debug aaa accounting through debug auto-config

debug aaa accounting through debug auto-config

debug aaa accounting

To display information on accountable events as they occur , use the **debugaaaaccounting** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug aaa accounting

no debug aaa accounting

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

The information displayed by the **debugaaaaccounting** command is independent of the accounting protocol used to transfer the accounting information to a server. Use the **debugtacacs** and **debugradius** protocol-specific commands to get more detailed information about protocol-level issues.

You can also use the **showaccounting**command to step through all active sessions and to print all the accounting records for actively accounted functions. The **showaccounting** command allows you to display the active "accountable events" on the system. It provides systems administrators a quick look at what is happening, and may also be useful for collecting information in the event of a data loss of some kind on the accounting server. The **showaccounting** command displays additional data on the internal state of the authentication, authorization, and accounting (AAA) security system if **debugaaaaccounting** is turned on as well.

Examples

The following is sample output from the **debugaaaaccounting** command:

```
Router# debug aaa accounting
```

```
16:49:21: AAA/ACCT: EXEC acct start, line 10
16:49:32: AAA/ACCT: Connect start, line 10, glare
16:49:47: AAA/ACCT: Connection acct stop:
task_id=70 service=exec port=10 protocol=telnet address=172.31.3.78 cmd=glare bytes_in=308
 bytes_out=76 paks_in=45 paks_out=54 elapsed_time=14
```

Related Commands

Command	Description
debug aaa authentication	Displays information on accountable events as they occur.
debug aaa authorization	Displays information on AAA/TACACS+ authorization.
debug radius	Displays information associated with the RADIUS.
debug tacacs	Displays information associated with the TACACS.

debug aaa authentication

To display information on authentication, authorization, and accounting (AAA) TACACS+ authentication, use the **debugaaaauthentication** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug aaa authentication

no debug aaa authentication

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
15.0(1)M	This command was introduced in a release earlier than Cisco IOS Release 15.0(1)M.
12.2(33)SRC	This command was integrated into a release earlier than Cisco IOS Release 12.2(33)SRC.
12.2(33)SXI	This command was integrated into a release earlier than Cisco IOS Release 12.2(33)SXI.
Cisco IOS XE Release 2.1	This command was integrated into a release earlier than Cisco IOS XE Release 2.1.

Usage Guidelines

Use the **debugaaaauthentication** command to learn the methods of authentication being used.

Examples

The following is sample output from the **debugaaaauthentication** command. A single EXEC login that uses the "default" method list and the first method, TACACS+, is displayed.

```
Router# debug aaa authentication
Nov 17 03:06:40.805 PST: AAA/BIND(0000000F): Bind i/f
Nov 17 03:06:40.805 PST: AAA/AUTHEN/LOGIN (0000000F): Pick method list 'default'
```

debug aaa authorization

To display information on authentication, authorization, and accounting (AAA) TACACS+ authorization, use the **debugaaaauthorization**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug aaa authorization

no debug aaa authorization

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

Use this command to learn the methods of authorization being used and the results of these methods.

Examples

The following is sample output from the**debugaaaauthorization**command. In this display, an EXEC authorization for user "carrel" is performed. On the first line, the username is authorized. On the second and third lines, the attribute value (AV) pairs are authorized. The debug output displays a line for each AV pair that is authenticated. Next, the display indicates the authorization method used. The final line in the display indicates the status of the authorization process, which, in this case, has failed.

```
Router# debug aaa authorization
2:23:21: AAA/AUTHOR (0): user='carrel'
2:23:21: AAA/AUTHOR (0): send AV service=shell
2:23:21: AAA/AUTHOR (0): send AV cmd*
2:23:21: AAA/AUTHOR (342885561): Method=TACACS+
2:23:21: AAA/AUTHOR/TAC+ (342885561): user=carrel
2:23:21: AAA/AUTHOR/TAC+ (342885561): send AV service=shell
2:23:21: AAA/AUTHOR/TAC+ (342885561): send AV cmd*
2:23:21: AAA/AUTHOR (342885561): Post authorization status = FAIL
```

The **aaaauthorization** command causes a request packet containing a series of AV pairs to be sent to the TACACS+ daemon as part of the authorization process. The daemon responds in one of the following three ways:

- Accepts the request as is
- Makes changes to the request
- Refuses the request, thereby refusing authorization

The table below describes AV pairs associated with the **debugaaaauthorization** command that may appear in the debug output.

Table 1: Attribute Value Pairs for Authorization

Attribute Value	Description
service=arap	Authorization for the AppleTalk remote access (ARA) protocol is being requested.

Attribute Value	Description
service=shell	Authorization for EXEC startup and command authorization is being requested.
service=ppp	Authorization for PPP is being requested.
service=slip	Authorization for SLIP is being requested.
protocol=lcp	Authorization for LCP is being requested (lower layer of PPP).
protocol=ip	Used with service=slip to indicate which protocol layer is being authorized.
protocol=ipx	Used with service=ppp to indicate which protocol layer is being authorized.
protocol=atalk	Used with service=ppp or service=arap to indicate which protocol layer is being authorized.
protocol=vines	Used with service=ppp for VINES over PPP.
protocol=unknown	Used for undefined or unsupported conditions.
cmd=*x*	Used with service=shell, if cmd=NULL, this is an authorization request to start an EXEC. If cmd is not NULL, this is a command authorization request and will contain the name of the command being authorized. For example, cmd=telnet.
cmd-arg=*x*	Used with service=shell. When performing command authorization, the name of the command is given by a cmd=*x* pair for each argument listed. For example, cmd-arg=archie.sura.net.
acl=*x*	Used with service=shell and service=arap. For ARA, this pair contains an access list number. For service=shell, this pair contains an access class number. For example, acl=2.
inacl=*x*	Used with service=ppp and protocol=ip. Contains an IP input access list for SLIP or PPP/IP. For example, inacl=2.
outacl=*x*	Used with service=ppp and protocol=ip. Contains an IP output access list for SLIP or PPP/IP. For example, outacl=4.

Attribute Value	Description
addr=*x*	Used with service=slip, service=ppp, and protocol=ip. Contains the IP address that the remote host should use when connecting via SLIP or PPP/IP. For example, addr=172.30.23.11.
routing=*x*	Used with service=slip, service=ppp, and protocol=ip. Equivalent in function to the /routing flag in SLIP and PPP commands. Can either be true or false. For example, routing=true.
timeout=*x*	Used with service=arap. The number of minutes before an ARA session disconnects. For example, timeout=60.
autocmd=*x*	Used with service=shell and cmd=NULL. Specifies an autocommand to be executed at EXEC startup. For example, autocmd=telnet yxz.com.
noescape=*x*	Used with service=shell and cmd=NULL. Specifies a noescape option to the username configuration command. Can be either true or false. For example, noescape=true.
nohangup=*x*	Used with service=shell and cmd=NULL. Specifies a nohangup option to the username configuration command. Can be either true or false. For example, nohangup=false.
priv-lvl=*x*	Used with service=shell and cmd=NULL. Specifies the current privilege level for command authorization as a number from 0 to 15. For example, priv-lvl=15.
zonelist=*x*	Used with service=arap. Specifies an AppleTalk zonelist for ARA. For example, zonelist=5.
addr-pool=*x*	Used with service=ppp and protocol=ip. Specifies the name of a local pool from which to get the address of the remote host.

debug aaa cache filterserver

To help troubleshoot your filter cache configurations, use the **debugaaacachefilterserver** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug aaa cache filterserver

no debug aaa cache filterserver

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(13)T	This command was introduced.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(33)SRC	This command was integrated into Cisco IOS Release 12.2(33)SRC.

Examples

The following is sample output from the **debugaaacachefilterserver** command:

```
Router# debug aaa cache filterserver

AAA/FLTSV: need "myfilter" (fetch), call 0x612DAC64
AAA/FLTSV: send req, call 0x612DAC50
AAA/FLTSV: method SERVER_GROUP myradius
AAA/FLTSV: recv reply, call 0x612DAC50 (PASS)
AAA/FLTSV: create cache
AAA/FLTSV: add attr "call-inacl"
AAA/FLTSV: add attr "call-inacl"
AAA/FLTSV: add attr "call-inacl"
AAA/FLTSV: skip attr "filter-cache-refresh"
AAA/FLTSV: skip attr "filter-cache-time"
AAA/CACHE: set "AAA filtserv cache" entry "myfilter" refresh? no
AAA/CACHE: set "AAA filtserv cache" entry "myfilter" cachetime 15
AAA/FLTSV: add attr to list "call-inacl" call 0x612DAC64
AAA/FLTSV: add attr to list "call-inacl" call 0x612DAC64
AAA/FLTSV: add attr to list "call-inacl" call 0x612DAC64
AAA/FLTSV: PASS call 0x612DAC64
AAA/CACHE: timer "AAA filtserv cache", next in 10 secs (0 entries)
AAA/CACHE: timer "AAA filtserv cache", next in 10 secs (1 entry)
AAA/CACHE: destroy "AAA filtserv cache" entry "myfilter"
```

AAA/CACHE: timer "AAA filtserv cache", next in 10 secs (0 entries)

Related Commands

Command	Description
aaa authorization cache filterserver	Enables AAA authorization caches and the downloading of ACL configurations from a RADIUS filter server.

debug aaa cache group

To debug the caching mechanism and ensure that entries are cached from authentication, authorization, and accounting (AAA) server responses and found when queried, use the **debugaaacachegroup** command in privileged EXEC mode.

debug aaa cache group

Syntax Description

This command has no arguments or keywords.

Command Default

Debug information for all cached entries is displayed.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(28)SB	This command was introduced.
12.2(33)SRC	This command was integrated into Cisco IOS Release 12.2(33)SRC.
15.0(1)M	This command was integrated into Cisco IOS Release 15.0(1)M.

Usage Guidelines

Use this command to display debug information about cached entries.

Examples

The following example displays the debug information about all cached entries:

```
Router# debug aaa cache group
```

Related Commands

Command	Description
clear aaa cache group	Clears an individual entry or all entries in the cache.
show aaa cache group	Displays cache entries stored by the AAA cache.

debug aaa common-criteria

To troubleshoot authentication, authorization, and accounting (AAA) common criteria password security policy information, use the **debug aaa common-criteria** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug aaa common-criteria

no debug aaa common-criteria

Syntax Description

This command has no arguments or keywords.

Command Default

Conditional debugging for this command is disabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
15.0(2)SE	This command was introduced.

Usage Guidelines

Use this command to display debug information about AAA common criteria policies.

Examples

The following example displays the debug information about AAA common criteria:

```
Device> enable
Device# debug aaa common-criteria
AAA common-criteria debugs debugging is on
*Aug  6 08:21:06.554: AAA CC: User flags:2,Policy flags:4
*Aug  6 08:21:06.554: AAA CC: Increment ref count to 1 for test
*Aug  6 08:21:06.554: AAA CC: User test linked to CC policy test
```

Related Commands

Command	Description
aaa common-criteria policy	Configures a AAA common criteria security policy.
show aaa common-criteria policy	Displays common criteria security policy details.

debug aaa dead-criteria transactions

To display authentication, authorization, and accounting (AAA) dead-criteria transaction values, use the **debugaaadead-criteriatransactions**command in privileged EXEC mode. To disable dead-criteria debugging, use the **no** form of this command.

debug aaa dead-criteria transactions

no debug aaa dead-criteria transactions

Syntax Description

This command has no arguments or keywords.

Command Default

If the command is not configured, debugging is not turned on.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(6)	This command was introduced.
12.3(7)T	This command was integrated into Cisco IOS Release 12.3(7)T. The command output includes two new fields: Current Tries and Elapsed Time.

Usage Guidelines

Dead-criteria transaction values may change with every AAA transaction. Some of the values that can be displayed are estimated outstanding transactions, retransmit tries, and dead-detect intervals. These values are explained in the table below.

Examples

The following example shows dead-criteria transaction information for a particular server group:

```
Router# debug aaa dead-criteria transactions
AAA Transaction debugs debugging is on
*Nov 14 23:44:17.403: AAA/SG/TRANSAC: Computed Retransmit Tries: 10, Current Tries: 3,
Current Max Tries: 10
*Nov 14 23:44:17.403: AAA/SG/TRANSAC: Computed Dead Detect Interval: 10s, Elapsed Time:
317s, Current Max Interval: 10s
*Nov 14 23:44:17.403: AAA/SG/TRANSAC: Estimated Outstanding Transactions: 6, Current Max
Transactions: 6
```

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

Table 2: debug aaa dead-criteria transactions Field Descriptions

Field	Description
AAA/SG/TRANSAC	AAA server-group transactions.

Field	Description
Computed Retransmit Tries	Currently computed number of retransmissions before the server is marked as dead.
Current Tries	Number of successive failures since the last valid response.
Current Max Tries	Maximum number of tries since the last successful transaction.
Computed Dead Detect Interval	Period of inactivity (the number of seconds since the last successful transaction) that can elapse before the server is marked as dead. The period of inactivity starts when a transaction is sent to a server that is considered live. The dead-detect interval is the period that the router waits for responses from the server before the router marks the server as dead.
Elapsed Time	Amount of time that has elapsed since the last valid response.
Current Max Interval	Maximum period of inactivity since the last successful transaction.
Estimated Outstanding Transactions	Estimated number of transactions that are associated with the server.
Current Max Transactions	Maximum transactions since the last successful transaction.

Related Commands

Command	Description
radius-server dead-criteria	Forces one or both of the criteria--used to mark a RADIUS server as dead--to be the indicated constant.
show aaa dead-criteria	Displays dead-criteria detection information for an AAA server.

debug aaa per-user

To display debugging information about PPP session per-user activities, use the**debugaaaper-user**command in privileged EXEC mode. To turn off debugging, use the **no** form of this command.

debug aaa per-user

no debug aaa per-user

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
11.3 T	This command has existed since Cisco IOS Release 11.3 T.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

The per-user module is responsible for installing per-user attributes for PPP sessions.

Examples

The following example displays the configuration commands that were generated by the per-user process:

```
Router# debug aaa per-user
AAA/PER-USER: line=[ip access-list standard Virtual-Access2#31]
AAA/PER-USER: line=[deny 10.0.0.2 0.0.0.0]
AAA/PER-USER: line=[permit any]
```

The fields in the display are self-explanatory.

Related Commands

Command	Description
debug aaa authorization	Displays information on AAA TACACS+ authorization.
debug ppp	Displays information on traffic and exchanges in an internetwork implementing the PPP.
debug radius	Displays information associated with RADIUS.
debug tacacs	Displays information associated with TACACS.

debug aaa pod

To display debug messages related to packet of disconnect (POD) packets, use the **debugaaapod** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug aaa pod

no debug aaa pod

Syntax Description

This command has no keywords or arguments.

Command Default

Debugging for POD packets is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.1(3)T	This command was introduced.
12.2(2)XB	Support for the voice applications as well as support for the Cisco AS5350, Cisco AS5400 and the Cisco 3600 series was added.
12.2(2)XB1	Support for the Cisco AS5800 was added.
12.2(11)T	Support for the Cisco AS5850 was added. This command was integrated into Cisco IOS Release 12.2(11)T.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Examples

The following example shows output from a successful POD request when using the **showdebug** command:

```
Router# debug aaa pod
AAA POD packet processing debugging is on
Router# show debug
General OS:
  AAA POD packet processing debugging is on
Router#
Apr 25 17:15:59.318:POD:172.19.139.206 request queued
Apr 25 17:15:59.318:voice_pod_request:
Apr 25 17:15:59.318:voip_populate_pod_attr_list:
Apr 25 17:15:59.318:voip_pod_get_guid:
Apr 25 17:15:59.318:voip_pod_get_vsa_attr_val:
Apr 25 17:15:59.318:voip_pod_get_vsa_attr_val:attr_len=50
Apr 25 17:15:59.318:voip_pod_get_vsa_attr_val:attr=h323-conf-id
Apr 25 17:15:59.318:voip_pod_get_vsa_attr_val:attr_len=50 value_len=35
Apr 25 17:15:59.318:voip_pod_get_guid:conf-id=FFA7785F F7F607BB
00000000 993FB1F4 n_bytes=35
Apr 25 17:15:59.318:voip_pod_get_guid:GUID = FFA7785F F7F607BB 00000000
993FB1F4
Apr 25 17:15:59.318:voip_populate_pod_attr_list:
```

```
Apr 25 17:15:59.318:voip_pod_get_vsa_attr_val:
Apr 25 17:15:59.318:voip_pod_get_vsa_attr_val:attr_len=23
Apr 25 17:15:59.318:voip_pod_get_vsa_attr_val:attr=h323-originate
Apr 25 17:15:59.318:voip_pod_get_vsa_attr_val:attr_len=23 value_len=6
Apr 25 17:15:59.318:voip_get_call_direction:
Apr 25 17:15:59.318:voip_get_call_direction:returning answer
Apr 25 17:15:59.318:voip_eval_pod_attr:
Apr 25 17:15:59.318:cc_api_trigger_disconnect:
Apr 25 17:15:59.322:POD:Sending ACK to 172.19.139.206/1700
Apr 25 17:15:59.322:voip_pod_clean:
```

Related Commands

Command	Description
aaa pod server	Enables the POD feature.

debug aaa redundancy

To display debug output that displays authentication, authorization, and accounting (AAA) redundancy events during session activation, session synchronization to the standby device, and dynamic session updates to the standby device, use the **debugaaaredundancy** command in privileged EXEC mode. To disable debugging for AAA redundancy, use the **no** form of this command.

debug aaa redundancy

no debug aaa redundancy

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(31)SB2	This command was introduced.
12.2(33)SRC	This command was integrated into Cisco IOS Release 12.2(33)SRC.
12.2(33)SRE	This command was integrated into Cisco IOS Release 12.2(33)SRE.

Usage Guidelines

This command displays the AAA synchronization data for the session synchronization to the standby device. This information might be useful for diagnosing any AAA problems related to the session synchronization.

