol, and constructs and parses TDP PDUs and PIEs.
- Tag distribution—The tag distribution level uses TDP sessions to exchange tags with TDP peer devices.

The **debug tag-switching tdp transport** command provides visibility of activity at the transport level, the **debug tag-switching tdp session** command at the protocol level, and the **debug tag-switching tdp peer** command at the tag distribution level.

When two devices establish a TCP connection for a TDP session, the device with the larger transport address plays an active role and the other plays a passive role. The active device attempts to establish a TCP connection to the well-known TDP port at the passive device. The passive device waits for the connection to the well-known port to be established.

Examples

The following is sample output from the **debug tag-switching transport connections** command:

```
Router# debug tag-switching tdp transport connections
```

```
Debug output at active peer:
```

```
tdp: Opening conn; adj 0x60F7C604, 210.9.0.9 <-> 172.27.32.28  
tdp: Conn is up; adj 0x60F7C604, 210.9.0.9:11018 <-> 172.27.32.28:711  
tdp: hold-timer expired for adj 0x60F7C604, will close conn  
tdp: Closing conn 210.9.0.9:11018 <-> 172.27.32.28:711, adj 0x60F7C604
```

```
Debug output at passive peer:
```

```
tdp: Incoming conn 172.27.32.28:711 <-> 210.9.0.9:11018  
tdp: Conn closed by peer; adj 0x60EB5FD4  
172.27.32.28:711 <-> 210.9.0.9:11018, Ethernet1/1/1  
tdp: Closing conn 172.27.32.28:711 <-> 210.9.0.9:11018, adj 0x60EB5FD4
```

Table 249 describes the significant fields shown in the display.

Table 249 *debug tag-switching tdp transport connections Field Descriptions*

Field	Description
tdp:	Identifies the source of the message as TDP.
adj 0xxxxxxxxx	Identifies the data structure used to represent the peer device at the transport level. Useful for correlating debug output.
a.b.c.d -> p.q.r.s	Indicates a TCP connection between a.b.c.d and p.q.r.s.
a.b.c.d:x -> p.q.r.s:y	Indicates a TCP connection between a.b.c.d, port x and p.q.r.s, port y.

Related Commands

Command	Description
debug tag-switching tdp transport events	Prints information about the events related to the TDP peer discovery mechanism, which is used to determine the devices with which to establish TDP sessions.

debug tag-switching tdp transport events

To print information about the events related to the TDP peer discovery mechanism, which is used to determine the devices with which to establish TDP sessions, use the **debug tag-switching tdp transport events** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug tag-switching tdp transport events

no debug tag-switching tdp transport events

Syntax Description This command has no arguments or keywords.

Command Modes Privileged EXEC

Usage Guidelines TDP sessions are supported by data structures and state machines at three levels:

- Transport—The transport level establishes and maintains TCP connections used to support TDP sessions.
- Protocol—The protocol level implements the TDP session setup protocol, and constructs and parses TDP PDUs and PIEs.
- Tag distribution—The tag distribution level uses TDP sessions to exchange tags with TDP peer devices.

The **debug tag-switching tdp transport** command provides visibility of activity at the transport level, the **debug tag-switching tdp session** command at the protocol level, and the **debug tag-switching tdp peer** command at the tag distribution level.

Examples The following is sample output from the **debug tag-switching tdp transport events** command:

```
Router# debug tag-switching tdp transport events

tdp: Rcvd hello; Ethernet1/1/1, from 10.105.0.9 (210.9.0.9:0), intf_id 0, opt 0x4
tdp: Hello from 10.105.0.9 (210.9.0.9:0) to 255.255.255.255, opt 0x4
tdp: New adj 0x60DF6E50 from 10.105.0.9 (210.9.0.9:0), Ethernet1/1/1
tdp: Rcvd hello; ATM3/0.1, from 200.26.0.4 (202.0.0.1:1), intf_id 1, opt 0x4, tcatm
tdp: Rcvd hello; Ethernet1/1/1, from 10.105.0.9 (210.9.0.9:0), intf_id 0, opt 0x4
tdp: Hello from 10.105.0.9 (210.9.0.9:0) to 255.255.255.255, opt 0x4
tdp: Ignore Hello Timer for Ethernet1/1/1; intf not TDP ready
tdp: Send hello; Ethernet1/1/1, src/dst 10.105.0.8/255.255.255.255, inst_id 0
tdp: Incoming conn 172.27.32.28:711 <-> 210.9.0.9:11019
tdp: Found adj 0x60DF6E50 for 210.9.0.9 (Hello xport addr opt)
tdp: New temporary adj 0x61033D38 from 210.9.0.9
tdp: Real adj 0x60DF6E50 bound to 210.9.0.9:0, replacing temp adj 0x61033D38
tdp: Adj 0x61033D38; state set to closed
tdp: Rcvd hello; Ethernet1/1/1, from 10.105.0.9 (210.9.0.9:0), intf_id 0, opt 0x4
tdp: Rcvd hello; ATM3/0.1, from 200.26.0.4 (202.0.0.1:1), intf_id 1, opt 0x4, tcatm
tdp: Send hello; ATM3/0.1, src/dst 99.101.0.8/255.255.255.255, inst_id 1, tcatm
tdp: Rcvd hello; Ethernet1/1/1, from 10.105.0.9 (210.9.0.9:0), intf_id 0, opt 0x4
tdp: Send hello; Ethernet1/1/1, src/dst 10.105.0.8/255.255.255.255, inst_id 0
tdp: Rcvd hello; ATM3/0.1, from 200.26.0.4 (202.0.0.1:1), intf_id 1, opt 0x4, tcatm
```

Table 250 describes the significant fields shown in the display.