Examples

The following example shows sample output from the **debugaaaredundancy**command collected while a session is activated and synchronized to the standby device:

```
Router# debug aaa redundancy
Logs on Active
==============
01:31:55: CCM: New State[Not Ready]
01:31:55: CCM: PPPoE Required
01:31:55: CCM: PPP Required
01:31:55: CCM: LTERM Required
01:31:55: CCM: PPPoE is Initiator
01:31:55: AAA/BIND(0000000B): Bind i/f Virtual-Template1
01:31:55: CCM: AAA Ready
01:31:55: AAA/CCM/(0000000B): AAA sso init completed successfully
01:31:55: SSS INFO: Element type is Access-Type = 3 (PPPoE)
01:31:55: SSS INFO: Element type is Protocol-Type = 0 (PPP)
01:31:55: SSS INFO: Element type is Media-Type = 1 (Ethernet)
01:31:55: SSS INFO: Element type is Switch-Id = 4105 (00001009)
01:31:55: SSS INFO: Element type is Segment-Hdl = 4114 (00001012)
01:31:55: SSS INFO: Element type is AAA-Id = 11 (0000000B)
01:31:55: SSS INFO: Element type is AccIe-Hdl = 33554441 (02000009)
01:31:55: SSS INFO: Element type is SHDB-Handle = 1476395017 (58000009)
01:31:55: SSS INFO: Element type is Input Interface = "GigabitEthernet6/0/0"
01:31:55: SSS MGR [uid:10]: Sending a Session Assert ID Mgr event
```

```
01:31:55: SSS MGR [uid:10]: Updating ID Mgr with the following keys:
  aaa-unique-id         11 (0xB)
01:31:55: SSS MGR [uid:10]: Handling Policy Service Authorize action (1 pending sessions)
01:31:55: SSS PM [uid:10][63D5D594]: RM/VPDN disabled: RM/VPDN author not needed
01:31:55: SSS PM [uid:10][63D5D594]: AAA author needed for registered user
01:31:55: SSS MGR [uid:10]: Got reply Need More Keys from PM
01:31:55: SSS MGR [uid:10]: Handling Need More Keys action
01:31:57: SSS INFO: Element type is Unauth-User = "user1"
01:31:57: SSS INFO: Element type is AccIe-Hdl = 33554441 (02000009)
01:31:57: SSS INFO: Element type is AAA-Id = 11 (0000000B)
01:31:57: SSS INFO: Element type is Access-Type = 0 (PPP)
01:31:57: SSS INFO: Element type is Protocol-Type = 0 (PPP)
01:31:57: SSS MGR [uid:10]: Sending a Session Update ID Mgr event
01:31:57: SSS MGR [uid:10]: Updating ID Mgr with the following keys:
  username              "user1"
  aaa-unique-id         11 (0xB)
01:31:57: SSS MGR [uid:10]: Handling Policy Send More Keys action
01:31:57: SSS PM [uid:10][63D5D594]: AAA author needed for registered user
01:31:57: SSS PM [uid:10][63D5D594]: SGBP disabled: SGF author not needed
01:31:57: SSS MGR [uid:10]: Got reply Local Terminate from PM
01:31:57: SSS MGR [uid:10]: Handling Connect Local Service action
01:31:57: SSS MGR [uid:10]: Need the resource type determined key
01:31:57: SSS MGR [uid:10]: Handling Need More Keys action
01:31:57: SSS MGR [uid:10]: Not yet ready to start the Local service
01:31:57: AAA/AUTHEN/PPP (0000000B): Pick method list 'default'
01:31:57: RADIUS/ENCODE(0000000B):Orig. component type = PPoE
01:31:57: RADIUS:  AAA Unsupported Attr: client-mac-address[42]  14
01:31:57: RADIUS:   30 30 30 61 2E 34 32 37 64 2E 65 63      [ 000a.427d.ec]
01:31:57: RADIUS:  AAA Unsupported Attr: interface         [171] 7
01:31:57: RADIUS:   36 2F 30 2F 30             [ 6/0/0]
01:31:57: RADIUS(0000000B): Config NAS IP: 0.0.0.0
01:31:57: RADIUS/ENCODE(0000000B): acct_session_id: 11
01:31:57: RADIUS(0000000B): sending
01:31:57: RADIUS/ENCODE: Best Local IP-Address 9.2.76.2 for Radius-Server 9.2.36.253
01:31:57: RADIUS(0000000B): Send Access-Request to 9.2.36.253:1645 id 1645/10, len 86
01:31:57: RADIUS:  authenticator FD E8 32 9A 71 15 50 44 - BE FF 19 D0 09 D4 8D 15
01:31:57: RADIUS:  Framed-Protocol     [7]   6   PPP                       [1]
01:31:57: RADIUS:  User-Name           [1]   9   "user1"
01:31:57: RADIUS:  User-Password       [2]   18  *
01:31:57: RADIUS:  NAS-Port-Type       [61]  6   Virtual                   [5]
01:31:57: RADIUS:  NAS-Port            [5]   6   0
01:31:57: RADIUS:  NAS-Port-Id         [87]  9   "6/0/0/0"
01:31:57: RADIUS:  Service-Type        [6]   6   Framed                    [2]
01:31:57: RADIUS:  NAS-IP-Address      [4]   6   9.2.76.2
01:31:57: RADIUS: Received from id 1645/10 9.2.36.253:1645, Access-Accept, len 32
01:31:57: RADIUS:  authenticator E4 68 43 2C 2F E7 B4 57 - 05 70 FF B1 22 13 E8 0F
01:31:57: RADIUS:  Idle-Timeout        [28]  6   200
01:31:57: RADIUS:  Service-Type        [6]   6   Framed                    [2]
01:31:57: RADIUS(0000000B): Received from id 1645/10
01:31:57: SSS INFO: Element type is Auth-User = "user1"
01:31:57: SSS INFO: Element type is AAA-Attr-List = C5000100
01:31:57: SSS INFO: Element type is   idletime             200 (0xC8)
01:31:57: SSS INFO: Element type is   service-type         2 [Framed]
01:31:57: SSS INFO: Element type is Resource-Determined = 1 (YES)
01:31:57: SSS INFO: Element type is Access-Type = 0 (PPP)
01:31:57: SSS INFO: Element type is Protocol-Type = 0 (PPP)
01:31:57: SSS INFO: Element type is Final = 1 (YES)
01:31:57: SSS MGR [uid:10]: Handling Connect Local Service action
01:31:57: SSS MGR [uid:10]: Rcvd an AAA attr list from SIP, pushing it to the PM
01:31:57: SSS MGR [uid:10]: Handling Send Policy Push Cng action
01:31:57: SSS AAA AUTHOR [uid:10]: Root SIP PPPoE
01:31:57: SSS AAA AUTHOR [uid:10]:  Enable PPPoE parsing
01:31:57: SSS AAA AUTHOR [uid:10]:  Enable PPP parsing
01:31:57: SSS AAA AUTHOR [uid:10]: Active key set to Unauth-User
01:31:57: SSS AAA AUTHOR [uid:10]: Authorizing key user1
01:31:57: SSS AAA AUTHOR [uid:10]: Spoofed AAA reply sent for key user1
01:31:57: SSS MGR [uid:10]: Not yet ready to start the Local service
01:31:57: SSS AAA AUTHOR [uid:10]: Received an AAA pass
01:31:57: SSS AAA AUTHOR [uid:10]: SIP PPP[60A0504C] parsed as Success
01:31:57: SSS AAA AUTHOR [uid:10]: SIP PPP[61571560] parsed as Ignore
01:31:57: SSS AAA AUTHOR [uid:10]: SIP PPPoE[61599FB0] parsed as Success
01:31:57: SSS AAA AUTHOR [uid:10]: SIP Root parser not installed
01:31:57: SSS AAA AUTHOR [uid:10]: No service authorization info found
```

```
01:31:57: SSS AAA AUTHOR [uid:10]: Active Handle present
01:31:57: SSS AAA AUTHOR [uid:10]: Freeing Active Handle; SSS Policy Context Handle =
63D5D594
01:31:57: SSS AAA AUTHOR [uid:10]: Free request
01:31:57: SSS MGR [uid:10]: Got reply Apply Config from PM
01:31:57: SSS MGR [uid:10]: Successfully applied policy config
01:31:57: SSS MGR [uid:10]: Handling Connect Local Service action
01:31:57: CCM: LTERM Required
01:31:57: SSS LTERM [uid:10]: Processing Local termination request
01:31:57: SSS LTERM [uid:10]: Sent create-clone request to vtemplate manager
01:31:57: SSS LTERM [uid:10]: Created vaccess interface Vi3
01:31:57: CCM: LTERM Ready
01:31:57: SSS LTERM [uid:10]: Segment provision successful
01:31:57: SSS MGR [uid:10]: Handling Local Service Connected action
01:31:57: SSS MGR [uid:10]: Apply for Vi3: segment 4114, owner 3825205277
01:31:57: SSS MGR [uid:10]:   Interface config 212C27B8
01:31:57: SSS MGR [uid:10]:   Per-user config 2146BD48
01:31:57: SSS LTERM [uid:10]: Switching session provisioned
01:31:57: SSS MGR [uid:10]: Handling Local Service Connected, Features Applied action
01:31:57: %LINK-3-UPDOWN: Interface Virtual-Access3, changed state to up
01:31:57: SSS LTERM [uid:10]: Installed Vi3 process path switching vector
01:31:57: SSS LTERM [uid:10]: Installed Vi3 fastsend path switching vector
01:31:57: AAA/BIND(0000000B): Bind i/f Virtual-Access3
01:31:57: CCM: PPPoE Ready
01:31:57: CCM: PPP Ready
01:31:57: CCM: PPP Old State[Not Ready] Event[All Ready]
01:31:57: CCM: New State[Ready]
01:31:57: AAA/CCM/(0000000B): No of sync avps = 4 Total sync data len = 94
01:31:57: CCM: PPP Adding Data Type[6] Subtype[0] Length[14]
01:31:57: CCM: PPP Adding Data Type[5] Subtype[0] Length[10]
01:31:57: CCM: PPP Adding Data Type[8] Subtype[0] Length[6]
01:31:57: CCM: PPP Adding Data Type[7] Subtype[0] Length[0]
01:31:57: CCM: PPP Adding Data Type[1] Subtype[0] Length[8]
01:31:57: CCM: PPP Adding Data Type[41] Subtype[0] Length[4]
01:31:57: CCM: PPPoE Adding Data Type[1] Subtype[0] Length[54]
01:31:57: CCM: PPPoE Adding Data Type[2] Subtype[0] Length[2]
01:31:57: CCM: PPPoE Adding Data Type[5] Subtype[0] Length[4]
01:31:57: CCM: PPPoE Adding Data Type[6] Subtype[0] Length[4]
01:31:57: CCM: PPPoE Adding Data Type[7] Subtype[0] Length[20]
01:31:57: CCM: PPPoE Adding Data Type[8] Subtype[0] Length[16]
01:31:57: CCM: AAA Adding Data Type[1] Subtype[0] Length[4]
01:31:57: AAA/CCM/(0000000B): Adding sync avp [AAA Unique Id] Length = 4
01:31:57: CCM: AAA Adding Data Type[2] Subtype[0] Length[2]
01:31:57: AAA/CCM/(0000000B): Adding sync avp [AAA Authen Method Index] Length = 2
01:31:57: CCM: AAA Adding Data Type[3] Subtype[0] Length[4]
01:31:57: AAA/CCM/(0000000B): Adding sync avp [AAA Acct Sess id] Length = 4
01:31:57: CCM: AAA Adding Data Type[4] Subtype[0] Length[84]
01:31:57: AAA/CCM/(0000000B): Adding sync avp [AAA Author Data] Length = 84
01:31:57:  AAA/CCM: Adding author data entry 32
01:31:57: CCM: LTERM Adding Data Type[1] Subtype[0] Length[4]
01:31:57: SSS LTERM [uid:10]: LTERM segment handle synced
01:31:57: CCM:  Send[Sync Session] Length[240] NumItems[17] Event[0x0]
01:31:57:      Client[PPP] Type[6] Subtype[0] Length[14]
01:31:57:       01 04 05 D4 03 04 C0 23 05 06 03 F4 37 79
01:31:57:      Client[PPP] Type[5] Subtype[0] Length[10]
01:31:57:       01 04 05 D4 05 06 9A 6B 68 FE
01:31:57:      Client[PPP] Type[8] Subtype[0] Length[6]
01:31:57:       03 06 07 01 01 01
01:31:57:      Client[PPP] Type[7] Subtype[0] Length[0]
01:31:57:
01:31:57:      Client[PPP] Type[1] Subtype[0] Length[8]
01:31:57:       73 75 6D 61 6E 74 68 00
01:31:57:      Client[PPP] Type[41] Subtype[0] Length[4]
01:31:57:       00 00 00 02
01:31:57:      Client[PPPoE] Type[1] Subtype[0] Length[54]
01:31:57:       00 03 A0 10 22 90 00 0A 42 7D EC 38 88 63 11 19
01:31:57:       00 00 00 22 01 02 00 06 61 61 61 5F 68 61 01 04
01:31:57:       00 10 98 99 BB 6D 59 B8 35 33 0B FB 14 B9 07 EB
01:31:57:       83 B4 01 01 00 00
01:31:57:      Client[PPPoE] Type[2] Subtype[0] Length[2]
01:31:57:       00 0A
01:31:57:      Client[PPPoE] Type[5] Subtype[0] Length[4]
01:31:57:       00 00 10 09
```

```
01:31:57:      Client[PPPoE] Type[6] Subtype[0] Length[4]
01:31:57:       00 00 10 12
01:31:57:      Client[PPPoE] Type[7] Subtype[0] Length[20]
01:31:57:       00 02 06 00 00 00 A6 B8 00 00 00 00 00 00 00 2A
01:31:57:       00 00 FF FF
01:31:57:      Client[PPPoE] Type[8] Subtype[0] Length[16]
01:31:57:       00 00 00 03 00 00 00 00 00 00 00 19 00 00 00 1D
01:31:57:
01:31:57:      Client[AAA] Type[1] Subtype[0] Length[4]
01:31:57:       00 00 00 0B
01:31:57:      Client[AAA] Type[2] Subtype[0] Length[2]
01:31:57:       00 00
01:31:57:      Client[AAA] Type[3] Subtype[0] Length[4]
01:31:57:       00 00 00 0B
01:31:57:      Client[AAA] Type[4] Subtype[0] Length[84]
01:31:57:       00 00 00 00 00 00 00 00 63 E8 73 D0 00 00 00 0B
01:31:57:       64 02 FE 71 00 00 00 00 00 00 00 00 00 00 00 00
01:31:57:       00 00 00 04 00 00 00 01 00 00 00 20 00 00 00 00
01:31:57:       58 00 00 09 02 0A 00 20 E4 68 43 2C 2F E7 B4 57
01:31:57:       05 70 FF B1 22 13 E8 0F 1C 06 00 00 00 C8 06 06
01:31:57:       00 00 00 02
01:31:57:      Client[LTERM] Type[1] Subtype[0] Length[4]
01:31:57:       00 00 20 13
01:31:57: CCM: New State[Dyn Sync]
01:31:58: %LINEPROTO-5-UPDOWN: Line protocol on Interface Virtual-Access3, changed state
to up
Logs on Standby
================
01:21:16: CCM ISSU: Received negotiation message type [ISSU_RC_USER_MESSAGE_COMP]
01:21:16: CCM: Receive[Sync Session] Length[240] NumItems[17] Flags[0x0]
01:21:16: CCM: New State[Not Ready]
01:21:16:      Client[PPP] Type[6] Subtype[0] Length[14]
01:21:16:       01 04 05 D4 03 04 C0 23 05 06 03 F4 37 79
01:21:16:      Client[PPP] Type[5] Subtype[0] Length[10]
01:21:16:       01 04 05 D4 05 06 9A 6B 68 FE
01:21:16:      Client[PPP] Type[8] Subtype[0] Length[6]
01:21:16:       03 06 07 01 01 01
01:21:16:      Client[PPP] Type[7] Subtype[0] Length[0]
01:21:16:
01:21:16:      Client[PPP] Type[1] Subtype[0] Length[8]
01:21:16:       73 75 6D 61 6E 74 68 00
01:21:16:      Client[PPP] Type[41] Subtype[0] Length[4]
01:21:16:       00 00 00 02
01:21:16:      Client[PPPoE] Type[1] Subtype[0] Length[54]
01:21:16:       00 03 A0 10 22 90 00 0A 42 7D EC 38 88 63 11 19
01:21:16:       00 00 00 22 01 02 00 06 61 61 61 5F 68 61 01 04
01:21:16:       00 10 98 99 BB 6D 59 B8 35 33 0B FB 14 B9 07 EB
01:21:16:       83 B4 01 01 00 00
01:21:16:      Client[PPPoE] Type[2] Subtype[0] Length[2]
01:21:16:       00 0A
01:21:16:      Client[PPPoE] Type[5] Subtype[0] Length[4]
01:21:16:       00 00 10 09
01:21:16:      Client[PPPoE] Type[6] Subtype[0] Length[4]
01:21:16:       00 00 10 12
01:21:16:      Client[PPPoE] Type[7] Subtype[0] Length[20]
01:21:16:       00 02 06 00 00 00 A6 B8 00 00 00 00 00 00 00 2A
01:21:16:       00 00 FF FF
01:21:16:      Client[PPPoE] Type[8] Subtype[0] Length[16]
01:21:16:       00 00 00 03 00 00 00 00 00 00 00 19 00 00 00 1D
01:21:16:
01:21:16:      Client[AAA] Type[1] Subtype[0] Length[4]
01:21:16:       00 00 00 0B
01:21:16:      Client[AAA] Type[2] Subtype[0] Length[2]
01:21:16:       00 00
01:21:16:      Client[AAA] Type[3] Subtype[0] Length[4]
01:21:16:       00 00 00 0B
01:21:16:      Client[AAA] Type[4] Subtype[0] Length[84]
01:21:16:       00 00 00 00 00 00 00 00 63 E8 73 D0 00 00 00 0B
01:21:16:       64 02 FE 71 00 00 00 00 00 00 00 00 00 00 00 00
01:21:16:       00 00 00 04 00 00 00 01 00 00 00 20 00 00 00 00
01:21:16:       58 00 00 09 02 0A 00 20 E4 68 43 2C 2F E7 B4 57
01:21:16:       05 70 FF B1 22 13 E8 0F 1C 06 00 00 00 C8 06 06
01:21:16:       00 00 00 02
```

```
01:21:16:      Client[LTERM] Type[1] Subtype[0] Length[4]
01:21:16:       00 00 20 13
01:21:16: CCM:PPPoE Recreate Session  Active[0x58000009] Standby[0x98000009]
01:21:16: CCM: PPPoE Required
01:21:16: CCM: PPP Required
01:21:16: CCM: LTERM Required
01:21:16: CCM: PPPoE is Initiator
01:21:16: AAA/CCM/: return checkpointed aaa id = 0000000B
01:21:16:  Adding cache entry for id B
01:21:16:  Author cache len 84 84 84
01:21:16: AAA/CCM/(0000000B):return acct_sess_id = 11
01:21:16: CCM: AAA Ready
01:21:16: AAA/CCM/(0000000B): AAA sso init completed successfully
01:21:16: SSS INFO: Element type is Access-Type = 3 (PPPoE)
01:21:16: SSS INFO: Element type is Protocol-Type = 0 (PPP)
01:21:16: SSS INFO: Element type is Media-Type = 1 (Ethernet)
01:21:16: SSS INFO: Element type is Switch-Id = 4105 (00001009)
01:21:16: SSS INFO: Element type is Segment-Hdl = 4114 (00001012)
01:21:16: SSS INFO: Element type is AAA-Id = 11 (0000000B)
01:21:16: SSS INFO: Element type is AccIe-Hdl = 4127195145 (F6000009)
01:21:16: SSS INFO: Element type is SHDB-Handle = 2550136841 (98000009)
01:21:16: SSS INFO: Element type is Input Interface = "GigabitEthernet6/0/0"
01:21:16: SSS MGR [uid:10]: Sending a Session Assert ID Mgr event
01:21:16: SSS MGR [uid:10]: Updating ID Mgr with the following keys:
  aaa-unique-id        11 (0xB)
01:21:16: SSS MGR [uid:10]: Handling Policy Service Authorize action (1 pending sessions)
01:21:16: SSS PM [uid:10][63D6963C]: RM/VPDN disabled: RM/VPDN author not needed
01:21:16: SSS MGR [uid:10]: Got reply Need More Keys from PM
01:21:16: SSS MGR [uid:10]: Handling Need More Keys action
01:21:16: SSS INFO: Element type is Unauth-User = "user1"
01:21:16: SSS INFO: Element type is AccIe-Hdl = 4127195145 (F6000009)
01:21:16: SSS INFO: Element type is AAA-Id = 11 (0000000B)
01:21:16: SSS INFO: Element type is Access-Type = 0 (PPP)
01:21:16: SSS INFO: Element type is Protocol-Type = 0 (PPP)
01:21:16: SSS MGR [uid:10]: Sending a Session Update ID Mgr event
01:21:16: SSS MGR [uid:10]: Updating ID Mgr with the following keys:
  username             "user1"
  aaa-unique-id        11 (0xB)
01:21:16: SSS MGR [uid:10]: Handling Policy Send More Keys action
01:21:16: SSS PM [uid:10][63D6963C]: SGBP disabled: SGF author not needed
01:21:16: SSS MGR [uid:10]: Got reply Local Terminate from PM
01:21:16: SSS MGR [uid:10]: Handling Connect Local Service action
01:21:16: SSS MGR [uid:10]: Need the resource type determined key
01:21:16: SSS MGR [uid:10]: Handling Need More Keys action
01:21:16: SSS MGR [uid:10]: Not yet ready to start the Local service
01:21:16: AAA/CCM/(0000000B):return authen_method_index = 0
01:21:16: RADIUS/ENCODE(0000000B):Orig. component type = PPoE
01:21:16: RADIUS:  AAA Unsupported Attr: client-mac-address[42]  14
01:21:16: RADIUS:   30 30 30 61 2E 34 32 37 64 2E 65 63      [ 000a.427d.ec]
01:21:16: RADIUS:  AAA Unsupported Attr: interface         [171] 7
01:21:16: RADIUS:   36 2F 30 2F 30              [ 6/0/0]
01:21:16: RADIUS(0000000B): Config NAS IP: 0.0.0.0
01:21:16: RADIUS/ENCODE(0000000B): acct_session_id: 11
01:21:16: RADIUS(0000000B): sending
01:21:16: RADIUS/ENCODE: Best Local IP-Address 2.1.1.1 for Radius-Server 9.2.36.253
01:21:16: RADIUS(0000000B): Send Access-Request to 9.2.36.253:1645 id 1645/10, len 86
01:21:16: RADIUS:  authenticator 14 48 25 90 A5 7B 53 02 - 11 05 01 13 6D 34 E2 04
01:21:16: RADIUS:  Framed-Protocol     [7]   6   PPP                       [1]
01:21:16: RADIUS:  User-Name           [1]   9   "user1"
01:21:16: RADIUS:  User-Password       [2]   18  *
01:21:16: RADIUS:  NAS-Port-Type       [61]  6   Virtual                   [5]
01:21:16: RADIUS:  NAS-Port            [5]   6   0
01:21:16: RADIUS:  NAS-Port-Id         [87]  9   "6/0/0/0"
01:21:16: RADIUS:  Service-Type        [6]   6   Framed                    [2]
01:21:16: RADIUS:  NAS-IP-Address      [4]   6   2.1.1.1
01:21:16: RADIUS: Cached response
01:21:16: RADIUS:  authenticator E4 68 43 2C 2F E7 B4 57 - 05 70 FF B1 22 13 E8 0F
01:21:16: RADIUS:  Idle-Timeout        [28]  6   200
01:21:16: RADIUS:  Service-Type        [6]   6   Framed                    [2]
01:21:16: RADIUS(0000000B): Received from id 1645/10
01:21:16: SSS INFO: Element type is Auth-User = "user1"
01:21:16: SSS INFO: Element type is AAA-Attr-List = 20000100
01:21:16: SSS INFO: Element type is   idletime             200 (0xC8)
```

```
01:21:16: SSS INFO: Element type is   service-type         2 [Framed]
01:21:16: SSS INFO: Element type is Resource-Determined = 1 (YES)
01:21:16: SSS INFO: Element type is Access-Type = 0 (PPP)
01:21:16: SSS INFO: Element type is Protocol-Type = 0 (PPP)
01:21:16: SSS INFO: Element type is Final = 1 (YES)
01:21:16: SSS MGR [uid:10]: Handling Connect Local Service action
01:21:16: SSS MGR [uid:10]: Rcvd an AAA attr list from SIP, pushing it to the PM
01:21:16: SSS MGR [uid:10]: Handling Send Policy Push Cng action
01:21:16: SSS MGR [uid:10]: Not yet ready to start the Local service
01:21:16: SSS MGR [uid:10]: Got reply Apply Config from PM
01:21:16: SSS MGR [uid:10]: Successfully applied policy config
01:21:16: SSS MGR [uid:10]: Handling Connect Local Service action
01:21:16: CCM: LTERM Required
01:21:16: SSS LTERM [uid:10]: Processing Local termination request
01:21:16: SSS LTERM [uid:10]: Sent create-clone request to vtemplate manager
01:21:16: SSS LTERM [uid:10]: Created vaccess interface Vi3
01:21:16: CCM: LTERM Ready
01:21:16: SSS LTERM [uid:10]: Segment provision successful
01:21:16: SSS MGR [uid:10]: Handling Local Service Connected action
01:21:16: SSS MGR [uid:10]: Apply for Vi3: segment 4114, owner 2566914077
01:21:16: SSS MGR [uid:10]:   Interface config 218170B8
01:21:16: SSS MGR [uid:10]:   Per-user config 63E06550
01:21:16: SSS LTERM [uid:10]: Switching session provisioned
01:21:16: SSS MGR [uid:10]: Handling Local Service Connected, Features Applied action
01:21:16: SSS LTERM [uid:10]: Installed Vi3 process path switching vector
01:21:16: SSS LTERM [uid:10]: Installed Vi3 fastsend path switching vector
01:21:16: CCM: PPPoE Ready
01:21:16: CCM: PPP Ready
01:21:16: CCM: PPP Old State[Not Ready] Event[All Ready]
01:21:16: CCM: New State[Ready]
```

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

Table 3: debug aaa redundancy Field Descriptions

Field	Description
(0000000B)	AAA unique ID for the session.
Adding sync avp	Adding synchronization attribute-value pair.
[AAA Unique ID]	The AAA synchronization data type.

Related Commands

Command	Description
debug ccm-manager	Displays debugging information about Cisco CallManager.

debug aaa sg-server selection

To obtain information about why the RADIUS and TACACS+ server group system in a router is choosing a particular server, use the **debugaaasg-serverselection**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug aaa sg-server selection

no debug aaa sg-server selection

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging is not turned on.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(1)	This command was introduced.
12.2(28)SB	This command was extended for RADIUS server load balancing to show which server is selected on the basis of a load balancing algorithm.
12.4(11)T	This command was integrated into Cisco IOS Release 12.4(11)T.
12.2(33)SRC	This command was integrated into Cisco IOS Release 12.2(33)SRC.

Examples

The following example shows that debugging has been set to display information about server selection:

```
Router# debug aaa sg-server selection
```

The following two debug outputs display the behavior of RADIUS transactions within a server group with the server-reorder-on-failure feature configured.

Examples

In the following sample output, the RADIUS server-reorder-on-failure feature is configured. The server retransmits are set to 0 (so each server is attempted only once before failover to the next configured server), and the transmissions per transaction are set to 4 (the transmissions will stop on the third failover). The third server in the server group (192.0.2.118) has accepted the transaction on the third transmission (second failover).

```
00:38:35: %SYS-5-CONFIG-I: Configured from console by console
00:38:53: RADIUS/ENCODE(OOOOOOOF) : ask "Username: "
00:38:53: RADIUS/ENCODE (0000000F) : send packet; GET-USER
00:38:58: RADIUS/ENCODE (0000000F) : ask "Password: "
00:38:58: RADIUS/ENCODE(0000000F) : send packet; GET-PASSWORD
00:38:59: RADIUS: AAA Unsupported [152] 4
00:38:59: RADIUS: 7474 [tt]
00:38:59: RADIUS (0000000F) : Storing nasport 2 in rad-db
```

```
00:38:59: RADIUS/ENCODE(0000000F) : dropping service type, "radius-server
attribute 6 on-for-login-auth" is off
00:38:59: RADIUS (0000000F) : Config NAS IP: 192.0.2.4
00:38:59: RADIUS/ENCODE (0000000F) : acct-session-id: 15
00:38:59: RADIUS (0000000F) : sending
00:38:59: RADIUS/ENCODE: Best Local IP-Address 192.0.2.130 for Radius-Server 192.0.2.1
00:38:59: RAPIUS(0000000F) : Send Access-Request to 192.0.2.1:1645 id 21645/11, len 78
00:38:59: RADIUS:: authenticator 4481 E6 65 2D 5F 6F OA -lE F5 81 8F 4E 1478 9C
00:38:59: RADIUS: User-Name [1] 7 "username"
00:38:59: RADIUS: User-Password [2] 18 *
00:38:59: RADIUS: NAS-Port fSl 6 2
00:~8:59: RADIUS: NAS-Port-Type [61] 6 Virtual [5]
00:38:59: RADIUS: Calling-Station-Id [31] 15 "192.0.2.23"
00:39:00: RADIUS: NAS-IP-Address [4] 6 192.0.2.130
00:39:02: RADIUS: Fail-over to (192.0.2.2:1645,1646) for id 21645/11
00:39:02: RADIUS/ENCODE: Best Local IP-Address 192.0.2.130 for Radius-Server 192.0.2.2
00:39:04: RADIUS: Fail-over to (192.0.2.118:1645,1646) for id 21645/11
00:39:04: RADIUS/ENCODE: Best Local IP-Address 192.0.2.130 for Radius-Server
192.0.2.118
00:39:05: RADIUS: Received from id 21645/11 192.0.2.118:1645, Access-Accept, len 26
00:39:05: RADIUS: authenticator 5609 56 F9 64 4E DF 19- F3 A2 DD 73 EE 3F 9826
00:39:05: RADIUS: Service-Type [6] 6 Login [1]
```

Examples

In the following sample output, the RADIUS server-reorder-on-failure feature is configured. The server retransmits are set to 0, and the transmissions per transaction are set to 8. In this transaction, the transmission to server 192.0.2.1 has failed on the eighth transmission.

```
00:42:30: RADIUS(00000011): Received from id 21645/13
00:43:34: RADIUS/ENCODE(00000012) : ask "Username: "
00:43:34: RADIUS/ENCODE(00000012) : send packet; GET-USER
00:43:39: RADIUS/ENCODE(00000012) : ask "Password: "
00:43:39: RADIUS/ENCODE(00000012) : send packet; GET-PASSWORD
00:43:40: RADIUS: AAA Unsupported [152] 4
00:43:40: RADIUS: 7474 [tt]
00:43:40: RADIUS(00000012) : Storing nasport 2 in rad-db
00:43:40: RADIUS/ENCODE(00000012): dropping service type, "radius-server attribute 6
on-for-login-auth" is off
00:43:40: RADIUS(00000012) : Co~fig NAS IP: 192.0.2.4
00:43:40: RADIUS/ENCODE(00000012) : acct-session-id: 18
00:43:40: RADIUS(00000012) : sending
00:43:40: RADIUS/ENCODE: Best Local IP-Address 192.0.2.130 for Radius-Server 192.0.2.118
00:43:40: RADIUS(00000012) : Send Access-Request to 192.0.2.118:1645 id 21645/14, len 78
00:43:40: RADIUS: authenticator B8 OA 51 3A AF A6 0018 -B3 2E 94 5E 07 OB 2A
00:43:40: RADIUS: User-Name [1] 7 "username"
00:43:40: RADIUS: User-Password [2] 18 *
00:43:40: RADIUS: NAS-Port [5] 6 2
00:43:40: RADIUS: NAS-Port-Type [61] 6 Virtual [5]
00:43:40: RADIUS: Calling-Station-]d [31] 15 "192.0.2.23"
00:43:40: RADIUS: NAS-IP-Address [4] 6 192.0.2.130
00:43:42: RADIUS: Fail-over to (192.0.2,1:1645,1646) for id 21645/14
00:43:42: RADIUS/ENCODE: Best Local IP-Address 192.0.2.130 for Radius-Server 192.0.2.1
00:43:44: RADIUS: Fail-over to (192.0.2.2:1645,1646) for id 21645/14
00:43:44: RADIUS/ENCODE: Best Local IP-Address 192.0.2.130 for Radius-Server 192.0.2.2
00:43:46: RADIUS: Fail-over to (192.0.2.118:1645,1646) for id 21645/14
00:43:46: RADIUS/ENCODE: Best Local IP-Address 192.0.2.130 for Radius-Server 192.0.2.118
00:43:48: RADIUS: Fail-over to (192.0.2.1:1645,1646) for id 21645/14
00:43:48: RADIUS/ENCODE: Best Local IP-Address 192.0.2.130 for Radius-Server 192.0.2.1
00:43:50: RADIUS: Fail-over to (192.0.2.2:1645,1646) for id 21645/14
00:43:50: RADIUS/ENCODE: Best Local IP-Address 192.0.2.130 for Radius-Server 192.0.2.2
00:43:52: RADIUS: Fail-over to (192.0.2.118:1645,1646) for id 21645/14
00:43:52: RADIUS/ENCODE: Best Local IP-Address 192.0.2.130 for Radius-Server 192.0.2.118
00:43:54: RADIUS: Fail-over to (192.0.2.1:1645,1646) for id 21645/14
00:43:54: RADIUS/ENCODE: Best Local IP-Address 192.0.2.130 for Radius-Server 192.0.2.1
00:43:56: RADIUS: No response from (192.0.2.1:1645,1646) for id 21645/14
00:43:56:RADIUS/DECODE: parse response no app start; FAIL
00:43:56: RADIUS/DECODE: parse response;FAIL
```

The field descriptions are self-explanatory.

Examples

In the following sample output, the RADIUS server load balancing feature is enabled with a batch size of 3. The server selection, based on the load balancing algorithm, is shown as five access-requests that are being sent to the server group.

```
Router# debug aaa sg-server selection
Jul 16 03:15:05: AAA/SG/SERVER_SELECT: Obtaining least loaded server.
Jul 16 03:15:05: AAA/SG/SERVER_SELECT: [3] transactions remaining in batch. Reusing server.
Jul 16 03:15:05: AAA/SG/SERVER_SELECT: Obtaining least loaded server.
Jul 16 03:15:05: AAA/SG/SERVER_SELECT: [2] transactions remaining in batch. Reusing server.
Jul 16 03:15:05: AAA/SG/SERVER_SELECT: Obtaining least loaded server.
Jul 16 03:15:05: AAA/SG/SERVER_SELECT: [1] transactions remaining in batch. Reusing server.
Jul 16 03:15:05: AAA/SG/SERVER_SELECT: Obtaining least loaded server.
Jul 16 03:15:05: AAA/SG/SERVER_SELECT: No more transactions in batch. Obtaining a new server.
Jul 16 03:15:05: AAA/SG/SERVER_SELECT: Obtaining a new least loaded server.
Jul 16 03:15:05: AAA/SG/SERVER_SELECT: Server[0] load: 3
Jul 16 03:15:05: AAA/SG/SERVER_SELECT: Server[1] load: 0
Jul 16 03:15:05: AAA/SG/SERVER_SELECT: Server[2] load: 0
Jul 16 03:15:05: AAA/SG/SERVER_SELECT: Selected Server[1] with load 0
Jul 16 03:15:05: AAA/SG/SERVER_SELECT: [3] transactions remaining in batch.
Jul 16 03:15:05: AAA/SG/SERVER_SELECT: Obtaining least loaded server.
```

Jul 16 03:15:05: AAA/SG/SERVER_SELECT: [2] transactions remaining in batch. Reusing server.

The field descriptions are self-explanatory.

Related Commands

Command	Description
load-balance	Enables RADIUS server load balancing for named RADIUS server groups.
radius-server load-balance	Enables RADIUS server load balancing for the global RADIUS server group.
radius-server retry method reorder	Specifies the reordering of RADIUS traffic retries among a server group.
radius-server transaction max-tries	Specifies the maximum number of transmissions per transaction that may be retried on a RADIUS server.
test aaa group	Tests RADIUS load balancing server response manually.

debug aaa test

To show when the idle timer or dead timer has expired, when test packets are being sent, server response status, and the server state for RADIUS server load balancing, use the **debugaaatest**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug aaa test

no debug aaa test

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(28)SB	This command was introduced.
12.4(11)T	This command was integrated into Cisco IOS Release 12.4(11)T.
12.2(33)SRC	This command was integrated into Cisco IOS Release 12.2(33)SRC.

Examples

In the following sample output, the RADIUS server load balancing feature is enabled. The idle timer has expired.

```
Router# debug aaa test
Router#
Jul 16 00:07:01: AAA/SG/TEST: Server (192.0.2.245:1700,1701) quarantined.
Jul 16 00:07:01: AAA/SG/TEST: Sending test request(s) to server (192.0.2.245:1700,1701)
Jul 16 00:07:01: AAA/SG/TEST: Sending 1 Access-Requests, 1 Accounting-Requests in current
batch.
Jul 16 00:07:01: AAA/SG/TEST(Req#: 1): Sending test AAA Access-Request.
Jul 16 00:07:01: AAA/SG/TEST(Req#: 1): Sending test AAA Accounting-Request.
Jul 16 00:07:01: AAA/SG/TEST: Obtained Test response from server (192.0.2.245:1700,1701)
Jul 16 00:07:01: AAA/SG/TEST: Obtained Test response from server (192.0.2.245:1700,1701)
Jul 16 00:07:01: AAA/SG/TEST: Necessary responses received from server (192.0.2.245:1700,1701)
Jul 16 00:07:01: AAA/SG/TEST: Server (192.0.2.245:1700,1701) marked ALIVE. Idle timer set
for 60 sec(s).
Jul 16 00:07:01: AAA/SG/TEST: Server (192.0.2.245:1700,1701) removed from quarantine.
```

Related Commands

Command	Description
load-balance	Enables RADIUS server load balancing for named RADIUS server groups.