Table 250 *debug tag-switching tdp transport events Field Descriptions*

Field	Description
tdp:	Identifies the source of the message as TDP.
adj 0xnnnnnnnn	Identifies the data structure used to represent the peer device at the transport level. Useful for correlating debug output.
a.b.c.d (p.q.r.s:n)	Network address and TDP identifier of the peer device.
intf_id	Interface identifier (nonzero for TC-ATM interfaces, 0 otherwise).
opt 0xn	Bits that describe options in the TDP discovery hello packet: <ul style="list-style-type: none"> • 0x1—Directed hello option • 0x2—Send directed hello option • 0x4—Transport address option

Related Commands

Command	Description
debug tag-switching tdp transport connections	Prints information about the TCP connections used to support TDP sessions.

debug tag-switching tdp transport timers

To print information about events that restart the “hold” timers that are part of the TDP discovery mechanism, use the **debug tag-switching tdp transport timers** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug tag-switching tdp transport timers

no debug tag-switching tdp transport timers

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

TDP sessions are supported by data structures and state machines at three levels:

- Transport—The transport level establishes and maintains TCP connections used to support TDP sessions.
- Protocol—The protocol level implements the TDP session setup protocol. The construction and parsing of TDP PDUs and PIEs occur at this level.
- Tag distribution—The tag distribution level uses TDP sessions to exchange tags with TDP peer devices.

The **debug tag-switching tdp transport** command provides visibility of activity at the transport level, the **debug tag-switching tdp session** command at the protocol level, and the **debug tag-switching tdp peer** command at the tag distribution level.

Examples

The following is sample output from the **debug tag-switching tdp transport timers** command:

```
Router# debug tag-switching tdp transport timers

tdp: Start holding timer; adj 0x60D5BC10, 200.26.0.4
tdp: Start holding timer; adj 0x60EA9360, 10.105.0.9
tdp: Start holding timer; adj 0x60D5BC10, 200.26.0.4
tdp: Start holding timer; adj 0x60EA9360, 10.105.0.9
tdp: Start holding timer; adj 0x60D5BC10, 200.26.0.4
tdp: Start holding timer; adj 0x60EA9360, 10.105.0.9
```

[Table 251](#) describes the significant fields shown in the display.

Table 251 debug tag-switching tdp transport timers Field Descriptions

Field	Description
tdp	Identifies the source of the message as TDP.
adj 0xnmmmmmm	Identifies the data structure used to represent the peer device at the transport level.
a.b.c.d	Network address of the peer device.

Related Commands	Command	Description
	debug tag-switching tdp transport events	Prints information about the events related to the TDP peer discovery mechanism, which is used to determine the devices with which to establish TDP sessions.

debug tag-switching tfib cef

To print detailed information about tag rewrites being created, resolved, and deactivated as CEF routes are added, changed, or removed, use the **debug tag-switching tfib cef** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug tag-switching tfib cef

no debug tag-switching tfib cef

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

Several lines of output are produced for each route placed into the TFIB. If your router has thousands of tagged routes, be careful about issuing this command. When Tag Switching is first enabled, each of these routes is placed into the TFIB. If the **debug tag-switching tfib cef** command is issued, several lines of output are displayed for each route.

Examples

The following is sample output from the **debug tag-switching tfib cef** command:

```
Router# debug tag-switching tfib cef

Cisco Express Forwarding related TFIB services debugging is on

tagcon: tc_ip_rtlookup fail on 10.0.0.0/8:subnet_lookup failed
TFIB: route tag chg 10.7.0.7/32,idx=1,inc=Withdrn,outg=Withdrn,enabled=0x2
TFIB: fib complete delete: prefix=10.7.0.7/32,inc tag=26,delete_info=1
TFIB: deactivate tag rew for 10.7.0.7/32,index=0
TFIB: set fib rew: pfx 10.7.0.7/32,index=0,add=0,tag_rew->adj=Ethernet2/3
TFIB: resolve tag rew,prefix=10.7.0.7/32,no tag_info,no parent
TFIB: fib scanner start:needed:1,unres:0,mac:0,loadinfo:0
TFIB: resolve tag rew,prefix=10.7.0.7/32,no tag_info,no parent
TFIB: fib upd loadinf 10.100.100.100/32,tag=Tun_hd,fib no loadin,tfib no loadin
TFIB: fib check cleanup for 10.100.100.100/32,index=0,return_value=0
TFIB: fib_scanner_end
TFIB: create dynamic entry for 10.11.0.11/32
TFIB: call find_route_tags,dist_method=1,next_hop=10.93.0.11,Et2/3
TFIB: route tag chg 10.11.0.11/32,idx=0,inc=26,outg=Unkn,enabled=0x3
TFIB: create tag info 10.11.0.11/32,inc tag=26,has no info
TFIB: resolve tag rew,prefix=10.11.0.11/32,has tag_info,no parent
TFIB: finish fib res 10.11.0.11/32:index 0,parent outg tag no parent
TFIB: fib upd loadinf 10.11.0.11/32,tag=26,fib no loadin,tfib no loadin
TFIB: set fib rew: pfx 10.11.0.11/32,index=0,add=1,tag_rew->adj=Ethernet2/3
tagcon: route_tag_change for: 10.250.0.97/32
      intag 33, outtag 28, nexthop tsr 10.11.0.11:0
TFIB: route tag chg 10.250.0.97/32,idx=0,inc=33,outg=28,enabled=0x3
TFIB: deactivate tag rew for 10.250.0.97/32,index=0
TFIB: set fib rew: pfx 10.250.0.97/32,index=0,add=0,tag_rew->adj=Ethernet2/3
TFIB: create tag info 10.250.0.97/32,inc tag=33,has old info
On VIP:
TFIB: route tag chg 10.13.72.13/32,idx=0,inc=34,outg=Withdrn,enabled=0x3
TFIB: deactivate tag rew for 10.13.72.13/32,index=0
```

```

TFIB: set fib rew: pfx 10.13.72.13/32,index=0,add=0,tag_rew->adj=
TFIB: create tag info 10.13.72.13/32,inc tag=34,has old info
TFIB: resolve tag rew,prefix=10.13.72.13/32,has tag_info,no parent
TFIB: finish fib res 10.13.72.13/32:index 0,parent outg tag no parent
TFIB: set fib rew: pfx 10.100.100.100/32,index=0,add=0,tag_rew->adj=
TFIB: create tag info 10.100.100.100/32,inc tag=37,has old info
TFIB: resolve tag rew,prefix=10.100.100.100/32,has tag_info,no parent
TFIB: finish fib res 10.100.100.100/32:index 0,parent outg tag no parent
TFIB: fib upd loadinf 10.100.100.100/32,tag=37,fib no loadin,tfib no loadin

```

Table 252 describes the significant fields shown in the display. See Table 254 for a description of special tag names seen in the debug output.

Table 252 debug tag-switching tfib cef Field Descriptions

Field	Description
TFIB	Name of the subsystem issuing the debug output.
tagcon	Name of subsystem issuing the debug output (Tag Control).
tc_ip_rtlookup fail on x.y.w.z/m: subnet_lookup failed	The destination with IP address and mask shown is not in the Routing Table.
route tag chg x.y.w.z/m	Request to create TFIB entry for specified prefix/mask.
idx=-1	Index within FIB entry of the path whose TFIB entry is being created. -1 means all paths for this FIB entry.
inc=s	Incoming tag of entry being processed.
outg=s	Outgoing tag of entry being processed.
enabled=0xn	Bit mask indicating types of Tag Switching currently enabled (0x1 = dynamic, 0x2 = TSP tunnels, 0x3 = both).
fib complete delete	Indicates FIB entry being deleted.
prefix=x.y.w.z/m	Destination prefix.
delete_info=1	Indicates that tag_info is also being deleted.
deactivate tag rew for x.y.w.z/m	Indicates tag rewrite for specified prefix is being deleted.
index=n	Index of path in FIB entry being processed.
set fib rew: pfx x.y.w.z/m	Indicates tag rewrite is being installed or deleted from the FIB entry for the specified destination for tag imposition purposes.
add=0	Indicates tag rewrite is being deleted from the FIB (no longer imposing tag).
tag_rew->adj=s	Adjacency of tag_rewrite for tag imposition.
resolve tag rew,prefix=x.y.w.z/m	Indicates the FIB route to the specified prefix is being resolved.
no tag_info	Indicates there is no tag_info for the destination (destination not tagged).
no parent	Indicates route is not recursive.
fib scanner start	Indicates the periodic scan of the FIB has started.
needed:1	Indicates TFIB needs the FIB to be scanned.
unres:n	Number of unresolved TFIB entries.
mac:n	Number of TFIB entries missing MAC strings.

Table 252 *debug tag-switching tfib cef Field Descriptions (continued)*

Field	Description
loadinfo:n	Indicates whether nonrecursive accounting state has changed and thus the loadinfo information in the TFIB needs to be adjusted.
fib upd loadinf <i>x.y.w.z/m</i>	Indicates that a check for nonrecursive accounting is being made and TFIB loadinfo information for specified prefix is being updated.
tag= <i>s</i>	Incoming tag of entry.
fib no loadin	Indicates corresponding FIB entry has no loadinfo.
tfib no loadin	Indicates TFIB entry has no loadinfo.
fib check cleanup for <i>x.y.w.z/m</i>	Indicates a check is being made on the TFIB entry for specified destination to see if rewrite needs to be removed from TFIB.
return_value= <i>x</i>	If 0, no change in TFIB entry. If 1, there was a change.
fib_scanner_end	FIB scan has come to an end.
create dynamic entry for <i>x.y.w.z/m</i>	TFIB has been enabled and a TFIB entry is being created for the specified destination.
call find_route_tags	Tags for that destination are being requested.
dist_method= <i>n</i>	Tag distribution method—for example, TDP, TC-ATM.
next_hop= <i>x.y.z.w</i>	Next hop for the destination.
interface name	Outgoing interface for the destination.
create tag info	Tag_info data structure is being created for the destination.
has no info	Destination does not already have a tag_info.
finish fib res <i>x.y.z.w/m</i>	TFIB entry for the specified route is being completed.
parent outg tag <i>s</i>	If recursive, specifies the outgoing tag of the route through which it recurses (the parent). If not recursive, <i>s</i> = “no parent.”
tagcon: route_tag_change for: <i>x.y.z.w/m</i>	Tag Control is notifying TFIB that tags are available for specified destination.
intag <i>s</i>	Incoming tag for the destination.
outtag <i>s</i>	Outgoing tag for the destination.
nexthop tsr <i>x.y.z.w.i</i>	TDP ID of next hop which sent the tag.

Related Commands

Command	Description
debug tag-switching tfib enc	Prints detailed information about tag encapsulations while tag rewrites are created or updated and placed into the TFIB.

debug tag-switching tfib enc

To print detailed information about tag encapsulations while tag rewrites are created or updated and placed into the Tag Forwarding Information Base (TFIB), use the **debug tag-switching tfib enc** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug tag-switching tfib enc

no debug tag-switching tfib enc

Syntax Description This command has no arguments or keywords.

Command Modes Privileged EXEC

Usage Guidelines Several lines of output are produced for each route placed into the TFIB. If your router has thousands of tagged routes, be careful about issuing this command. When Tag Switching is first enabled, each of these routes is placed into the TFIB, and a tag encapsulation is created. If the **debug tag-switching tfib enc** command is issued, several lines of output are displayed for each route.

This output shows you which adjacency the tag rewrite is being created on and the tags.