Command	Description
radius-server host	Enables RADIUS automated testing for load balancing.
radius-server load-balance	Enables RADIUS server load balancing for the global RADIUS server group.
test aaa group	Tests RADIUS load balancing server response manually.

debug acircuit

To display errors and events that occur on the attachment circuits (the circuits between the provider edge (PE) and customer edge (CE) routers), use the**debugacircuit** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug acircuit {**error**| **event**}

no debug acircuit {**error**| **event**}

Syntax Description

error	Displays errors that occur in attachment circuits.
event	Displays events that occur in attachment circuits.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(23)S	This command was introduced.
12.2(14)S	This command was integrated into Cisco IOS Release 12.2(14)S.
12.2(15)T	This command was integrated into Cisco IOS Release 12.2(15)T.
12.2(25)S	This command was integrated into Cisco IOS Release 12.2(25)S.
12.2(27)SBC	Support for this command was integrated into Cisco IOS Release 12.2(27)SBC.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

Use the **debugacircuit** command to identify provisioning events, setup failures, circuit up and down events, and configuration failures on attachment circuits.

An attachment circuit connects a PE router to a CE router. A router can have many attachment circuits. The attachment circuit manager controls all the attachment circuits from one central location. Therefore, when you enable the debug messages for the attachment circuit, you receive information about all the attachment circuits.

Examples

The following is sample output from the **debugacircuitevent** command when you enable an interface:

```
Router# debug acircuit event
*Jan 28 15:19:03.070: ACLIB: ac_cstate() Handling circuit UP for interface Se2/0
```

```
*Jan 28 15:19:03.070: ACLIB [11.0.1.1, 200]: pthru_intf_handle_circuit_up() calling
acmgr_circuit_up
*Jan 28 15:19:03.070: ACLIB [11.0.1.1, 200]: Setting new AC state to Ac-Connecting
*Jan 28 15:19:03.070: ACMGR: Receive <Circuit Up> msg
*Jan 28 15:19:03.070: Se2/0 ACMGR: circuit up event, SIP state chg down to connecting,
action is service request
*Jan 28 15:19:03.070: Se2/0 ACMGR: Sent a sip service request
*Jan 28 15:19:03.070: ACLIB [11.0.1.1, 200]: AC updating switch context.
*Jan 28 15:19:03.070: Se2/0 ACMGR: Rcv SIP msg: resp connect forwarded, hdl 9500001D,
l2ss_hdl 700001E
*Jan 28 15:19:03.070: Se2/0 ACMGR: service connected event, SIP state chg connecting to
connected, action is respond forwarded
*Jan 28 15:19:03.070: ACLIB: pthru_intf_response hdl is 9500001D, response is 1
*Jan 28 15:19:03.070: ACLIB [11.0.1.1, 200]: Setting new AC state to Ac-Connected
```

The following is sample output from the **debugacircuitevent** command when you disable an interface:

```
Router# debug acircuit event
*Jan 28 15:25:57.014: ACLIB: SW AC interface INTF-DOWN for interface Se2/0
*Jan 28 15:25:57.014: ACLIB [11.0.1.1, 200]: Setting new AC state to Ac-Idle
*Jan 28 15:25:57.014: ACLIB: SW AC interface INTF-DOWN for interface Se2/0
*Jan 28 15:25:57.014: Se2/0 ACMGR: Receive <Circuit Down> msg
*Jan 28 15:25:57.014: Se2/0 ACMGR: circuit down event, SIP state chg connected to end,
action is service disconnect
*Jan 28 15:25:57.014: Se2/0 ACMGR: Sent a sip service disconnect
*Jan 28 15:25:57.014: ACLIB [11.0.1.1, 200]: AC deleting switch context.
*Jan 28 15:25:59.014: %LINK-5-CHANGED: Interface Serial2/0, changed state to
administratively down
*Jan 28 15:25:59.014: ACLIB: ac_cstate() Handling circuit DOWN for interface Se2/0
*Jan 28 15:26:00.014:%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial2/0, changed
state to down
```

The following example shows output from the **debugacircuit**command for an xconnect session on an Ethernet interface:

```
Router# debug acircuit
23:28:35: ACLIB [10.0.3.201, 5]: SW AC interface UP for Ethernet interface Et2/1
23:28:35: ACLIB [10.0.3.201, 5]: pthru_intf_handle_circuit_up() calling acmgr_circuit_up
23:28:35: ACLIB [10.0.3.201, 5]: Setting new AC state to Ac-Connecting
23:28:35: ACLIB [10.0.3.201, 5]: SW AC interface UP for Ethernet interface Et2/1
23:28:35: ACLIB [10.0.3.201, 5]: pthru_intf_handle_circuit_up() ignoring up event. Already
 connected or connecting.
23:28:35: ACMGR: Receive <Circuit Up> msg
23:28:35: Et2/1 ACMGR: circuit up event, SIP state chg down to connecting, action is service
 request
23:28:35: Et2/1 ACMGR: Sent a sip service request
23:28:37: %LINK-3-UPDOWN: Interface Ethernet2/1, changed state to up
23:28:38: %LINEPROTO-5-UPDOWN: Line protocol on Interface Ethernet2/1, changed state to up

23:28:53: Et2/1 ACMGR: Rcv SIP msg: resp connect forwarded, hdl D6000002, sss_hdl 9E00000F

23:28:53: Et2/1 ACMGR: service connected event, SIP state chg connecting to connected,
action is respond forwarded
23:28:53: ACLIB: pthru_intf_response hdl is D6000002, response is 1
23:28:53: ACLIB [10.0.3.201, 5]: Setting new AC state to Ac-Connected
```

The command output is self-explanatory.

Related Commands

Command	Description
debug vpdn	Displays errors and events relating to L2TP configuration and the surrounding Layer 2 tunneling infrastructure.
debug xconnect	Displays errors and events related to an xconnect configuration.

debug acircuit checkpoint

To enable the display of attachment circuit checkpoints, use the debug acircuit checkpoint command in privileged EXEC mode. To disable the display of these messages, use the no form of this command.

debug acircuit checkpoint

no debug acircuit checkpoint

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging of attachment circuit checkpoints is disabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(25)S	This command was introduced.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(33)SRC	This command was integrated into Cisco IOS Release 12.2(33)SRC.

Usage Guidelines

Use this command when Any Transport over MPLS (AToM) is configured for nonstop forwarding/stateful switchover (NSF/SSO) and Graceful Restart.

Use debug commands with caution. They use a significant amount of CPU resources and can affect system performance.

Examples

The **debugacircuitcheckpoint** command is issued on the active route processor (RP):

```
Router# debug mpls l2transport checkpoint
Router# debug acircuit checkpoint
Router# show debug
AToM HA:
  AToM checkpointing events and errors debugging is on
AC HA:
  Attachment Circuit Checkpoint debugging is on
Router# conf terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# interface Fa5/1/1.2
Router(config-subif)# xconnect 10.55.55.2 1002 pw-class mpls
AToM HA [10.55.55.2, 1002]: Build provision msg, SSM sw/seg 8192/8194 [0x2000/0x2002] PW
id 9216 [0x2400] local label 21
AC HA: Dynamic Sync. Event:4 Sw:8192[2000] Se:16385[4001]
AToM HA: CF sync send complete
AC HA CF: Sync send complete. Code:0
```

On the standby RP, the following messages indicate that it receives checkpointing data:

```
AC HA [10.55.55.2, 1002]: Add to WaitQ. Flags:1
AToM HA [10.55.55.2, 1002]: Received 32-byte provision version 1 CF message
AC HA CF: ClientId:89, Entity:0 Length:40
AToM HA [10.55.55.2, 1002]: Process chkpt msg provision [1], ver 1
AToM HA [10.55.55.2, 1002]: Reserved SSM sw/seg 8192/8194 [0x2000/0x2002] PW id 9216 [0x2400]
AC HA: Process Msg:35586. Ptr:44CBFD90. Val:0
AC HA: Sync. Event:4 CktType:4 Sw:8192[2000] Se:16385[4001]
AC HA [10.55.55.2, 1002]: Remove from WaitQ. Flags:1[OK][OK]
```

During a switchover from an active RP to a backup RP, the debug messages look similar to the following:

```
%HA-5-MODE: Operating mode is hsa, configured mode is sso.
AC HA RF: CId:83, Seq:710, Sta:RF_STATUS_OPER_REDUNDANCY_MODE_CHANGE, Opr:5, St:STANDBY
HOT, PSt:ACTIVE
AToM HA: CID 84, Seq 715, Status RF_STATUS_OPER_REDUNDANCY_MODE_CHANGE, Op 5, State STANDBY
 HOT, Peer ACTIVE
AC HA RF: CId:83, Seq:710, Sta:RF_STATUS_PEER_PRESENCE, Opr:0, St:STANDBY HOT, PSt:ACTIVE
AToM HA: CID 84, Seq 715, Status RF_STATUS_PEER_PRESENCE, Op 0, State STANDBY HOT, Peer
ACTIVE
AC HA RF: CId:83, Seq:710, Sta:RF_STATUS_PEER_COMM, Opr:0, St:STANDBY HOT, PSt:DISABLED
AToM HA: CID 84, Seq 715, Status RF_STATUS_PEER_COMM, Op 0, State STANDBY HOT, Peer DISABLED
%HA-2-CUTOVER_NOTICE: Cutover initiated. Cease all console activity until system restarts.
%HA-2-CUTOVER_NOTICE: Do not add/remove RSPs or line cards until switchover completes.
%HA-2-CUTOVER_NOTICE: Deinitializing subsystems...
%OIR-6-REMCARD: Card removed from slot 4, interfaces disabled
%OIR-6-REMCARD: Card removed from slot 5, interfaces disabled
%OIR-6-REMCARD: Card removed from slot 9, interfaces disabled
%HA-2-CUTOVER_NOTICE: Reinitializing subsystems...
%HA-2-CUTOVER_NOTICE: System preparing to restart...
%HA-5-NOTICE: Resuming initialization...
AC HA RF: CId:83, Seq:710, Sta:RF_STATUS_REDUNDANCY_MODE_CHANGE, Opr:7, St:STANDBY HOT,
PSt:DISABLED
.
.
.
%LDP-5-GR: LDP restarting gracefully.  Preserving forwarding state for 250 seconds.
AC HA RF: CId:83, Seq:710, Sta:RF_PROG_ACTIVE, Opr:0, St:ACTIVE, PSt:DISABLED
AToM HA: CID 84, Seq 715, Event RF_PROG_ACTIVE, Op 0, State ACTIVE, Peer DISABLED
AC HA: Process Msg:35588. Ptr:0. Val:0
AC HA: Switchover: Standby->Active
AC HA RF: Reconciling
```

Related Commands

Command	Description
debug mpls l2transport checkpoint	Enables the display of AToM events when AToM is configured for NSF/SSO and Graceful Restart.

debug adjacency

To enable the display of information about the adjacency database, use the **debug adjacency**command in privileged EXEC mode. To disable the display of these events, use the **no** form of this command.

debug adjacency [**epoch**| **ipc**| **state**| **table**] [*prefix*] [*interface*] [**connectionid** *id*] [**link** {**ipv4**| **ipv6**| **mpls**}]

no debug adjacency [**epoch**| **ipc**| **state**| **table**] [*prefix*] [*interface*] [**connectionid** *id*] [**link** {**ipv4**| **ipv6**| **mpls**}]

Syntax Description

epoch	(Optional) Displays adjacency epoch events.
ipc	(Optional) Displays interprocess communication (IPC) events for adjacencies.
state	(Optional) Displays adjacency system state machine events.
table	(Optional) Displays adjacency table operations.
prefix	(Optional) Displays debugging events for the specified IP address or IPv6 address. **Note** On the Cisco 10000 series routers, IPv6 is supported in Cisco IOS Release 12.2(28)SB and later releases.
interface	(Optional) Displays debugging events for the specified interface. For line cards, you must specify the line card if_number (interface number). Use the **show cef interface** command to obtain line card if_numbers.
connectionid *id*	(Optional) Displays debugging events for the specified client connection identification number.
link {**ipv4** | **ipv6** | **mpls**	(Optional) Displays debugging events for the specified link type (IP, IPv6, or Multiprotocol Label Switching [MPLS] traffic). **Note** On the Cisco 10000 series routers, IPv6 is supported in Cisco IOS Release 12.2(28)SB and later releases.

Command Default

Debugging events are not displayed.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(7)XE	This command was introduced on the Cisco 7600 series routers.
12.1(1)E	This command was implemented on the Cisco 7600 series routers.
12.2(14)SX	This command was implemented on the Supervisor Engine 720.
12.2(25)S	This command was integrated into Cisco IOS Release 12.2(25)S, and the *prefix* , *interface* , **connectionid***id* , and **link** {**ipv4** | **ipv6** | **mpls**} keywords and arguments were added.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB and implemented on the Cisco 10000 series routers.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

Because debugging output is assigned high priority in the CPU process, you should use debug commands only to troubleshoot specific problems or during troubleshooting sessions with Cisco technical support staff. Also, you should use debug commands during periods of lower network traffic and fewer users. Debugging during these periods decreases the likelihood that increased debug command processing overhead will affect system use.

You can use any combination of the *prefix* , *interface* , **connectionid***id* , and **link** {**ipv4** | **ipv6** | **mpls**} keywords and arguments (in any order) as a filter to enable debugging for a specified subset of adjacencies.

Note On the Cisco 10000 series routers, IPv6 is supported in Cisco IOS Release 12.2(28)SB and later releases.

Examples

The following example shows how to display information on the adjacency database:

```
Router# debug adjacency
*Jan 27 06:22:50.543: ADJ-ios_mgr: repopulate adjs on up event for Ethernet3/0
*Jan 27 06:22:50.543: ADJ: IPV6 adj out of Ethernet3/0, addr FE80::20C:CFFF:FEDF:6854
(incomplete) no src set: init/update from interface
*Jan 27 06:22:50.543: ADJ: IPV6 adj out of Ethernet3/0, addr FE80::20C:CFFF:FEDF:6854
(incomplete) no src set: set bundle to IPv6 adjacency oce
*Jan 27 06:22:50.543: ADJ: IPV6 adj out of Ethernet3/0, addr FE80::20C:CFFF:FEDF:6854
(incomplete) no src set: allocated, setup and inserted OK
*Jan 27 06:22:50.543: ADJ: IPV6 adj out of Ethernet3/0, addr FE80::20C:CFFF:FEDF:6854
(incomplete) src IPv6 ND: source IPv6 ND added OK
*Jan 27 06:22:50.543: ADJ: IPV6 adj out of Ethernet3/0, addr FE80::20C:CFFF:FEDF:6854
(incomplete) src IPv6 ND: computed macstring (len 14): OK
*Jan 27 06:22:50.543: ADJ: IPV6 adj out of Ethernet3/0, addr FE80::20C:CFFF:FEDF:6854 src
IPv6 ND: made complete (macstring len 0 to 14/0 octets)
00:04:40: %LINK-3-UPDOWN: Interface Ethernet3/0, changed state to up
00:04:41: %LINEPROTO-5-UPDOWN: Line protocol on Interface Ethernet3/0, changed
```

Related Commands

Command	Description
clear adjacency	Clears the Cisco Express Forwarding adjacency table.
clear arp-cache	Deletes all dynamic entries from the ARP cache.
show adjacency	Displays Cisco Express Forwarding adjacency table information.
show mls cef adjacency	Displays information about the hardware Layer 3 switching adjacency node.

debug adjacency (vasi)

To display debugging information for the VRF-Aware Service Infrastructure (VASI) adjacency, use the **debugadjacency** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug adjacency {**vasileft**| **vasiright**} *number*

no debug interface {**vasileft**| **vasiright**} *number*

Syntax Description

vasileft	Displays information about the vasileft interface.
vasiright	Displays information about the vasiright interface.
number	Identifier of the VASI interface. The range is from 1 to 256.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
Cisco IOS XE Release 2.6	This command was introduced.

Examples

The following is sample output from the **debugadjacency**command:

```
Router# debug adjacency veasileft 1
Condition 1 set
```

Related Commands

debug vasi	Displays debugging information for the VASI.
debug interface (vasi)	Displays debugging information for the VASI interface descriptor block.
interface (vasi)	Configures VASI virtual interface.
show vasi pair	Displays the status of a VASI pair.

debug alarm-interface

To show real-time activities in the data channel or the management channel of the Alarm Interface Controller (AIC), use the **debugalarm-interface** command in privileged EXEC mode. To disable debugging output, use the**no** form of this command.

debug alarm-interface *slot-number* {**data**| **management**}

no debug alarm-interface *slot-number* {**data**| **management**}

Syntax Description

slot-number	Router chassis slot where the AIC network module is installed.
data	Displays AIC serial data channel and asynchronous craft port communication activity.
management	Displays IOS-to-AIC communication activity.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(2)XG	This command was introduced for the Cisco 2600 series and the Cisco 3600 series.
12.2(8)T	This command was integrated into Cisco IOS Release 12.2(8)T.

Usage Guidelines

This command allows you to observe the management channel activity from the AIC in the specified slot. Such activity shows that the software running on the AIC CPU has reached a minimum level of working order.