Examples The following is sample output from the **debug tag-switching tfib enc** command. This example shows the encapsulations for three routes that have been created and placed into the TFIB:

```
Router# debug tag-switching tfib enc

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

[Table 253](#) describes the significant fields shown in the display.

Table 253 debug tag-switching tfib enc Field Descriptions

Field	Description
TFIB	Indicates the source of the message as the TFIB subsystem.
finish res	Indicates that the TFIB resolution is being finished.
inc tag=x or inc=x	Incoming (local) tag for the TFIB entry is being created. Tags can be numbers or special values.

Table 253 *debug tag-switching tfib enc Field Descriptions (continued)*

Field	Description
outg=y	Outgoing (remote) tag for the TFIB entry is being created.
next_hop=a.b.c.d	IP address of next hop for the destination.
interface	Outgoing interface through which a packet will be sent.
get ip adj	Indicates that the IP adjacency to use in the TFIB entry is being determined.
get tag adj	Indicates that the tag-switching adjacency to use for the TFIB entry is being determined.
addr = a.b.c.d	IP address of the adjacency.
is_p2p=x	If 1, this is a point-to-point adjacency. If 0, it is not.
fibidb = s	Interface of the adjacency.
linktype = x	Link type of the adjacency (7 = LINK_IP or 79 = LINK_TAG).
sizes x,y,z	x = length of macstring, y = length of tag encapsulation, z = tag MTU.
type = x	Tag encapsulation type. 0= normal, 1 = TCATM, 2 = TSP tunnel.
idb:s	Outgoing interface.
update_mac	Indicates that the macstring of the adjacency is being updated.

[Table 254](#) describes the special tags that may appear in the debug output.

Table 254 *Special Tags Seen in Debug Output*

Special Tag	Meaning
Unassn—Inital value	No tag assigned yet.
Unused	Destination does not have a tag (for example, a BGP route).
Withdrn	Tag for this destination has been withdrawn.
Unkn	Destination should have a tag, but it is not known yet.
Get_res	Recursive route that will get a tag when resolved.
Exp_null	Explicit null tag—used over TC-ATM.
Imp_null	Implicit null tag—for directly connected routes.
Tun_hd	Head of TSP tunnel.

Related Commands

Command	Description
debug tag-switching tfib state	Traces what is happening as Tag Switching is enabled or disabled.

debug tag-switching tfib state

To trace what is happening as Tag Switching is enabled or disabled, use the **debug tag-switching tfib state** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug tag-switching tfib state

no debug tag-switching tfib state

Syntax Description This command has no arguments or keywords.

Command Modes Privileged EXEC

Usage Guidelines Use this command when you wish to trace what happens to the TFIB when you issue the **tag-switching ip** or **tag-switching tsp-tunnel** command.

Examples The following is sample output from the **debug tag-switching tfib state** command:

```
Router# debug tag-switching tfib state

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

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

On linecard only:

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

[Table 255](#) describes the significant fields shown in the display.

Table 255 debug tag-switching tfib state Field Descriptions

Field	Description
TFIB	Identifies the source of the message as the TFIB subsystem.
Upd tag sb <i>x</i>	Indicates that the status of the <i>x</i> th Tag Switching subblock is being updated, where <i>x</i> is the interface number. There is a Tag Switching subblock for each interface on which Tag Switching has ever been enabled.

Table 255 *debug tag-switching tfib state Field Descriptions (continued)*

Field	Description
(status:0xC1,tmtu:1500,VPI:1-1VC=0/32,et:0/0/0),lc 0x0)	Values of the fields shown in the Tag Switching subblock: the status byte, the MTU, the range of ATM VPs, control VP and control VC (if this is a TC-ATM interface), the encapsulation type, encapsulation information, tunnel interface number, and the line card number to which the update message is being sent (0 means all line cards)
intf status chg	Indicates that there was an interface status change.
idb=Et4/0/2	Interface whose status changed.
status=0xC1	New status bits in the Tag Switching subblock of the idb.
oldstatus=0xC3	What the old status bits were before the change.
Interface dyntag change, change in state to Ethernet4/0/2	Indicates that there was a change in dynamic tag status for the particular interface.
enable entered	Indicates that the code that enables the TFIB was invoked.
TFIB already enabled	Indicates that TFIB was already enabled when this call was made.
table exists	Indicates that a TFIB table had already been allocated in a previous call.
cleanup: tfib[x] still non-0	Indicates TFIB is in the process of being deleted, but that slot x is still around.
disable lc mesg recvd, type=0x1	Indicates that a message to disable Tag Switching type 1 (dynamic) was received by the line card.
disable entered, table exists,type=0x1	Indicates that a call to disable dynamic Tag Switching was issued.
Ethernet4/0/1 fibidb subblock message received	Indicates that a message giving fibidb status change was received on the lc.
enable lc msg recvd,type=0x1	Indicates that a message to enable Tag Switching type 1 (dynamic) was received by the line card.
Tunnel301 set encapfix to 0x6016A97C	Indicates that fibidb Tunnel301 on the line card received an encapsulation fixup.
types now 0x3, returning	Gives the value of the bitmask indicating the type of Tag Switching enabled. 0x1 means dynamic Tag Switching; 0x2 means tsp-tunnels; and 0x3 means both.

Related Commands

Command	Description
debug tag-switching tfib enc	Prints detailed information about tag encapsulations while tag rewrites are created or updated and placed into the TFIB.
debug tag-switching tfib struct	Traces the allocation and freeing of TFIB-related data structures—the TFIB itself, tag-rewrites, and tag-info data.