Examples

The following is sample output from the **debugalarm-interface** command:

```
Router# debug alarm-interface
AIC Slot 1:STATUS received
```

The following is sample output from the **debugalarm-interface1data** command:

```
Router# debug alarm-interface 1 data
AIC Slot 1:STATUS received
aic_fastsend:particle count=1, len=1504
aic_pak_to_txring:scattered particle count=1, tx bytes=1504, leftover=0
aic_interrupt:# 30419  gstar=0x1000000
```

```
aic_safe_start:particle count=1, len=524
aic_pak_to_txring:scattered particle count=1, tx bytes=524, leftover=0
aic_process_TXivq:ivq - 0x42040000 at 15, slice 1
aic_interrupt:# 30420  gstar=0x1000000
aic_process_TXivq:ivq - 0x42040000 at 16, slice 1
aic_interrupt:# 30421  gstar=0x10000000
aic_scc_rx_intr:sts_dlen=0xC5E10000, len=1504, RSTA=0xA0
aic_serial_RX_interrupt:rxtype=1, len=1504, aic_scc_rx_intr:last_rxbd has aged, 2
aic_process_RXivq:ivq - 0x60000    at 13, slice 1
aic_interrupt:# 30422  gstar=0x10000000
aic_scc_rx_intr:sts_dlen=0xC20D0000, len=524, RSTA=0xA0
aic_serial_RX_interrupt:rxtype=1, len=524, aic_process_RXivq:ivq - 0x60000    at 14,
slice 1
aic_interrupt:# 30423  gstar=0x20000000
aic_scc_rx_intr:sts_dlen=0xC00D0000, len=12, RSTA=0xA0
aic_mgmt_RX_interrupt:len=12
aic_mgmt_fastsend:particle count=1, len=20 / 20
aic_pak_to_txring:scattered particle count=1, tx bytes=20, leftover=0
aic_scc_rx_intr:last_rxbd has aged, 2
aic_process_RXivq:ivq - 0x10060000 at 37, slice 1
aic_interrupt:# 30424  gstar=0x2000000
aic_process_TXivq:ivq - 0x52040000 at 24, slice 1
```

Related Commands

Command	Description
alarm-interface	Enters the alarm interface mode and configures the AIC.
reset	Resets the AIC CPU.

debug alps ascu

To enable debugging for airline product set (ALPS) agent set control units (ASCUs) use the **debugalpsascu** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug alps ascu {**event**| **packet**| **detail**| **all**| **format** {**ipars**| **router**| **both**}} [*interface* [*ascu id*]]

no debug alps ascu {**event**| **packet**| **detail**| **all**| **format** {**ipars**| **router**| **both**}} [*interface* [*ascu id*]]

Syntax Description

event	Displays ASCU events or protocol errors.
packet	Displays sent or received packets.
detail	Displays all ASCU protocol events.
all	Enables event, packet, and detail debugging.
format ipars router both	Specifies how to display ASCU addresses and the hexadecimal data in the debug output: • **ipars--** Displays only the IPARS hexadecimal output. • **router--** Displays only the router hexadecimal output. • **both--** Displays both the IPARS and router hexadecimal output. The only difference between the IPARS output and the router output is the format of the hexadecimal data.
interface	(Optional) Enables debugging on a specified interface. Applies only to the **event**, **packet**, **detail**, and **all** keywords.
ascu id	(Optional) Enables debugging for a specified ASCU.

Command Default

Debugging is off.

Command Modes

Privileged EXEC

Command History

Release	Modification
11.3(6)T	This command was introduced for limited availability.
12.0(1)	This command was available for general release.
12.0(5)T	This command was modified.
12.1(2)T	The **format**, **ipars**, **router**, and **both** keywords were added. The output for this command was modified to include IPARS and router formats.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

To enable debugging for a group of ASCUs, enter a separate command for each ASCU interface and IA combination.

The *interface* option applies only to the **event**, **packet**, **detail**, and **all** keywords.

Note To specify the particular debug tracing level (**event**, **packet**, **detail** or **all**) *and* the format (router, pairs or both), you must configure the **debugalpsascu**command two times: once to configure the debug tracing level and once to configure the format.

Examples

The following output is from the **debugalpsascuevent** command, showing events or protocol errors in **router** format for ASCU 42 on interface Serial7:

```
Router# debug alps ascu format router

Router# debug alps ascu event Serial7 42

ALPS ASCU: T1 expired for ascu 42 on i/f Serial7
ALPS ASCU: DOWN event while UP for ascu 42 on i/f Serial7 : C1 count = 1
```

Note If you specify the **ipars** or **both** format for the **event** or **detail** tracing level, both the IPARS and router formats will be displayed.

The following output is from the **debugalpsascuevent** command, showing events or protocol errors in **ipars** format for ASCU 42 on interface Serial7:

```
Router# debug alps ascu format ipars

Router# debug alps ascu event Serial7 42

ALPS ASCU: T1 expired for ascu 42/2F on i/f Serial7
ALPS ASCU: DOWN event while UP for ascu 42/2F on i/f Serial7 : C1 count = 1
```

The following output is from the **debugalpsascudetail** command, showing all protocol events in **router** format for ASCU 42 on interface Serial6:

```
Router# debug alps ascu format router

Router# debug alps ascu detail Serial6 42

ALPS ASCU: Tx ALC POLL MSG (+ 0 pad bytes) to ascu 42 on i/f Serial6
ALPS ASCU: ALC GO AHD MSG rcvd from ascu 42 on i/f Serial6
ALPS ASCU: Tx ALC POLL MSG (+ 0 pad bytes) to ascu 42 on i/f Serial6
ALPS ASCU: ALC GO AHD MSG rcvd from ascu 42 on i/f Serial6
ALPS ASCU: Tx ALC POLL MSG (+ 0 pad bytes) to ascu 42 on i/f Serial6
ALPS ASCU: Rx ALC DATA MSG (14 bytes + CCC) from ascu 42 on i/f Serial6, fwd to ckt
RTP_MATIP
ALPS ASCU: ALC GO AHD MSG rcvd from ascu 42 on i/f Serial6
ALPS ASCU: Tx ALC DATA MSG (14 bytes + CCC + 0 pad bytes) to ascu 42 on i/f Serial6
ALPS ASCU: Tx ALC POLL MSG (3 bytes + CCC + 0 pad bytes) to ascu 42 on i/f Serial6
```

Note If you specify the **ipars** or **both** format for the **event** or **detail** tracing level, both the IPARS and router formats will be displayed.

The following output is from the **debugalpsascudetail** command, showing all protocol events in **both** format for ASCU 42 on interface Serial6:

```
Router# debug alps ascu format both

Router# debug alps ascu detail Serial6 42

ALPS ASCU: Tx ALC POLL MSG (+ 0 pad bytes) to ascu 42/2F on i/f Serial6
ALPS ASCU: ALC GO AHD MSG rcvd from ascu 42/2F on i/f Serial6
ALPS ASCU: Tx ALC POLL MSG (+ 0 pad bytes) to ascu 42/2F on i/f Serial6
ALPS ASCU: ALC GO AHD MSG rcvd from ascu 42/2F on i/f Serial6
ALPS ASCU: Tx ALC POLL MSG (+ 0 pad bytes) to ascu 42/2F on i/f Serial6
ALPS ASCU: Rx ALC DATA MSG (14 bytes + CCC) from ascu 42/2F on i/f Serial6, fwd to ckt
RTP_MATIP
ALPS ASCU: ALC GO AHD MSG rcvd from ascu 42/2F on i/f Serial6
ALPS ASCU: Tx ALC DATA MSG (14 bytes + CCC + 0 pad bytes) to ascu 42/2F on i/f Serial6
ALPS ASCU: Tx ALC POLL MSG (3 bytes + CCC + 0 pad bytes) to ascu 42/2F on i/f Serial6
```

The following output is from the **debugalpsascupacket** command, showing all packets sent or received in **router** format for ASCU 42 on interface Serial6:

```
Router# debug alps ascu packet format router Serial6 42

ALPS ASCU: Tx ALC SERVICE MSG (18 bytes + CCC + 0 pad bytes) to ascu 42 on i/f Serial6
02321D26 0C261616
140C0D18 26163135 0611C6
ALPS ASCU: Rx ALC DATA MSG (14 bytes + CCC) from ascu 42 on i/f Serial6, fwd ckt
RTP_MATIP
42607866 65717866
65717966 755124
ALPS ASCU: Tx ALC DATA MSG (14 bytes + CCC + 0 pad bytes) to ascu 42 on i/f Serial6
022038 26253138
26253139 263511E4
```

The following output is from the **debugalpsascupacket** command, showing all packets sent or received in **ipars** format for ASCU 42 on interface Serial6:

```
Router# debug alps ascu packet format ipars Serial6 42

ALPS ASCU: Tx ALC SERVICE MSG (18 bytes + CCC + 0 pad bytes) to ascu 42/2F on i/f Serial6
ALPS IPARS Format:
2F2C1126 33262525
35331339 26251C14 271DC6
ALPS ASCU: Rx ALC DATA MSG (14 bytes + CCC) from ascu 42/2F on i/f Serial6, fwd ckt
RTP_MATIP
```

```
ALPS IPARS Format:
2F3E3826 161C3826
161C1826 141D24
ALPS ASCU: Tx ALC DATA MSG (14 bytes + CCC + 0 pad bytes) to ascu 42/2F on i/f Serial6
ALPS IPARS Format:
2F3E38 26161C38
26161C18 26141DE4
```

The following output is from the **debugalpsascupacket** command, showing all packets sent or received in **both** formats for ASCU 42 on interface Serial6:

```
Router# debug alps ascu packet format both Serial6 42

ALPS ASCU: Tx ALC SERVICE MSG (18 bytes + CCC + 0 pad bytes) to ascu 42/2F on i/f Serial6
ALPS Router Format:
02321D26 0C261616
140C0D18 26163135 0611C6
ALPS IPARS Format:
2F2C1126 33262525
35331339 26251C14 271DC6
ALPS ASCU: Rx ALC DATA MSG (14 bytes + CCC) from ascu 42/2F on i/f Serial6, fwd ckt
RTP_MATIP
ALPS Router Format:
42607866 65717866
65717966 755124
ALPS IPARS Format:
2F3E3826 161C3826
161C1826 141D24
ALPS ASCU: Tx ALC DATA MSG (14 bytes + CCC + 0 pad bytes) to ascu 42/2F on i/f Serial6
ALPS Router Format:
022038 26253138
26253139 263511E4
ALPS IPARS Format:
2F3E38 26161C38
26161C18 26141DE4
```

debug alps circuit event

To enable event debugging for airline product set (ALPS) circuits, use the **debugalpscircuitevent**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug alps circuit event [*name*]

no debug alps circuit event [*name*]

Syntax Description

name	(Optional) Name given to identify an ALPS circuit on the remote customer premises equipment (CPE).

Command Default

If no circuit name is specified, then debugging is enabled for every ALPS circuit.

Command Modes

Privileged EXEC

Command History

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

Usage Guidelines

To enable debugging for a single ALPS circuit, specify the name of the circuit.

To enable debugging for a group of circuits, enter a separate command for each circuit name.

Examples

The following is sample output from the **debugalpscircuitevent** command for circuit RTP_AX25:

```
alps-rcpe# debug alps circuit event RTP_AX25

ALPS P1024 CKT: FSM - Ckt= RTP_AX25, State= OPEN, Event= DISABLE:
(CloseAndDisable)->DISC
ALPS P1024 CKT: FSM - Ckt= RTP_AX25, State= DISC, Event= ENABLE:
(TmrStartNullRetry)->INOP
ALPS P1024 CKT: Ckt= RTP_AX25, Open - peer set to 200.100.40.2
ALPS P1024 CKT: Ckt= RTP_AX25, Open - peer open.
ALPS P1024 CKT: FSM - Ckt= RTP_AX25, State= INOP, Event= RETRY_TIMEOUT:
(Open)->OPNG
ALPS P1024 CKT: FSM - Ckt= RTP_AX25, State= OPNG, Event= CKT_OPEN_CFM:
(CacheAndFwdAscuData)->OPEN
alps-ccpe# debug alps circuit event RTP_AX25

ALPS AX.25 FSM: Ckt= RTP_AX25, State= OPEN, Event= CktClose, Rsn= 12:
(PvcKill,CktRemove,TmrStartClose)->INOP
ALPS AX.25 FSM: Ckt= RTP_AX25, State= INOP, Event= X25PvcInact, Rsn= 0:
(-,-,-)->INOP
ALPS AX.25 FSM: Ckt= RTP_AX25, State= INOP, Event= X25VcDeleted, Rsn= 0:
```

```
(-,CktDestroy,TmrStop)->INOP
ALPS AX.25 FSM: Ckt= RTP_AX25, State= INOP, Event= CktOpReq, Rsn= 4:
(PvcMake,CktAdd,TmrStartOpen)->OPNG
ALPS AX.25 FSM: Ckt= RTP_AX25, State= OPNG, Event= X25ResetTx, Rsn= 0:
(-,-,-)->OPNG
ALPS AX.25 FSM: Ckt= RTP_AX25, State= OPNG, Event= X25VcUp, Rsn= 0:
(-,OpnCfm,TmrStop)->OPEN
```

debug alps peer

To enable event or packet debugging for airline product set (ALPS) peers, use the **debugalpspeer**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug alps peer {**event**| **packet**} [*ip-address*]

no debug alps peer {**event**| **packet**} [*ip-address*]

Syntax Description

event	Specifies debugging for an event.
packet	Specifies debugging for a packet.
ip-address	(Optional) Remote peer IP address.

Command Default

If no IP address is specified, then debugging is enabled for every peer connection.

Command Modes

Privileged EXEC

Command History

Release	Modification
11.3(6)T	This command was introduced for limited availability.
12.0(1)	This command was available for general release.
12.0(5)T	The **packet** keyword was added. The format for the output was modified for consistency.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

To enable debugging for a single remote ALPS peer, specify the peer IP address.

To enable debugging for a set of remote peers, enter the command for each peer IP address.

Examples

The following is sample output from the **debugalpspeerpacket** command:

```
Router# debug alps peer packet

ALPS PEER:Peer (10.227.50.106, MATIP_A_CKT-1) - TX Peer Data Msg (18 bytes)
040A5320:                        01 00001241
040A5330:45546B5F 6F4F7757 67477B5B 51
ALPS PEER:Peer (10.227.50.106, MATIP_A_CKT-1) - RX Peer Data Msg (18 bytes)
04000550:         01000012 4145546B 5F6F4F77
04000560:5767477B 5B51
```

```
ALPS PEER:Peer (10.227.50.106, MATIP_A_CKT-1) - TX Peer Data Msg (18 bytes)
0409F6E0:                01 00001241 45546B5F
0409F6F0:6F4F7757 67477B5B 51
ALPS PEER:Peer (10.227.50.106, MATIP_A_CKT-1) - RX Peer Data Msg (18 bytes)
04000680:                  01000012 4145546B
04000690:5F6F4F77 5767477B 5B51
```

debug alps peer event

To enable event debugging for airline product set (ALPS) peers, use the **debugalpspeerevent**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug alps peer event *ipaddr*

no debug alps peer event *ipaddr*

Syntax Description

ipaddr	Peer IP address.

Command Default

If no IP address is specified, debugging is enabled for every peer connection.

Command Modes

Privileged EXEC

Command History

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

Usage Guidelines

To enable debugging for a single remote ALPS peer, specify the peer IP address.

To enable debugging for a set of remote peers, enter the command for each peer IP address.

Examples

The following is sample output from the **debugalpspeerevent** command:

```
Router# debug alps peer event
ALPS PEER: FSM - Peer 200.100.25.2, Event ALPS_CLOSED_IND, State OPENED
ALPS PEER: peer 200.100.25.2 closed - closing peer circuits.
ALPS PEER: Promiscuous peer created for 200.100.25.2
ALPS PEER: TCP Listen - passive open 200.100.25.2(11003) -> 10000
ALPS PEER: FSM - Peer 200.100.25.2, Event ALPS_OPEN_IND, State DISCONN
ALPS PEER: peer 200.100.25.2 opened OK.
```

debug alps snmp

To enable debugging for airline product set (ALPS) Simple Network Management Protocol (SNMP) agents, use the **debugalpssnmp**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug alps snmp

no debug alps snmp

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging for SNMP agents is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
11.3(6)T	This command was introduced for limited availability.
12.0(1)T	This command was available for general release.
12.0(5)T	This command was added to the documentation.
12.1(2)T	The output for this command was modified to reflect MIB and SNMP changes.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Examples

The following output is from the **debugalpssnmp** command. The first line shows a circuit event status change. The second line shows an ASCU status change. The third line shows a peer connection status change.

```
Router# debug alps snmp

ALPS CktStatusChange Notification for circuit CKT-1
ALPS AscuStatusChange Notification for ascu (Serial3, 41)
PeerConnStatusChange Notification for peer (10.227.50.106, MATIP_A_CKT-1)
```

The following output shows that an open failure has occurred on circuit 1:

```
ALPS CktOpenFailure Notification for circuit CKT1
```

The following output shows that a partial rejection to an ALPS circuit peer open request has occurred on circuit 1:

```
ALPS CktPartialReject Notification for ascu (Serial2, 41) on circuit CKT1
```

debug ancp

To enable the display of debugging information related to the Access Node Control Protocol (ANCP), use the **debugancp**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ancp {**adjacency**| **details**| **errors**| **events**| **neighbor**| **packets** [**brief**]| **port-event** {**details**| **events**}| **port-management** {**details**| **events**}}

no debug ancp {**adjacency**| **details**| **errors**| **events**| **neighbor**| **packets** [**brief**]| **port-event** {**details**| **events**}| **port-management** {**details**| **events**}}

Syntax Description

adjacency	Displays information about adjacency messages on an ANCP server.
details	Displays detailed static configuration information relating to ANCP and dynamic line conditions.
errors	Displays information about ANCP protocol errors.
events	Displays information about ANCP protocol events.
neighbor	Displays information about ANCP neighbors.
packets	Displays information about ANCP control packets.
brief	(Optional) Displays static configuration information for ANCP control packets.
port-event	Displays ANCP event messages (port up and port down) related to data ports.
port-management	Displays ANCP operations, administration, and maintenance (OAM) messages.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(28)ZV	This command was introduced on the Cisco 10000 series router.
12.2(31)ZV1	This command was modified. L2CP was replaced with ANCP.
Cisco IOS XE Release 2.4	This command was integrated into Cisco IOS XE Release 2.4.

Release	Modification
Cisco IOS XE Release 3.2S	This command was modified. The **port-event** and **port-management** keywords were added.

Usage Guidelines

You can use the **debugconditioninterface** command to conditionalize all the **debug** commands where the ANCP TCP connections are terminated.

Examples

The following is sample output from the **debugancpadjacency** command. The output fields are self-explanatory.

```
Router# debug ancp adjacency
*Sep 20 08:23:53.833:  Sending adj (SYN) msg nbr_info 0 nbr 10.1.1.1/46459 tcb 50C2050
*Sep 20 08:23:53.834:  ANCP: Using  Mac address aabb.cc00.7a00
*Sep 20 08:23:53.834:  ANCP : Sending adj (SYN) msg to writeQ, msg len 48 interface
Ethernet0/0.1 TCB 50C2050
*Sep 20 08:23:53.836:  ANCP: Received adjacency (SYN) message, from 10.1.1.1 port 46459 tcb
 50C2050
*Sep 20 08:23:53.836:  ANCP: 2 capability tlv(s) received
*Sep 20 08:23:53.837:  Capability received (mask) : 9
*Sep 20 08:23:53.837:  Sending adj (SYNACK) msg nbr_info 6368300 nbr 10.1.1.1/46459 tcb
50C2050
*Sep 20 08:23:53.837:  ANCP: Using  Mac address aabb.cc00.7a00
*Sep 20 08:23:53.837:  ANCP : Sending adj (SYNACK) msg to writeQ, msg len 48 interface
Ethernet0/0.1 TCB 50C2050
*Sep 20 08:23:53.837:  ANCP ADJ: received SYN from nbr, state SYNSENT Sending SYNACK to
nbr. State transitons to SYNRCVD
*Sep 20 08:23:53.841:  ANCP: Received adjacency (SYNACK) message, from 10.1.1.1 port 46459
 tcb 50C2050
*Sep 20 08:23:53.842:  ANCP: 2 capability tlv(s) received
*Sep 20 08:23:53.842:  Capability received (mask) : 9
*Sep 20 08:23:53.842:  Sending adj (ACK) msg nbr_info 6368300 nbr 10.1.1.1/46459 tcb 50C2050
*Sep 20 08:23:53.842:  ANCP: Using  Mac address aabb.cc00.7a00
```

The following is sample output from the **debugancpevents** command:

```
Router# debug ancp events
ANCP protocol events debugging is on
ANCP: TCP accepted on address 10.1.1.2 for remote peer 10.1.1.1
Sending adj msg, code 1 nbr_info 0 nbr 10.1.1.1/40224 tcb 549D930
TCP read_notify called, calling ancp_io_read_request. ANCP IO should service the read request
ANCP: Received adjacency message, code 1 Src 10.1.1.1 Src port 40224 tcb 549D930
ANCP: 2 capability tlv(s) received
Sending adj msg, code 2 nbr_info 5FA92F0 nbr 10.1.1.1/40224 tcb 549D930
ANCP ADJ: received SYN from nbr, state SYNSENT Sending SYNACK to nbr. State transitons to
SYNRCVD
TCP read_notify called, calling ancp_io_read_request. ANCP IO should service the read request
ANCP: Received adjacency message, code 2 Src 10.1.1.1 Src port 40224 tcb 549D930
ANCP: 2 capability tlv(s) received
Sending adj msg, code 3 nbr_info 5FA92F0 nbr 10.1.1.1/40224 tcb 549D930
ANCP ADJ: received SYNACK from nbr, state SYNRCVD Sending ACK to nbr. State transtions to
ESTAB
TCP read_notify called, calling ancp_io_read_request. ANCP IO should service the read request
ANCP: Received adjacency message, code 3 Src 10.1.1.1 Src port 40224 tcb 549D930
ANCP: 2 capability tlv(s) received
ANCP ADJ: received ACK from nbr, state ESTAB Sending ACK to nbr thou timer. State stays
ESTAB
TCP read_notify called, calling ancp_io_read_request. ANCP IO should service the read request
Sending adj msg, code 3 nbr_info 5FA92F0 nbr 10.1.1.1/40224 tcb 549D930
ANCP: Received adjacency message, code 3 Src 10.1.1.1 Src port 40224 tcb 549D930
ANCP: 2 capability tlv(s) received
ANCP ADJ: received ACK from nbr, state ESTAB Sending ACK to nbr thou timer. State stays
ESTAB
Sending adj msg, code 3 nbr_info 5FA92F0 nbr 10.1.1.1/40224 tcb 549D930
TCP read_notify called, calling ancp_io_read_request. ANCP IO should service the read request
```

```
ANCP: Received adjacency message, code 3 Src 10.1.1.1 Src port 40224 tcb 549D930
ANCP: 2 capability tlv(s) received
ANCP ADJ: received ACK from nbr, state ESTAB Sending ACK to nbr thou timer. State stays
ESTAB
```

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

Table 4: debug ancp events Field Descriptions

Field	Description
Received adjacency message	Adjacency message received from the ANCP neighbor. Adjacency protocol messages synchronize the Network Access Server and access nodes.
2 capability tlv(s) received	Supported capability type-length-values are included in the adjacency message.

The following is sample output from the **debugancpport-eventdetails**command. The output fields are self-explanatory.

```
Router# debug ancp port-event details
*Sep 20 08:24:55.608:  ANCP: Found db entry based on access-loop-circuit-id alcid1, received
 in Port UP message
*Sep 20 08:24:55.608:  Advance 12 bytes (aligned) to next TLV.
*Sep 20 08:24:55.609:  ANCP: Processing DSL_LINE_ATTRIBUTES sub-TLVs: PORT UP
*Sep 20 08:24:55.609:  ANCP: Port UP: received DSL Line Attrs, tlv length in msg 32 aligned
 len 32
*Sep 20 08:24:55.609:  DSL_TYPE received. 4
*Sep 20 08:24:55.609:  Advance 8 bytes (aligned) to next sub-TLV.
*Sep 20 08:24:55.609:  ACTUAL_DATA_RATE_UP received. 7878
*Sep 20 08:24:55.609:  Advance 8 bytes (aligned) to next sub-TLV.
*Sep 20 08:24:55.609:  ACTUAL_DATA_RATE_DN received. 9090
*Sep 20 08:24:55.610:  Advance 8 bytes (aligned) to next sub-TLV.
*Sep 20 08:24:55.610:  DSL_LINE_STATE received. 1
*Sep 20 08:24:55.610:  Advance 8 bytes (aligned) to next sub-TLV.
*Sep 20 08:24:55.610:  Advance 36 bytes (aligned) to next TLV.
```

The following is sample output from the **debugancpport-managementdetails**command. The output fields are self-explanatory.

```
Router# debug ancp port-management details
*Sep 20 08:29:22.263:
 3120 3500
 0000 0000
 8001 0034
 0000 0000
 0000 0000
 0000 0000
 0000 0900
 0000 0000
 0020 0500
 0001 0010
 0001 0006
 616C 6369
 6431 0000
*Sep 20 08:29:22.266:  ANCP_PORT_MGMT: Ethernet0/0.1 Port Mgmt message, Total TLVs : 1
*Sep 20 08:29:22.266:  ANCP_PORT_MGMT: Received Access-Loop-Cir-ID alcid1
*Sep 20 08:29:22.266:  ANCP_PORT_MGMT: Received Result: success(0x3) Code: Specified access
 line does not exist(0x500).
```

*Sep 20 08:29:22.266: Advance 12 bytes (aligned) to next TLV.

Related Commands

Command	Description
debug condition interface	Limits the output for some **debug** commands on the basis of interface, VC, or VLAN.
show ancp neighbor	Displays statistics of ANCP neighbor information and neighborship information with local ANCP ports.
show ancp status	Displays ANCP-related information for the ANCP endpoints configured on a BRAS interface.

debug appfw

To display debug messages about Cisco IOS Firewall events, use the **debugappfw**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug appfw {**application** *protocol*| **function-trace**| **object-creation**| **object-deletion**| **events**| **timers**| **detailed**}

no debug appfw {**application** *protocol*| **function-trace**| **object-creation**| **object-deletion**| **events**| **timers**| **detailed**}

Syntax Description

application *protocol*	Displays messages about protocol events of firewall-inspected applications, including details about the protocol's packets. Currently, the only supported protocol is HTTP. (Issue the **http** keyword.)
function-trace	Displays messages about software functions called by Cisco IOS Firewall.
object-creation	Displays messages about software objects that are being created by Cisco IOS Firewall. Cisco IOS Firewall-inspected sessions begin when the object is created.
object-deletion	Displays messages about software objects that are being deleted by Cisco IOS Firewall. Cisco IOS Firewall-inspected sessions close when the object is deleted.
events	Displays messages about Cisco IOS software events, including Cisco IOS Firewall packet processing.
timers	Displays messages about Cisco IOS Firewall timer events, such as an idle timeout by the Cisco IOS Firewall.
detailed	Detailed information for all other enabled Cisco IOS Firewall debugging is displayed. **Note** This keyword should be used in conjunction with other Cisco IOS Firewall debugging commands.

Command Modes

Privileged EXEC

Command History

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

Examples

The following sample configuration shows how to configure an HTTP policy with application firewall debugging enabled:

```
Router(config)# appfw policy-name myPolicyAPPFW  FUNC:appfw_policy_find
APPFW  FUNC:appfw_policy_find -- Policy myPolicy is not found
APPFW  FUNC:appfw_policy_alloc
APPFW  FUNC:appfw_policy_alloc -- policy_alloc 0x65727278
APPFW  FUNC:appfw_policy_alloc -- Policy 0x65727278 is set to valid
APPFW  FUNC:appfw_policy_alloc -- Policy myPolicy has been created
APPFW  FUNC:appfw_policy_command -- memlock policy 0x65727278

! Debugging sample for application (HTTP) creation

Router(cfg-appfw-policy)# application httpAPPFW  FUNC:appfw_http_command

APPFW  FUNC:appfw_http_appl_find
APPFW  FUNC:appfw_http_appl_find -- Application not found
APPFW  FUNC:appfw_http_appl_alloc
APPFW  FUNC:appfw_http_appl_alloc -- appl_http 0x64D7A25C
APPFW  FUNC:appfw_http_appl_alloc -- Application HTTP parser structure 64D7A25C created
! Debugging sample for HTTP-specific application inspection
Router(cfg-appfw-policy-http)#
Router(cfg-appfw-policy-http)# strict-http action reset alarm
APPFW  FUNC:appfw_http_subcommand
APPFW  FUNC:appfw_http_subcommand -- strict-http cmd turned on
Router# debug appfw detailed

APPFW Detailed Debug debugging is on
fw7-7206a#debug appfw object-creation
APPFW Object Creations debugging is on
fw7-7206a#debug appfw object-deletion
APPFW Object Deletions debugging is on
```

debug apple arp

To enable debugging of the AppleTalk Address Resolution Protocol (AARP), use the **debugapplearp** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug apple arp [*type number*]

no debug apple arp [*type number*]

Syntax Description

type	(Optional) Interface type.
number	(Optional) Interface number.

Command Modes

Privileged EXEC

Usage Guidelines

This command is helpful when you experience problems communicating with a node on the network you control (a neighbor). If the **debugapplearp** display indicates that the router is receiving AARP probes, you can assume that the problem does not reside at the physical layer.

Examples

The following is sample output from the **debugapplearp** command:

```
Router# debug apple arp
Ether0: AARP: Sent resolve for 4160.26
Ether0: AARP: Reply from 4160.26(0000.0c00.0453) for 4160.154(0000.0c00.8ea9)
Ether0: AARP: Resolved waiting request for 4160.26(0000.0c00.0453)
Ether0: AARP: Reply from 4160.19(0000.0c00.0082) for 4160.154(0000.0c00.8ea9)
Ether0: AARP: Resolved waiting request for 4160.19(0000.0c00.0082)
Ether0: AARP: Reply from 4160.19(0000.0c00.0082) for 4160.154(0000.0c00.8ea9)
```

Explanations for representative lines of output follow.

The following line indicates that the router has requested the hardware MAC address of the host at network address 4160.26:

```
Ether0: AARP: Sent resolve for 4160.26
```

The following line indicates that the host at network address 4160.26 has replied, giving its MAC address (0000.0c00.0453). For completeness, the message also shows the network address to which the reply was sent and its hardware MAC address (also in parentheses).

```
Ether0: AARP: Reply from 4160.26(0000.0c00.0453) for 4160.154(0000.0c00.8ea9)
```

The following line indicates that the MAC address request is complete:

```
Ether0: AARP: Resolved waiting request for 4160.26(0000.0c00.0453)
```

debug apple domain

To enable debugging of the AppleTalk domain activities, use the **debugappledomain**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug apple domain

no debug apple domain

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

Use the **debugappledomain** command to observe activity for domains and subdomains. Use this command in conjunction with the **debugappleremap** command to observe interaction between remapping and domain activity. Messages are displayed when the state of a domain changes, such as creating a new domain, deleting a domain, and updating a domain.

Examples

The following is sample output from the **debugappledomain** command intermixed with output from the **debugappleremap** command; the two commands show related events:

```
Router# debug apple domain
Router# debug apple remap
AT-REMAP: RemapProcess for net 30000 domain AURP Domain 1
AT-REMAP: ReshuffleRemapList for subdomain 1
AT-REMAP: Could not find a remap for cable 3000-3001
AT-DOMAIN: atdomain_DisablePort for Tunnel0
AT-DOMAIN: CleanUpDomain for domain 1 [AURP Domain 1]
AT-DOMAIN: Disabling interface Ethernet1
AT-DOMAIN: atdomain_DisablePort for Ethernet1
AT-DOMAIN: CleanUpDomain for domain 1 [AURP Domain 1]
AT-DOMAIN: CleanSubDomain for inbound subdomain 1
AT-REMAP: Remap for net 70 inbound subdomain 1 has been deleted
AT-DOMAIN: DeleteAvRemapList for inbound subdomain 1
AT-DOMAIN: DeleteRemapTable for subdomain 1
AT-DOMAIN: DeleteAvRemapList for inbound subdomain 1
AT-DOMAIN: CleanSubDomain for outbound subdomain 1
AT-DOMAIN: DeleteRemapTable for subdomain 1
AT-REMAP: RemapProcess for net 30000 domain AURP Domain 1 Remapped Net 10000
AT-REMAP: Remap for net 50 outbound subdomain 1 has been deleted
AT-DOMAIN: DeleteAvRemapList for outbound subdomain 1
AT-DOMAIN: DeleteAvRemapList for outbound subdomain 1
AT-DOMAIN: CleanUpDomain for domain 1 [AURP Domain 1]
AT-DOMAIN: CleanSubDomain for inbound subdomain 1
AT-DOMAIN: DeleteRemapTable for subdomain 1
AT-DOMAIN: DeleteAvRemapList for inbound subdomain 1
AT-DOMAIN: CleanSubDomain for outbound subdomain 1
AT-DOMAIN: DeleteRemapTable for subdomain 1
AT-DOMAIN: DeleteAvRemapList for outbound subdomain 1
```

Related Commands

Command	Description
debug apple remap	Enables debugging of the AppleTalk remap activities.

debug apple eigrp-all

To enable debugging output from the Enhanced IGRP routines, use the **debugappleeigrp-all**privileged EXEC command. The **no** form of this command disables debugging output.

debug apple eigrp-all

no debug apple eigrp-all

Syntax Description

This command has no arguments or keywords.

Command History

Release	Modification
10.0	This command was introduced.
12.2(13)T	This command is no longer supported in Cisco IOS Mainline relesaes or in Technology-based(T-train) releases. It might continue to appear in 12.2S-family releases.

Usage Guidelines

The **debugappleeigrp-all** command can be used to monitor acquisition of routes, aging route table entries, and advertisement of known routes through Enhanced IGRP.

Caution Because the **debugappleeigrp-all** command can generate many messages, use it only when the CPU utilization of the router is less than 50 percent.

Examples

The following is sample output from the **debugappleeigrp-all**command:

```
Router# debug apple eigrp-all
3:54:34: atigrp2_router: peer is 83.195
3:54:37: AT: atigrp2_write: about to send packet
3:54:37: Ethernet2: output AT packet: enctype UNKNOWN, size 65
3:54:37: 07FFFFFF0000FFFFFFFFFFFF00000C1485B00046|0041ACD100000053FF8F58585802059110
3:54:37: 000000000000000000000000000000010001000C010001000000000F0204000C0053005300
3:54:37: AT: atigrp2, src=Ethernet2:83.143, dst=83-83, size=52, EIGRP pkt sent
3:54:39: atigrp2_router: peer is 83.195
3:54:42: AT: atigrp2_write: about to send packet
3:54:42: Ethernet2: output AT packet: enctype UNKNOWN, size 65
3:54:42: 07FFFFFF0000FFFFFFFFFFFF00000C1485B00046|0041ACD100000053FF8F58585802059110
3:54:42: 000000000000000000000000000000010001000C010001000000000F0204000C0053005300
3:54:42: AT: atigrp2, src=Ethernet2:83.143, dst=83-83, size=52, EIGRP pkt sent
```

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

Table 5: debug apple eigrp Field Descriptions

Field	Description
atigrp2_router:	AppleTalk address of the neighbor.

Field	Description
AT:	Indicates that this is an AppleTalk packet.
Ethernet2:	Name of the interface through which the router received the packet.
src=	Name of the interface sending the Enhanced IGRP packet, as well at its AppleTalk address.
dst=	Cable range of the destination of the packet.
size=	Size of the packet (in bytes).

debug apple errors

To display errors occurring in the AppleTalk network, use the **debugappleerrors** command in privileged EXEC mode. To disable debugging output, use the no form of this command.

debug apple errors [*type number*]

no debug apple errors [*type number*]

Syntax Description

type	(Optional) Interface type.
number	(Optional) Interface number.

Command Modes

Privileged EXEC

Usage Guidelines

In a stable AppleTalk network, the **debugappleerrors** command produces little output.

To solve encapsulation problems, enable **debugappleerrors** and **debugapplepacket** together.

Examples

The following is sample output from the **debugappleerrors** command when a router is brought up with a zone that does not agree with the zone list of other routers on the network:

```
Router# debug apple errors
%AT-3-ZONEDISAGREES: Ethernet0: AppleTalk port disabled; zone list incompatible with 4160.19
%AT-3-ZONEDISAGREES: Ethernet0: AppleTalk port disabled; zone list incompatible with 4160.19
%AT-3-ZONEDISAGREES: Ethernet0: AppleTalk port disabled; zone list incompatible with 4160.19
```

As the output suggests, a single error message indicates zone list incompatibility; this message is sent out periodically until the condition is corrected or the **debugappleerrors** command is turned off.

Most of the other messages that the **debugappleerrors** command can generate are obscure or indicate a serious problem with the AppleTalk network. Some of these other messages follow.

In the following message, RTMPRsp, RTMPReq, ATP, AEP, ZIP, ADSP, or SNMP could replace NBP, and "llap dest not for us" could replace "wrong encapsulation":

```
Packet discarded, src 4160.12-254,dst 4160.19-254,NBP,wrong encapsulation
```

In the following message, in addition to an invalid echo packet error, other possible errors are unsolicited AEP echo reply, unknown echo function, invalid ping packet, unknown ping function, and bad responder packet type:

```
Ethernet0: AppleTalk packet error; no source address available
AT: pak_reply: dubious reply creation, dst 4160.19
AT: Unable to get a buffer for reply to 4160.19
Processing error, src 4160.12-254,dst 4160.19-254,AEP, invalid echo packet
```

The **debugappleerrors** command can print out additional messages when other debugging commands are also turned on. When you turn on both the **debugappleerrors** and **debugappleevents**commands, the following message can be generated:

```
Proc err, src 4160.12-254,dst 4160.19-254,ZIP,NetInfo Reply format is invalid
```

In the preceding message, in addition to the NetInfo Reply format is invalid error, other possible errors are NetInfoReply not for me, NetInfoReply ignored, NetInfoReply for operational net ignored, NetInfoReply from invalid port, unexpected NetInfoReply ignored, cannot establish primary zone, no primary has been set up, primary zone invalid, net information mismatch, multicast mismatch, and zones disagree.

When you turn on both the **debugappleerrors** and **debugapplenbp**commands, the following message can be generated:

```
Processing error,...,NBP,NBP name invalid
```

In the preceding message, in addition to the NBP name invalid error, other possible errors are NBP type invalid, NBP zone invalid, not operational, error handling brrq, error handling proxy, NBP fwdreq unexpected, No route to srcnet, Proxy to "*" zone, Zone "*" from extended net, No zone info for "*", and NBP zone unknown.