debug tag-switching tfib struct

To trace the allocation and freeing of TFIB-related data structures—the TFIB itself, tag-rewrites, and tag-info data, use the **debug tag-switching tfib struct** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug tag-switching tfib struct

no debug tag-switching tfib struct

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Examples

The following is sample output from the **debug tag-switching tfib struct** command:

```
Router# debug tag-switching tfib struct

TFIB data structure changes debugging is on

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

[Table 256](#) describes the significant fields shown in the display.

Table 256 debug tag-switching tfib struct Field Descriptions

Field	Description
TFIB	Subsystem issuing the message.
delete tag rew	Tag_rewrite is being freed.

Table 256 *debug tag-switching tfib struct Field Descriptions (continued)*

Field	Description
remove from tfib	Tag rewrite is being removed from the TFIB.
inc tag- <i>s</i>	Incoming tag of the entry being processed.
set loadinfo	Loadinfo field in the TFIB entry is being set (used for nonrecursive accounting).
tag= <i>s</i>	Incoming tag of the entry being processed.
no old loadinfo	TFIB entry did not have a loadinfo before.
no new loadinfo	TFIB entry should not have a loadinfo now.
TFIB not in use. Checking for entries.	Tag Switching has been disabled, and the TFIB is being freed up.
cleanup: tfib[<i>x</i>] still non-0	TFIB is being checked for any entries in use, and entry <i>x</i> is the lowest numbered slot still in use.
TFIB freed	TFIB table has been freed.
enable, TFIB allocated, size <i>x</i> bytes, maxtag = <i>y</i>	Tag Switching has been enabled, and a TFIB of <i>x</i> bytes has been allocated. The largest legal tag is <i>y</i> .
create tag rewrite	Tag_rewrite is being created.
inc <i>s</i>	Incoming tag.
outg <i>s</i>	Outgoing tag.
add to tfib at <i>s</i>	Tag_rewrite has been placed in the TFIB at slot <i>s</i> .
first in circular list	TFIB slot had been empty, and this is the first rewrite in the list.
mac=0,enc=0	Length of the mac string and total encapsulation length, including tags.
added to circular list	Tag_rewrite is being added to a TFIB slot which already had an entry. This rewrite is being inserted in the circular list.

Related Commands

Command	Description
debug tag-switching tfib cef	Prints detailed information about tag rewrites being created, resolved, and deactivated as CEF routes are added, changed, or removed.
debug tag-switching tfib enc	Prints detailed information about tag encapsulations while tag rewrites are created or updated and placed into the TFIB.
debug tag-switching tfib state	Traces what is happening as tag-switching is enabled or disabled.

debug tag-switching tfib tsp

To print detailed information about tag rewrites being created and deleted as TSP tunnels are added or removed, use the **debug tag-switching tfib tsp** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug tag-switching tfib tsp

no debug tag-switching tfib tsp

Syntax Description This command has no arguments or keywords.

Command Modes Privileged EXEC

Examples The following is sample output from the **debug tag-switching tfib tsp** command:

```
Router# debug tag-switching tfib tsp

TSP-tunnel related TFIB services debugging is on

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

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

Table 257 describes the significant fields shown in the display.

Table 257 *debug tag-switching tfib state Field Descriptions*

Field	Description
tagtun	Name of routine entered
next hop=x.y.z.w	Next hop for the tunnel being created
inc=x	Incoming tag for this hop of the tunnel being created
outg=x	Outgoing tag (1 means Implicit Null tag)
idb=s	Outgoing interface for the tunnel being created
if_number=7	Interface number of outgoing interface
tsptunnel	Name of routine entered
tsptun update loadinfo	Procedure being performed
tag=x	Incoming tag of TFIB slot whose loadinfo is being updated
loadinfo_reqd=x	Indicates whether a loadinfo is expected for this entry (nonrecursive accounting is on)
no new loadinfo	Means no change in loadinfo required
no old loadinfo	Means there was no loadinfo before
tagtun tag chg linec	Linecard is being informed of the TSP tunnel
fiblc=x	Which linecard being informed (0 means all)
in tg=x	Incoming tag of new TSP tunnel
o tg=x	Outgoing tag of new TSP tunnel
if=x	Outgoing interface number
nh=x.y.w.z	Next hop IP address
tagtun_delete	Procedure being performed: delete a TSP tunnel
tagtun tag del linec	Inform linecard of TSP tunnel deletion
tagtun chg msg	Linecard has received a message to create a TSP tunnel

Related Commands

Command	Description
debug tag-switching tfib enc	Prints detailed information about tag encapsulations while tag rewrites are created or updated and placed into the TFIB.

debug tag-switching tsp-tunnels events

To enable logging of significant events that affect TSP tunnels, use the **debug tag-switching tsp-tunnels events** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug tag-switching tsp-tunnels events

no debug tag-switching tsp-tunnels events

Syntax Description This command has no arguments or keywords.

Command Modes Privileged EXEC

Usage Guidelines The **debug tag-switching tsp-tunnels events** command displays notifications received by the TSP tunnels module. The notifications displayed are generally associated with timers, interfaces, or interface addresses.

Examples The following is sample output from the **debug tag-switching tsp-tunnels events** command:

```
Router# debug tag-switching tsp-tunnels events
```

```
TSP-TUNNEL: received event for Tunnels Checkup Timer: timer fired
TSP-TUNNEL: received event for Tunnel1206: Interface administratively down
```

[Table 258](#) describes the significant fields shown in the display.

Table 258 *debug tag-switching tsp-tunnel events Field Descriptions*

Field	Description
TSP-TUNNEL	Identifies the source of the message as TSP tunnels.
received event for	Identifies the object affected by the event.
timer fired	Describes the event that has occurred for the named object. This field is object-specific.

Related Commands

Command	Description
debug tag-switching tsp-tunnels signalling	Enables logging of TSP tunnel signalling activity.

debug tag-switching tsp-tunnels signalling

To enable logging of TSP tunnel signalling activity, use the **debug tag-switching tsp-tunnels signalling** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug tag-switching tsp-tunnels signalling

no debug tag-switching tsp-tunnels signalling

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

TSP tunnels are signalled using the Resource Reservation Protocol (RSVP). TSP tunnel establishment is initiated at the head-end router by making an active open request to RSVP, which causes a message to be sent toward the tailend router. At the tailend router, RSVP notifies the TSP tunnels module of the incoming open request. If the tailend router accepts the request, it makes a corresponding passive open request to RSVP, which causes a message to return to the headend router, establishing the TSP tunnel state at each hop along the way.

The **debug tag-switching tsp-tunnels signalling** command displays the signalling requests made to RSVP by the TSP tunnels modules at the headend and tailend routers. It also displays the signalling requests made by RSVP to the TSP tunnels modules at every hop along the path, in order to establish the TSP tunnel state.