When you turn on both the **debugappleerrors** and **debugapplerouting**commands, the following message can be generated:

```
Processing error,...,RTMPReq, unknown RTMP request
```

In the preceding message, in addition to an unknown RTMP request error, other possible errors are RTMP packet header bad, RTMP cable mismatch, routed RTMP data, RTMP bad tuple, and Not Req or Rsp.

debug apple events

To display information about AppleTalk special events, neighbors becoming reachable or unreachable, and interfaces going up or down, use the **debugappleevents** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug apple events [*type number*]

no debug apple events [*type number*]

Syntax Description

type	(Optional) Interface type.
number	(Optional) Interface number.

Command Modes

Privileged EXEC

Usage Guidelines

Only significant events (for example, neighbor and route changes) are logged.

The **debugappleevents** command is useful for solving AppleTalk network problems because it provides an overall picture of the stability of the network. In a stable network, the **debugappleevents** command does not return any information. If the command generates numerous messages, those messages can indicate possible sources of the problems.

When configuring or making changes to a router or interface for AppleTalk, enable the **debugappleevents** command to alert you to the progress of the changes or to any errors that might result. Also use this command periodically when you suspect network problems.

The **debugappleevents** command is also useful to determine whether network flapping (nodes toggling online and offline) is occurring. If flapping is excessive, look for routers that only support 254 networks.

When you enable the **debugappleevents**command, you will see any messages that the **appleevent-logging**configuration command normally displays. Turning on the **debugappleevents**command, however, does not cause the **appleevent-logging**command to be maintained in nonvolatile memory. Only turning on the **appleevent-logging** command explicitly stores it in nonvolatile memory. Furthermore, if the **appleevent-logging** command is already enabled, turning on or off the **debugappleevents** command does not affect the **appleevent-logging** command.

Examples

The **debugappleevents** command is useful in tracking the discovery mode state changes through which an interface progresses. When no problems are encountered, the state changes progress as follows:

1. Line down.
2. Restarting.
3. Probing (for its own address [node ID] using AARP).
4. Acquiring (sending out GetNetInfo requests).
5. Requesting zones (the list of zones for its cable).

6. Verifying (that the router's configuration is correct. If not, a port configuration mismatch is declared).
7. Checking zones (to make sure its list of zones is correct).
8. Operational (participating in routing).

Explanations for individual lines of output follow.

The following message indicates that a port is set. In this case, the zone multicast address is being reset.

```
Ether0: AT: Resetting interface address filters
```

The following messages indicate that the router is changing to restarting mode:

```
%AT-5-INTRESTART: Ether0: AppleTalk port restarting; protocol restarted
Ether0: AppleTalk state changed; unknown -> restarting
```

The following message indicates that the router is probing in the startup range of network numbers (65280 to 65534) to discover its network number:

```
Ether0: AppleTalk state changed; restarting -> probing
```

The following message indicates that the router is enabled as a nonrouting node using a provisional network number within its startup range of network numbers. This type of message only appears if the network address the router will use differs from its configured address. This is always the case for a discovery-enabled router; it is rarely the case for a nondiscovery-enabled router.

```
%AT-6-ADDRUSED: Ether0: AppleTalk node up; using address 65401.148
```

The following messages indicate that the router is sending out GetNetInfo requests to discover the default zone name and the actual network number range in which its network number can be chosen:

```
Ether0: AppleTalk state changed; probing -> acquiring
%AT-6-ACQUIREMODE: Ether0: AT port initializing; acquiring net configuration
```

Now that the router has acquired the cable configuration information, the following message indicates that it restarts using that information:

```
Ether0: AppleTalk state changed; acquiring -> restarting
```

The following messages indicate that the router is probing for its actual network address:

```
Ether0: AppleTalk state changed; restarting -> line down
Ether0: AppleTalk state changed; line down -> restarting
Ether0: AppleTalk state changed; restarting -> probing
```

The following message indicates that the router has found an actual network address to use:

```
%AT-6-ADDRUSED: Ether0: AppleTalk node up; using address 4160.148
```

The following messages indicate that the router is sending out GetNetInfo requests to verify the default zone name and the actual network number range from which its network number can be chosen:

```
Ether0: AppleTalk state changed; probing -> acquiring
%AT-6-ACQUIREMODE: Ether0: AT port initializing; acquiring net configuration
```

The following message indicates that the router is requesting the list of zones for its cable:

```
Ether0: AppleTalk state changed; acquiring -> requesting zones
```

The following messages indicate that the router is sending out GetNetInfo requests to make sure its understanding of the configuration is correct:

```
Ether0: AppleTalk state changed; requesting zones -> verifying
AT: Sent GetNetInfo request broadcast on Ethernet0
```

The following message indicates that the router is rechecking its list of zones for its cable:

```
Ether0: AppleTalk state changed; verifying -> checking zones
```

The following message indicates that the router is now fully operational as a routing node and can begin routing:

```
Ether0: AppleTalk state changed; checking zones -> operational
```

A nondiscovery-enabled router can come up when no other router is on the wire; however, it must assume that its configuration (if accurate syntactically) is correct, because no other router can verify it. Notice that the last line indicates this situation.

The following is sample output from the **debugappleevents** command that describes a discovery-enabled router coming up when there is no seed router on the wire:

```
Router# debug apple events
Ether0: AT: Resetting interface address filters
%AT-5-INTRESTART: Ether0: AppleTalk port restarting; protocol restarted
Ether0: AppleTalk state changed; unknown -> restarting
Ether0: AppleTalk state changed; restarting -> probing
%AT-6-ADDRUSED: Ether0: AppleTalk node up; using address 65401.148
Ether0: AppleTalk state changed; probing -> acquiring
AT: Sent GetNetInfo request broadcast on Ether0
AT: Sent GetNetInfo request broadcast on Ether0
AT: Sent GetNetInfo request broadcast on Ether0
AT: Sent GetNetInfo request broadcast on Ether0
AT: Sent GetNetInfo request broadcast on Ether0
```

When you attempt to bring up a nonseed router without a seed router on the wire, it never becomes operational; instead, it hangs in the acquiring mode and continues to send out periodic GetNetInfo requests.

When a nondiscovery-enabled router is brought up on an AppleTalk internetwork that is in compatibility mode (set up to accommodate extended as well as nonextended AppleTalk) and the router has violated internetwork compatibility:

The following three configuration command lines indicate the part of the configuration of the router that caused the configuration mismatch:

```
lestat(config)# interface ethernet 0
lestat(config-if)# apple cab 41-41
lestat(config-if)# apple zone Marketing
```

The router shown had been configured with a cable range of 41-41 instead of 40-40, which would have been accurate. Additionally, the zone name was configured incorrectly; it should have been "Marketing," rather than being misspelled as "Markting."

debug apple nbp

To display debugging output from the Name Binding Protocol (NBP) routines, use the **debugapplenbp** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug apple nbp [*type number*]

no debug apple nbp [*type number*]

Syntax Description

type	(Optional) Interface type.
number	(Optional) Interface number.

Command Modes

Privileged EXEC

Usage Guidelines

To determine whether the router is receiving NBP lookups from a node on the AppleTalk network, enable **debugapplenbp** at each node between the router and the node in question to determine where the problem lies.

Caution Because the **debugapplenbp** command can generate many messages, use it only when the CPU utilization of the router is less than 50 percent.

Examples

The following is sample output from the **debugapplenbp** command:

```
Router# debug apple nbp
AT: NBP ctrl = LkUp, ntuples = 1, id = 77
AT: 4160.19, skt 2, enum 0, name: =:ciscoRouter@Low End SW Lab
AT: LkUp =:ciscoRouter@Low End SW Lab
AT: NBP ctrl = LkUp-Reply, ntuples = 1, id = 77
AT: 4160.154, skt 254, enum 1, name: lestat.Ether0:ciscoRouter@Low End SW Lab
AT: NBP ctrl = LkUp, ntuples = 1, id = 78
AT: 4160.19, skt 2, enum 0, name: =:IPADDRESS@Low End SW Lab
AT: NBP ctrl = LkUp, ntuples = 1, id = 79
AT: 4160.19, skt 2, enum 0, name: =:IPGATEWAY@Low End SW Lab
AT: NBP ctrl = LkUp, ntuples = 1, id = 83
AT: 4160.19, skt 2, enum 0, name: =:ciscoRouter@Low End SW Lab
AT: LkUp =:ciscoRouter@Low End SW Lab
AT: NBP ctrl = LkUp, ntuples = 1, id = 84
AT: 4160.19, skt 2, enum 0, name: =:IPADDRESS@Low End SW Lab
AT: NBP ctrl = LkUp, ntuples = 1, id = 85
AT: 4160.19, skt 2, enum 0, name: =:IPGATEWAY@Low End SW Lab
AT: NBP ctrl = LkUp, ntuples = 1, id = 85
AT: 4160.19, skt 2, enum 0, name: =:IPGATEWAY@Low End SW Lab
```

The first three lines describe an NBP lookup request:

```
AT: NBP ctrl = LkUp, ntuples = 1, id = 77
AT: 4160.19, skt 2, enum 0, name: =:ciscoRouter@Low End SW Lab
AT: LkUp =:ciscoRouter@Low End SW Lab
```

The table below describes the fields in the first line of output.

Table 6: debug apple nbp Field Descriptions--First Line of Output

Field	Description
AT: NBP	Indicates that this message describes an AppleTalk NBP packet.
ctrl = LkUp	Identifies the type of NBP packet. Possible values are as follows: • LkUp--NBP lookup request. • LkUp-Reply--NBP lookup reply.
ntuples = 1	Indicates the number of name-address pairs in the lookup request packet. Range: 1 to 31 tuples.
id = 77	Identifies an NBP lookup request value.

The table below describes the fields in the second line of output.

Table 7: debug apple nbp Field Descriptions--Second Line of Output

Field	Description
AT:	Indicates that this message describes an AppleTalk packet.
4160.19	Indicates the network address of the requester.
skt 2	Indicates the internet socket address of the requester. The responder will send the NBP lookup reply to this socket address.
enum 0	Indicates the enumerator field. Used to identify multiple names registered on a single socket. Each tuple is assigned its own enumerator, incrementing from 0 for the first tuple.

Field	Description
name: =:ciscoRouter@Low End SW Lab	Indicates the entity name for which a network address has been requested. The AppleTalk entity name includes three components: • Object (in this case, a wildcard character [=], indicating that the requester is requesting name-address pairs for all objects of the specified type in the specified zone). • Type (in this case, ciscoRouter). • Zone (in this case, Low End SW Lab).

The third line in the output essentially reiterates the information in the two lines above it, indicating that a lookup request has been made regarding name-address pairs for all objects of the ciscoRouter type in the Low End SW Lab zone.

Because the router is defined as an object of type ciscoRouter in zone Low End SW Lab, the router sends an NBP lookup reply in response to this NBP lookup request. The following two lines of output show the response of the router:

```
AT: NBP ctrl = LkUp-Reply, ntuples = 1, id = 77
AT: 4160.154, skt 254, enum 1, name: lestat.Ether0:ciscoRouter@Low End SW Lab
```

In the first line, ctrl = LkUp-Reply identifies this NBP packet as an NBP lookup request. The same value in the id field (id = 77) associates this lookup reply with the previous lookup request. The second line indicates that the network address associated with the entity name of the router (lestat.Ether0:ciscoRouter@Low End SW Lab) is 4160.154. The fact that no other entity name/network address is listed indicates that the responder only knows about itself as an object of type ciscoRouter in zone Low End SW Lab.

debug apple packet

To display per-packet debugging output, use the **debugapplepacket** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug apple packet [*type number*]

no debug apple packet [*type number*]

Syntax Description

type	(Optional) Interface type.
number	(Optional) Interface number.

Command Modes

Privileged EXEC

Usage Guidelines

With this command, you can monitor the types of packets being slow switched. It displays at least one line of debugging output per AppleTalk packet processed.

The output reports information online when a packet is received or a transmission is attempted.

When invoked in conjunction with the **debugapplerouting**, **debugapplezip**, and **debugapplenbp** commands, the**debugapplepacket** command adds protocol processing information in addition to generic packet details. It also reports successful completion or failure information.

When invoked in conjunction with the **debugappleerrors** command, the **debugapplepacket** command reports packet-level problems, such as those concerning encapsulation.

Caution

Because the **debugapplepacket** command can generate many messages, use it only when the CPU utilization of the router is less than 50 percent.

Examples

The following is sample output from the **debugapplepacket** command:

```
Router# debug apple packet
Ether0: AppleTalk packet: enctype SNAP, size 60, encaps000000000000000000000000
AT: src=Ethernet0:4160.47, dst=4160-4160, size=10, 2 rtes, RTMP pkt sent
AT: ZIP Extended reply rcvd from 4160.19
AT: ZIP Extended reply rcvd from 4160.19
AT: src=Ethernet0:4160.47, dst=4160-4160, size=10, 2 rtes, RTMP pkt sent
Ether0: AppleTalk packet: enctype SNAP, size 60, encaps000000000000000000000000
Ether0: AppleTalk packet: enctype SNAP, size 60, encaps000000000000000000000000
```

The table below describes the fields in the first line of output.

Table 8: debug apple packet Field Descriptions--First Line of Output

Field	Description
Ether0:	Name of the interface through which the router received the packet.
AppleTalk packet	Indicates that this is an AppleTalk packet.
enctype SNAP	Encapsulation type for the packet.
size 60	Size of the packet (in bytes).
encaps000000000000000000000000	Encapsulation.

The table below describes the fields in the second line of output.

Table 9: debug apple packet Field Descriptions--Second Line of Output

Field	Description
AT:	Indicates that this is an AppleTalk packet.
src=Ethernet0:4160.47	Name of the interface sending the packet and its AppleTalk address.
dst=4160-4160	Cable range of the destination of the packet.
size=10	Size of the packet (in bytes.)
2 rtes	Indicates that two routes in the routing table link these two addresses.
RTMP pkt sent	Type of packet sent.

The third line indicates the type of packet received and its source AppleTalk address. This message is repeated in the fourth line because AppleTalk hosts can send multiple replies to a given GetNetInfo request.

debug apple remap

To enable debugging of the AppleTalk remap activities, use the **debugappleremap**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug apple remap

no debug apple remap

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

Use the **debugappleremap**command with the **debugappledomain** command to observe activity between domains and subdomains. Messages from the **debugappleremap** command are displayed when a particular remapping function occurs, such as creating remaps or deleting remaps.

Examples

The following is sample output from the **debugappleremap** command intermixed with output from the **debugappledomain**command; the two commands show related events.

```
Router# debug apple remap
Router# debug apple domain
AT-REMAP: RemapProcess for net 30000 domain AURP Domain 1
AT-REMAP: ReshuffleRemapList for subdomain 1
AT-REMAP: Could not find a remap for cable 3000-3001
AT-DOMAIN: atdomain_DisablePort for Tunnel0
AT-DOMAIN: CleanUpDomain for domain 1 [AURP Domain 1]
AT-DOMAIN: Disabling interface Ethernet1
AT-DOMAIN: atdomain_DisablePort for Ethernet1
AT-DOMAIN: CleanUpDomain for domain 1 [AURP Domain 1]
AT-DOMAIN: CleanSubDomain for inbound subdomain 1
AT-REMAP: Remap for net 70 inbound subdomain 1 has been deleted
AT-DOMAIN: DeleteAvRemapList for inbound subdomain 1
AT-DOMAIN: DeleteRemapTable for subdomain 1
AT-DOMAIN: DeleteAvRemapList for inbound subdomain 1
AT-DOMAIN: CleanSubDomain for outbound subdomain 1
AT-DOMAIN: DeleteRemapTable for subdomain 1
AT-REMAP: RemapProcess for net 30000 domain AURP Domain 1 Remaped Net 10000
AT-REMAP: Remap for net 50 outbound subdomain 1 has been deleted
AT-DOMAIN: DeleteAvRemapList for outbound subdomain 1
AT-DOMAIN: DeleteAvRemapList for outbound subdomain 1
AT-DOMAIN: CleanUpDomain for domain 1 [AURP Domain 1]
AT-DOMAIN: CleanSubDomain for inbound subdomain 1
AT-DOMAIN: DeleteRemapTable for subdomain 1
AT-DOMAIN: DeleteAvRemapList for inbound subdomain 1
AT-DOMAIN: CleanSubDomain for outbound subdomain 1
AT-DOMAIN: DeleteRemapTable for subdomain 1
AT-DOMAIN: DeleteAvRemapList for outbound subdomain 1
```

Related Commands

Command	Description
debug apple domain	Enables debugging of the AppleTalk domain activities.

debug apple routing

To enable debugging output from the Routing Table Maintenance Protocol (RTMP) routines, use the **debugapplerouting** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug apple routing [*type number*]

no debug apple routing [*type number*]

Syntax Description

type	(Optional) Interface type.
number	(Optional) Interface number.

Command Modes

Privileged EXEC

Usage Guidelines

This command can be used to monitor acquisition of routes, aging of routing table entries, and advertisement of known routes. It also reports conflicting network numbers on the same network if the network is misconfigured.

Caution Because the **debugapplerouting** command can generate many messages, use it only when router CPU utilization is less than 50 percent.

Examples

The following is sample output from the **debugapplerouting** command:

```
Router# debug apple routing
AT: src=Ethernet0:4160.41, dst=4160-4160, size=19, 2 rtes, RTMP pkt sent
AT: src=Ethernet1:41069.25, dst=41069, size=427, 96 rtes, RTMP pkt sent
AT: src=Ethernet2:4161.23, dst=4161-4161, size=427, 96 rtes, RTMP pkt sent
AT: Route ager starting (97 routes)
AT: Route ager finished (97 routes)
AT: RTMP from 4160.19 (new 0,old 94,bad 0,ign 0, dwn 0)
AT: RTMP from 4160.250 (new 0,old 0,bad 0,ign 2, dwn 0)
AT: RTMP from 4161.236 (new 0,old 94,bad 0,ign 1, dwn 0)
AT: src=Ethernet0:4160.41, dst=4160-4160, size=19, 2 rtes, RTMP pkt sent
```

The table below describes the fields in the first line of sample **debugapplerouting** output.

Table 10: debug apple routing Field Descriptions--First Line of Output

Field	Description
AT:	Indicates that this is AppleTalk debugging output.
src=Ethernet0:4160.41	Indicates the source router interface and network address for the RTMP update packet.

Field	Description
dst=4160-4160	Indicates the destination network address for the RTMP update packet.
size=19	Displays the size of this RTMP packet (in bytes).
2 rtes	Indicates that this RTMP update packet includes information on two routes.
RTMP pkt sent	Indicates that this type of message describes an RTMP update packet that the router has sent (rather than one that it has received).