Examples

The following is sample output from the **debug tag-switching tsp-tunnels signalling** command:

```
Router# debug tag-switching tsp-tunnels signalling

Open at tunnel head:
  TSP-TUNNEL-SIG: tunnel 10.106.0.6 1206: RSVP head-end open
Open at tunnel tail:
  TSP-TUNNEL-SIG: received NEW PATH TAIL ARRIVAL event about tunnel 10.106.0.6 1206
  TSP-TUNNEL-SIG: tunnel 10.106.0.6 1206: RSVP tailend open
  TSP-TUNNEL-SIG: tunnel 10.106.0.6 1206 NEW PATH TAIL ARRIVAL event handled successfully
State setup at tail hop:
  TSP-TUNNEL-SIG: received ADD RESV request for tunnel 10.106.0.6 1206
  TSP-TUNNEL-SIG: tunnel 10.106.0.6 1206: path previous hop is 10.2.0.10 (Et4/0/2)
  TSP-TUNNEL-SIG: sending ADD RESV reply for tunnel 10.106.0.6 1206
State setup at intermediate hop:
  TSP-TUNNEL-SIG: received ADD RESV request for tunnel 10.106.0.6 1206
  TSP-TUNNEL-SIG: tunnel 10.106.0.6 1206: path previous hop is 10.32.0.6 (AT1/0.1)
  TSP-TUNNEL-SIG: tunnel 10.106.0.6 1206: path next hop is 10.2.0.12 (Et4/0/2)
  TSP-TUNNEL-SIG: sending ADD RESV reply for tunnel 10.106.0.6 1206
State setup at head hop:
  TSP-TUNNEL-SIG: received ADD RESV request for tunnel 10.106.0.6 1206
  TSP-TUNNEL-SIG: tunnel 10.106.0.6 1206: path next hop is 10.32.0.10 (AT0/0.1)
  TSP-TUNNEL-SIG: sending ADD RESV reply for tunnel 10.106.0.6 1206
```

Table 259 describes the significant fields shown in the display.

Table 259 *debug tag-switching tsp-tunnels signalling Field Descriptions*

Field	Description
TSP-TUNNEL-SIG	Identifies the source of the message as TSP tunnels signalling.
tunnel 10.106.0.6 1206	Identifies the tunnel being signalled.
RSVP head-end open	TSP tunnels module has made an active open request to RSVP for a tunnel head.
received NEW PATH TAIL ARRIVAL	RSVP has notified the TSP tunnels module of an incoming setup request.
RSVP tailend open	TSP tunnels module has made a passive open request to RSVP for a tunnel tail.
received ADD RESV request	TSP tunnels module has received a state setup request from RSVP for a tunnel.
path previous hop is	Indicates the address of the previous hop in the path of the TSP tunnel being signalled, followed by the interface (in parentheses) being used to reach that hop.
path next hop is	Indicates the address of the next hop in the path of the TSP tunnel being signalled, followed by the interface (in parentheses) being used to reach that hop.
sending ADD RESV reply	TSP tunnels module is sending a response to an earlier state setup request made by RSVP. If an error is not explicitly indicated, then the request was completed without failure.

The display output for signalling teardown uses the same format:

Output at tunnel head:

```
TSP-TUNNEL-SIG: tunnel 10.106.0.6 1206: RSVP head-end close
TSP-TUNNEL-SIG: received DELETE RESV request for tunnel 10.106.0.6 1206
TSP-TUNNEL-SIG: tunnel 10.106.0.6 1206: path next hop is 10.32.0.10 (AT0/0.1)
TSP-TUNNEL-SIG: sending DELETE RESV reply for tunnel 10.106.0.6 1206
```

Output at intermediate hop:

```
TSP-TUNNEL-SIG: received DELETE RESV request for tunnel 10.106.0.6 1206
TSP-TUNNEL-SIG: tunnel 10.106.0.6 1206: path previous hop is 10.32.0.6 (AT1/0.1)
TSP-TUNNEL-SIG: tunnel 10.106.0.6 1206: path next hop is 10.2.0.12 (Et4/0/2)
TSP-TUNNEL-SIG: sending DELETE RESV reply for tunnel 10.106.0.6 1206
```

Output at tunnel tail:

```
TSP-TUNNEL-SIG: received PATH TAIL DELETION event about tunnel 10.106.0.6 1206
TSP-TUNNEL-SIG: tunnel 10.106.0.6 1206: RSVP tailend close
TSP-TUNNEL-SIG: received DELETE RESV request for tunnel 10.106.0.6 1206
TSP-TUNNEL-SIG: tunnel 10.106.0.6 1206: path previous hop is 10.2.0.10 (Et4/0/2)
TSP-TUNNEL-SIG: sending DELETE RESV reply for tunnel 10.106.0.6 1206
TSP-TUNNEL-SIG: PATH TAIL DELETION event handled successfully
```

Related Commands

Command	Description
debug tag-switching tsp-tunnels events	Enables logging of significant events that affect TSP tunnels.
debug tag-switching tsp-tunnels tagging	Enables logging of TSP tunnel Tag Switching state programming.

debug tag-switching tsp-tunnels tagging

To enable logging of TSP tunnel Tag Switching state programming, use the **debug tag-switching tsp-tunnels tagging** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug tag-switching tsp-tunnels tagging

no debug tag-switching tsp-tunnels tagging

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

The **debug tag-switching tsp-tunnels tagging** command displays the setup and removal of the tag-switched path state (and any additional forwarding state) at a router hop that is associated with TSP tunnels.

Examples

The following is sample output from the **debug tag-switching tsp-tunnels tagging** command:

```
Router# debug tag-switching tsp-tunnels tagging

TSP-TUNNEL-TAGGING: tunnel 10.106.0.6 1206: fabric PROGRAM request
TSP-TUNNEL-TAGGING: tunnel 10.106.0.6 1206: programming tag VC 1/43 on output interface
ATM0/0.1
TSP-TUNNEL-TAGGING: background thread starting
TSP-TUNNEL-TAGGING: background thread started (pid 57)
TSP-TUNNEL-TAGGING: descriptor D7AB40: created [1 total]
TSP-TUNNEL-TAGGING: descriptor D7AB40: continuing "Program" request
TSP-TUNNEL-TAGGING: descriptor D7AB40: allocated outgoing ATM VC 1/43 (vcd 12)
TSP-TUNNEL-TAGGING: descriptor D7AB40: set "Interface Point Out State" to, allocated
TSP-TUNNEL-TAGGING: # of resource points held for "TC-ATM" interfaces: 1
TSP-TUNNEL-TAGGING: enabling TFIB fabric programming
TSP-TUNNEL-TAGGING: TFIB is now enabled
TSP-TUNNEL-TAGGING: descriptor D7AB40: set "Fabric State" to, enabled
TSP-TUNNEL-TAGGING: descriptor D7AB40: set "Fabric Kind" to, default (TFIB)
TSP-TUNNEL-TAGGING: descriptor D7AB40: set "Fabric State" to, set
TSP-TUNNEL-TAGGING: tunnel 10.106.0.6 1206: fabric PROGRAM reply
TSP-TUNNEL-TAGGING: background thread awaiting events
```

Table 260 describes the significant fields shown in the display.

Table 260 debug tag-switching tsp-tunnels tagging Command Fields Descriptions

Field	Description
TSP-TUNNEL-TAGGING	Identifies the source of the message as TSP tunnel tagging.
tunnel 10.106.0.6 1206	Identifies the tunnel being programmed.

Table 260 *debug tag-switching tsp-tunnels tagging Command Fields Descriptions (continued)*

Field	Description
fabric PROGRAM request	Identifies the request that has been entered by the signalling code. The verb “PROGRAM” indicates that the state is being installed. All other verbs indicate that the state is being removed.
programming tag	Identifies an input or output interface being programmed and the tag being set up or removed.
descriptor xxxxxx	Identifies the switching fabric resource being used to send, switch, or receive tunnel data.
allocated outgoing...	Indicates that an outgoing tag (or VPI/VCI) has been allocated on the outgoing interface.
enabling TFIB fabric programming	TFIB switching fabric is required to support this tunnel and is enabled for use by this tunnel.
TFIB is now enabled	TFIB’s state has changed to active—it was not previously in use.
programming TFIB: 0x1A(26)* --> 0x1(1)	TFIB is being programmed to forward tunnel packets as follows: packets arriving with tag 26 will be forwarded with tag 1. Tag 1 is the Implicit Null tag.
set “<resource>” to, <state>	Resource identified by <resource> has been placed in the state described by <state>.
fabric PROGRAM reply	Indicates that the request has been completed and that a reply is being sent to the signalling code. If an error is not explicitly indicated, the request was completed without a failure.

Related Commands

Command	Description
debug tag-switching tsp-tunnels signalling	Enables logging of TSP tunnel signalling activity.

debug tag-switching xtagatm cross-connect

To display requests and responses for establishing and removing cross-connects on the controlled ATM switch, use the **debug tag-switching xtagatm cross-connect** command. To disable debugging output, use the **no** form of this command.

debug tag-switching xtagatm cross-connect

no debug tag-switching xtagatm cross-connect

Syntax Description

This command has no arguments or keywords.

Defaults

No default behavior or values.

Command History

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

Usage Guidelines

Use the **debug tag-switching xtagatm cross-connect** command to monitor requests to establish or remove cross-connects from XTagATM interfaces to the VSI master, and the VSI master's responses to these requests.



Note

Use this command with care, because it generates output for each cross-connect operation performed by the LSC. In a network configuration with a large number of label virtual circuits (LVCs), the volume of output generated may interfere with system timing and the proper operation of other router functions. Use this command only in situations in which the LVC setup or teardown rate is low.

Examples

The following is sample output from the **debug tag-switching xtagatm cross-connect** command:

```
Router# debug tag-switching xtagatm cross-connect

XTagATM: cross-conn request; SETUP, userdata 0x17, userbits 0x1, prec 7
      0xC0100 (Ct1-If) 1/32 <-> 0xC0200 (XTagATM0) 0/32
XTagATM: cross-conn response; DOWN, userdata 0x60CDCB5C, userbits 0x2, result
OK
      0xC0200 1/37 --> 0xC0300 1/37
```

Table 261 describes the significant fields shown in the sample command output shown above.

Table 261 Debug Tag-Switching XTagATM Cross-Connect Field Descriptions

Field	Description
XTagATM	Identifies the source of the debug message as an XTagATM interface.
cross-conn	Indicates that the debug message pertains to a cross-connect setup or teardown operation.
request	A request from an XTagATM interface to the VSI master to set up or tear down a cross-connect.
response	Response from the VSI master to an XTagATM interface that a cross-connect was set up or removed.
SETUP	A request for the setup of a cross-connect.
TEARDOWN	A request for the teardown of a cross-connect.
UP	The cross-connect is established.
DOWN	The cross-connect is not established.
userdata, userbits	Values passed with the request that are returned in the corresponding fields shown in the matching response.
prec	The precedence for the cross-connect.
result	Indicates the status of the completed request.
0xC0100 (Ctl-If) 1/32	Indicates the following: that one endpoint of the cross-connect is on the interface whose logical interface number is 0xC0100; that this interface is the VSI control interface; that the VPI value at this endpoint is 1; and that the VCI value at this end of the cross-connect is 32.
<->	Indicates that this is a bidirectional cross-connect.
0xC0200 (XTagATM0) 0/32	Indicates the following: that the other endpoint of the cross-connect is on the interface whose logical interface number is 0xC0200; that this interface is associated with XTagATM interface 0; that the VPI value at this endpoint is 0; and that the VCI value at this end of the cross-connect is 32.
->	Indicates that this response pertains to a unidirectional cross-connect.

Related Commands

Command	Description
<code>show xtagatm cross-connect</code>	Displays information about remotely connected ATM switches.

debug tag-switching xtagatm errors

To display information about error and abnormal conditions that occur on XTagATM interfaces, use the **debug tag-switching xtagatm errors** command. To disable debugging output, use the **no** form of this command.

debug tag-switching xtagatm errors

no debug tag-switching xtagatm errors

Syntax Description

This command has no arguments or keywords.

Defaults

No default behavior or values.

Command History

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

Usage Guidelines

Use the **debug tag-switching xtagatm errors** command to display information about abnormal conditions and events that occur on XTagATM interfaces.

Examples

The following is sample output from the **debug tag-switching xtagatm errors** command:

```
Router# debug tag-switching xtagatm errors

XTagATM VC: XTagATM0 1707 2/352 (ATM1/0 1769 3/915): Cross-connect setup
failed NO_RESOURCES
```

This message indicates that an attempt to set up a cross-connect for a terminating VC on XTagATM interface 0 failed, and that the reason for the failure was a lack of resources on the controlled ATM switch.

debug tag-switching xtagatm events

To display information about major events that occur on XTagATM interfaces, not including events for specific XTagATM VCs and switch cross-connects, use the following **debug tag-switching xtagatm events** command. To disable debugging output, use the **no** form of this command.

debug tag-switching xtagatm events

no debug tag-switching xtagatm events

Syntax Description

This command has no arguments or keywords.

Defaults

No default behavior or values.

Command History

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

Usage Guidelines

Use the **debug tag-switching xtagatm events** command to monitor major events that occur on XTagATM interfaces. This command monitors only events that pertain to XTagATM interfaces as a whole and does not include any events that pertain to individual XTagATM VCs or individual switch cross-connects. The specific events monitored when the **debug tag-switching xtagatm events** command is in effect include the following:

- Receipt of asynchronous notifications sent by the VSI master through the external ATM API (ExATM API) to an XTagATM interface.
- Resizing of the table that is used to store switch cross-connect information. This table is resized automatically as the number of cross-connects increases.
- Marking of XTagATM VCs as stale when an XTagATM interface shuts down, thereby ensuring that the stale interfaces are refreshed before new XTagATM VCs can be created on the interface.

Examples

The following is sample output from the **debug tag-switching xtagatm events** command:

```
Router# debug tag-switching xtagatm events

XTagATM: desired cross-connect table size set to 256
XTagATM: ExATM API intf event Up, port 0xA0100 (None)
XTagATM: ExATM API intf event Down, port 0xA0100 (None)
XTagATM: marking all VCs stale on XTagATM0
```

Table 262 describes the significant fields shown in the sample command output shown above.

Table 262 debug tag-Switching xtagatm events Field Descriptions

Field	Description
XTagATM	Identifies the source of the debug message as an XTagATM interface.
desired cross-connect table size set to 256	Indicates that the table of cross-connect information has been set to hold 256 entries. A single cross-connect table is shared among all XTagATM interfaces. The cross-connect table is automatically resized as the number of cross-connects increases.
ExATM API	Indicates that the information in the debug output pertains to an asynchronous notification sent by the VSI master to the XTagATM driver.
event Up/Down	Indicates the specific event that was sent by the VSI master to the XTagATM driver.
port 0xA0100 (None)	Indicates that the event pertains to the VSI interface whose logical interface number is 0xA0100, and that this logical interface is not bound (through the extended-port interface configuration command) to any XTagATM interface.
marking all VCs stale on XTagATM0	Indicates that all existing XTagATM VCs on interface XTagATM0 are marked as stale, and that XTagATM0 remains down until all of these VCs are refreshed.

debug tag-switching xtagatm vc

To display information about events that affect individual XTagATM terminating VCs, use the **debug tag-switching xtagatm vc** command. To disable debugging output, use the **no** form of this command.

debug tag-switching xtagatm vc

no debug tag-switching xtagatm vc

Syntax Description

This command has no arguments or keywords.

Defaults

No default behavior or values.

Command History

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

Usage Guidelines

Use the **debug tag-switching xtagatm vc** command to display detailed information about all events that affect individual XTagATM terminating VCs.

**Note**

Use this command with care, because it results in extensive output when many XTagATM VCs are set up or torn down. This output can interfere with system timing and normal operation of other router functions. Use the **debug tag-switching xtagatm vc** command only when a few XTagATM VCs are created or removed.

Examples

The following is sample output from the **debug tag-switching xtagatm vc** command:

```
Router# debug tag-switching xtagatm vc

XTagATM VC: XTagATM1 18 0/32 (ATM1/0 0 0/0): Setup, Down --> UpPend
XTagATM VC: XTagATM1 18 0/32 (ATM1/0 88 1/32): Complete, UpPend --> Up
XTagATM VC: XTagATM1 19 1/33 (ATM1/0 0 0/0): Setup, Down --> UpPend
XTagATM VC: XTagATM0 43 0/32 (ATM1/0 67 1/84): Teardown, Up --> DownPend
```

Table 263 describes the significant fields shown in the display.

Table 263 Debug Tag-Switching XTagATM VC Field Descriptions

Field	Description
XTagATM VC	Identifies the source of the debug message as the XTagATM interface terminating VC facility.
XTagATM <ifnum>	Identifies the particular XTagATM interface number for the terminating VC.
vcd vpi/vci	Indicates the VCD and VPI/VCI values for the terminating VC.
(ctl-if vcd vpi/vci)	Indicates the control interface, the VCD, and the VPI and VCI values for the private VC corresponding to the XTagATM VC on the control interface.
Setup, Complete, Teardown	Indicates the name of the particular event that has occurred for the indicated VC.
oldstate -> newstate	Indicates the state of the terminating VC before and after the processing of the indicated event.