The following two messages indicate that the ager has started and finished the aging process for the routing table and that this table contains 97 entries:

```
AT: Route ager starting (97 routes)
AT: Route ager finished (97 routes)
```

The table below describes the fields in the following line of the **debugapplerouting** command output:

```
AT: RTMP from 4160.19 (new 0,old 94,bad 0,ign 0, dwn 0)
```

Table 11: debug apple routing Field Descriptions

Field	Description
AT:	Indicates that this is AppleTalk debugging output.
RTMP from 4160.19	Indicates the source address of the RTMP update the router received.
new 0	Displays the number of routes in this RTMP update packet that the router did not already know about.
old 94	Displays the number of routes in this RTMP update packet that the router already knew about.
bad 0	Displays the number of routes the other router indicates have gone bad.
ign 0	Displays the number of routes the other router ignores.
dwn 0	Displays the number of poisoned tuples included in this packet.

debug apple zip

To display debugging output from the Zone Information Protocol (ZIP) routines, use the **debugapplezip** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug apple zip [*type number*]

no debug apple zip [*type number*]

Syntax Description

type	(Optional) Interface type.
number	(Optional) Interface number.

Command Modes

Privileged EXEC

Usage Guidelines

This command reports significant events such as the discovery of new zones and zone list queries. It generates information similar to that generated by the debug apple routing command, but generates it for ZIP packets instead of Routing Table Maintenance Protocol (RTMP) packets.

You can use the **debugapplezip** command to determine whether a ZIP storm is taking place in the AppleTalk network. You can detect the existence of a ZIP storm when you see that no router on a cable has the zone name corresponding to a network number that all the routers have in their routing tables.

Examples

The following is sample output from the **debugapplezip** command:

```
Router# debug apple zip
AT: Sent GetNetInfo request broadcast on Ether0
AT: Recvd ZIP cmd 6 from 4160.19-6
AT: 3 query packets sent to neighbor 4160.19
AT: 1 zones for 31902, ZIP XReply, src 4160.19
AT: net 31902, zonelen 10, name US-Florida
```

The first line indicates that the router has received an RTMP update that includes a new network number and is now requesting zone information:

```
AT: Sent GetNetInfo request broadcast on Ether0
```

The second line indicates that the neighbor at address 4160.19 replies to the zone request with a default zone:

```
AT: Recvd ZIP cmd 6 from 4160.19-6
```

The third line indicates that the router responds with three queries to the neighbor at network address 4160.19 for other zones on the network:

```
AT: 3 query packets sent to neighbor 4160.19
```

The fourth line indicates that the neighbor at network address 4160.19 responds with a ZIP extended reply, indicating that one zone has been assigned to network 31902:

```
AT: 1 zones for 31902, ZIP XReply, src 4160.19
```

The fifth line indicates that the router responds that the zone name of network 31902 is US-Florida, and the zone length of that zone name is 10:

```
AT: net 31902, zonelen 10, name US-Florida
```

debug appn all

To turn on all possible debugging messages for Advanced Peer-to-Peer Networking (APPN), use the **debugappnall** command in privileged EXEC mode. To disable debugging output, use the **no**form of this command.

debug appn all

no debug appn all

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

This command shows all APPN events. Use other forms of the **debugappn** command to display specific types of events.

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

Caution Debugging output takes priority over other network traffic. The **debugappnall** command generates more output than any other **debugappn** command and can alter timing in the network node. This command can severely diminish router performance or even render it unusable. In virtually all cases, it is best to use specific **debugappn** commands.

Examples

Refer to the documentation for specific **debugappn** commands for examples and explanations.

Related Comma

Note Refer to the other forms of the **debugappn** command to enable specific debug output selectively.

Command	Description
debug appn cs	Displays the APPN CS component activity.
debug appn ds	Displays debugging information on APPN DS component activity.
debug appn hpr	Displays information related to HPR code execution.

Command	Description
debug appn ms	Displays debugging information on APPN MS component activity.
debug appn nof	Displays information on APPN NOF component activity.
debug appn pc	Displays debugging information on APPN PC component activity.
debug appn ps	Displays debugging information on APPN PS component activity.
debug appn scm	Displays debugging information on APPN SCM component activity.
debug appn ss	Displays SS events.
debug appn ss	Displays debugging information on APPN TRS component activity.

debug appn cs

To display Advanced Peer-to-Peer Networking (APPN) Configuration Services (CS) component activity, use the **debugappncs** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug appn cs

no debug appn cs

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

The CS component is responsible for defining link stations, ports, and connection networks. It is responsible for the activation and deactivation of ports and link stations and handles status queries for these resources.

Examples

The following is sample output from the **debugappncs** command. In this example a link station is being stopped.

```
Router# debug appn cs
Turned on event 008000FF
Router# appn stop link PATTY
APPN: ----- CS ----- Deq STOP_LS message
APPN: ----- CS ----- FSM LS: 75 17 5 8
APPN: ----- CS ----- Sending DEACTIVATE_AS - station PATTY
APPN: ----- CS ----- deactivate_as_p->ips_header.lpid = A80A60
APPN: ----- CS ----- deactivate_as_p->ips_header.lpid = A80A60
APPN: ----- CS ----- Sending DESTROY_TG to PC - station PATTY - lpid=A80A60
APPN: ----- CS ----- Deq DESTROY_TG - station PATTY
APPN: ----- CS ----- FSM LS: 22 27 8 0
APPN: ----- CS ----- Sending TG update for LS PATTY to TRS
APPN: ----- CS ----- ENTERING XID_PROCESSING: 4
%APPN-6-APPNSENDMSG: Link Station PATTY stopped
```

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

Table 12: debug appn cs Field Descriptions

Field	Description
APPN	APPN debugging output.
CS	CS component output.
Deq	CS received a message from another component.
FSM LS	Link station finite state machine is being referenced.
Sending	CS is sending a message to another component.

Related Commands

Command	Description
debug appn all	Turns on all possible debugging messages for APPN.

debug appn ds

To display debugging information on Advanced Peer-to-Peer Networking (APPN) Directory Services (DS) component activity, use the **debugappnds** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug appn ds

no debug appn ds

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

The DS component manages searches for resources in the APPN network. DS is also responsible for registration of resources within the network.

Examples

The following is sample output from the **debugappnds** command. In this example a search has been received.

```
Router# debug appn ds
Turned on event 080000FF
APPN: NEWDS: LS: search from: NETA.PATTY
APPN: NEWDS: pcid: DD3321E8B5667111
APPN: NEWDS: Invoking FSM NNSolu
APPN: NEWDS: LSfsm_NNSolu: 00A67AA0 pcid: DD3321E8B5667111 row: 0 col: 0 inp: 80200000
APPN: NEWDS: LSfsm_parent: 00A89940 row: 0 col: 0 inp: 80000000
APPN: NEWDS: Rcvd a LMRQ
APPN: NEWDS: LSfsm_NNSolu: 00A67AA0 pcid: DD3321E8B5667111 row: 12 col: 1 inp: 40000000
APPN: NEWDS: LSfsm_parent: 00A89940 row: 8 col: 1 inp: 40000000
APPN: NEWDS: LSfsm_child: 00A89BE8 row: 0 col: 0 inp: 80000080
APPN: NEWDS: PQenq REQUEST_ROUTE(RQ) to TRS
APPN: NEWDS: LSfsm_child: 00A8A1C0 row: 1 col: 0 inp: 80000008
APPN: NEWDS: LSfsm_NNSolu: 00A67AA0 pcid: DD3321E8B5667111 row: 5 col: 1 inp: 80C04000
APPN: NEWDS: LSfsm_child: 00A8A1C0 row: 7 col: 1 inp: 80844008
APPN: NEWDS: Rcvd a LMRY
APPN: NEWDS: LSfsm_NNSolu: 00A67AA0 pcid: DD3321E8B5667111 row: 16 col: 6 inp: 40800000
APPN: NEWDS: LSfsm_child: 00A8A1C0 row: 14 col: 5 inp: 40800000
APPN: NEWDS: LSfsm_parent: 00A89940 row: 3 col: 1 inp: 80840000
APPN: NEWDS: send locate to node: NETA.PATTY
```

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

Table 13: debug appn ds Field Descriptions

Field	Description
APPN	APPN debugging output.
NEWDS	DS component output.
search from	Locate was received from NETA.PATTY.
LSfsm_	Locate Search finite state machine is being referenced.

Field	Description
PQenq	Message was sent to another component.
Rcvd	Message was received from another component.
send locate	Locate will be sent to NETA.PATTY.

Related Commands

Command	Description
debug appn all	Turns on all possible debugging messages for APPN.

debug appn hpr

To display debugging information related to High Performance Routing (HPR) code execution, use the **debugappnhpr**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug appn hpr

no debug appn hpr

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Examples

The following is sample output from the **debugappnhpr** command:

```
Router# debug appn hpr
APPN: -- ncl.ncl_map_dlc_type() -- mapping TOKEN_RING(4) to NCL_TR(3)
APPN: -- ncl.ncl_port() -- called with port_type:3, cisco_idb:893A14, hpr_ssap:C8
APPN: -- ncl.process_port_change() -- port coming up
APPN: -- ncl.process_port_change() -- PORT_UP
APPN: -- ncl.ncl_port_fsm -- FSM Invoked: Input:0, State:0->1, Action:0
APPN: -- ncl.ncl_port_fsm -- FSM Invoked: Input:1, State:1->2, Action:1
APPN: -- ncl.ncl_unmap_dlc_type() -- mapping NCL(3) to CLS(3)
APPN: ----- ANR  ----- Sending ACTIVATE_SAP.req
APPN: -- cswncsnd.main() -- received LSA_IPS ips.
APPN: -- ncl.ncl_port_fsm -- FSM Invoked: Input:3, State:2->3, Action:4
APPN: -- ncl.ncl_assign_anr() -- Assigned ANR,anr:8002
APPN: -- ncl.ncl_map_dlc_type() -- mapping TOKEN_RING(4) to NCL_TR(3)
APPN: -- ncl.ncl_populate_anr() -- anr:8002, dlc_type:3, idb 893A14
APPN: -- ncl.ncl_populate_anr() -- send anr_tbl_update to owning cswncsnd
APPN: -- ncl.ncl_ls_fsm -- FSM Invoked: Input:0, State:0->1, Action:0
APPN: ncl.ncl_send_reqopn_stn_req
APPN: -- ncl.ncl_unmap_dlc_type() -- mapping NCL(3) to CLS(3)
APPN: -- ncl.ncl_ls_fsm() -- send anr_tbl_update to owning cswncsnd
APPN: -- cswncsnd.main() -- received ANR_TBL_UPDATE ips.
APPN: -- cswncsnd.apply_anr_table_update() -- ANR:8002
APPN: -- cswncsnd.main() -- received ANR_TBL_UPDATE ips.
APPN: -- cswncsnd.apply_anr_table_update() -- ANR:8002
APPN: -- cswncsnd.main() -- received LSA_IPS ips.
APPN: -- ncl.ncl_ls_fsm -- FSM Invoked: Input:1, State:1->2, Action:1
APPN: -- ncl.ncl_ls_fsm -- P_CEP_ID:AAF638
APPN: -- ncl.ncl_ls_fsm() -- send anr_tbl_update to owning cswncsnd
APPN: -- cswncsnd.main() -- received ANR_TBL_UPDATE ips.
APPN: -- cswncsnd.apply_anr_table_update() -- ANR:8002
APPN: rtpm:  rtp_send() sent data over connection B9D5E8
APPN: hpr timer: rtt start time clocked at 135952 ms
APPN: -- cswncsnd.main() -- received NCL_SND_MSG ips.
APPN: -- cswncsnd.process_nlp_from_rtp() -- label: 8002, send to p_cep 00AAF638.
APPN: hpr timer: rtt end time clocked at 135972 ms
APPN: hpr timer: round trip time measured at 20 ms
```

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

Table 14: debug appn hpr Field Descriptions

Field	Description
APPN	APPN debugging output.

Field	Description
NCL	Network control layer debugging output. Network control layer is the component that handles ANR packets.
ncl_port_fsm	Network control layer port finite state machine has been invoked.
ncl_assign_anr	ANR label has been assigned to an activating link station.
ncl_populate_anr	System is updating the ANR record with information specific to the link station.
ncl_ls_fsm	Network control layer link finite state machine has been invoked.
rtp_send	RTP is about to send a packet.
hpr timer	Debugging output related to an HPR timer.
rtt start time	RTP is measuring the round-trip time for an HPR status request packet. This is the start time.
NCL_SND_MSG	Network control layer has been requested to send a packet.
process_nlp_from_rtp	Network control layer has been requested by RTP to send a packet.
rtt end time	RTP is measuring the round-trip time for an HPR status request packet. This is the time.
round trip time	Round-trip time for this HPR status exchange has been computed.

Related Commands

Command	Description
debug appn all	Turns on all possible debugging messages for APPN.

debug appn ms

To display debugging information on Advanced Peer-to-Peer Networking (APPN) Management Services (MS) component activity, use the **debugappnms** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug appn ms

no debug appn ms

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

The MS component is responsible for generating, sending, and forwarding network management information in the form of traps and alerts to a network management focal point, such as Netview, in the APPN network.

Examples

The following is sample output from the **debugappnms** command. In this example an error occurred that caused an alert to be generated.

```
Router# debug appn ms
APPN: ----- MSS00 ---- Deq ALERT_MSU msg
APPN: --- MSP70 --- ALERT MV FROM APPN WITH VALID LGTH
APPN: --- MSCPL --- Find Active FP
APPN: --- MSP30 --- Entering Build MS Transport
APPN: --- MSP31 --- Entering Building Routing Info.
APPN: --- MSP34 --- Entering Build GDS
APPN: --- MSP32 --- Entering Building UOW correlator
APPN: --- MSP34 --- Entering Build GDS
APPN: --- MSP30 --- Building GDS 0x1310
APPN: --- MSP30 --- Building MS Transport
APPN: --- MSP72 --- ACTIVE FP NOT FOUND, SAVE ONLY
APPN: --- MSUTL --- UOW <= 60, ALL COPIED in extract_uow
APPN: --- MSCAT --- by enq_cached_ms QUEUE SIZE OF QUEUE after enq 4
```

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

Table 15: debug appn ms Field Descriptions

Field	Description
APPN	Indicates that this is APPN debugging output.
MSP	Indicates that this is MS component output.

Related Commands

Command	Description
debug appn all	Turns on all possible debugging messages for APPN.

debug appn nof

To display debugging information on Advanced Peer-to-Peer Networking (APPN) Node Operator Facility (NOF) component activity, use the**debugappnnof**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug appn nof

no debug appn nof

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

The NOF component is responsible for processing commands entered by the user such as start, stop, show, and configuration commands. NOF forwards these commands to the proper component and waits for the response.

Examples

The following is sample output from the **debugappnnof** command. In this example, an APPN connection network is being defined.

```
Router# debug appn nof
Turned on event 010000FF
Router# config term
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)# appn connection-network NETA.CISCO
Router(config-appn-cn)# port TR0
Router(config-appn-cn)# complete
router(config)#
APPN: ----- NOF ----- Define Connection Network Verb Received
APPN: ----- NOF ----- send define_cn_t ips to cs
APPN: ----- NOF ----- waiting for define_cn rsp from cs
router(config)#
```

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

Table 16: debug appn nof Field Descriptions

Field	Description
APPN	APPN debugging output.
NOF	NOF component output.
Received	Configuration command was entered.
send	Message was sent to CS.
waiting	Response was expected from CS.

Related Commands

Command	Description
debug appn all	Turns on all possible debugging messages for APPN.

debug appn pc

To display debugging information on Advanced Peer-to-Peer Networking (APPN) Path Control (PC) component activity, use the**debugappnpc**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug appn pc

no debug appn pc

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

The PC component is responsible for passing Message Units (MUs) between the Data Link Control (DLC) layer and other APPN components. PC implements transmission priority by passing higher priority MUs to the DLC before lower priority MUs.

Examples

The following is sample output from the **debugappnpc** command. In this example an MU is received from the network.

```
Router# debug appn pc
Turned on event 040000FF
APPN: ----- PC-----PC Deq REMOTE msg variant_name 2251
APPN: --PC-- mu received to PC lpid: A80AEC
APPN: --PC-- mu received from p_cep_id: 67C6F8
APPN: ----- PC-----PC Deq LSA_IPS from DLC
APPN: --PCX dequeued a DATA.IND
APPN: --- PC processing DL_DATA.ind
APPN: --PC-- mu_error_checker with no error, calling frr
APPN: --PC-- calling frr for packet received on LFSID: 1 2 3
APPN: ----- PC-----PC is sending MU to SC A90396
APPN: ----- SC-----send mu: A90396, rpc: 0, nws: 7, rh.b1: 90
APPN: SC: Send mu.snf: 8, th.b0: 2E, rh.b1: 90, dcf: 8
```

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

Table 17: debug appn pc Field Descriptions

Field	Description
APPN	APPN debugging output.
PC	PC component output.
Deq REMOTE	Message was received from the network.
mu received	Message is an MU.
DATA.IND	MU contains data.

Field	Description
sending MU	MU is session traffic for an ISR session. The MU is forwarded to the Session Connector component for routing.

Related Commands

Command	Description
debug appn all	Turns on all possible debugging messages for APPN.

debug appn ps

To display debugging information on Advanced Peer-to-Peer Networking (APPN) Presentation Services (PS) component activity, use the **debugappnps**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug appn ps

no debug appn ps

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

The PS component is responsible for managing the Transaction Programs (TPs) used by APPN. TPs are used for sending and receiving searches, receiving resource registration, and sending and receiving topology updates.

Examples

The following is sample output from the **debugappnps** command. In this example a CP capabilities exchange is in progress.

```
Router# debug appn ps
Turned on event 200000FF
APPN: ---- CCA --- CP_CAPABILITIES_TP has started
APPN: ---- CCA --- About to wait for Partner to send CP_CAP
APPN: ---- CCA --- Partner LU name: NETA.PATTY
APPN: ---- CCA --- Mode Name: CPSVCMG
APPN: ---- CCA --- CGID: 78
APPN: ---- CCA --- About to send cp_cp_session_act to SS
APPN: ---- CCA --- Waiting for cp_cp_session_act_rsp from SS
APPN: ---- CCA --- Received cp_cp_session_act_rsp from SS
APPN: ---- CCA --- About to send CP_CAP to partner
APPN: ---- CCA --- Send to partner completed with rc=0, 0
APPN: ---- RCA --- Allocating conversation
APPN: ---- RCA --- Sending CP_CAPABILITIES
APPN: ---- RCA --- Getting conversation attributes
APPN: ---- RCA --- Waiting for partner to send CP_CAPABILITIES
APPN: ---- RCA --- Normal processing complete with cgid = 82
APPN: ---- RCA --- Deallocating CP_Capabilities conversation
```

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

Table 18: debug appn ps Field Descriptions

Field	Description
APPN	APPN debugging output.
CCA	CP Capabilities TP output.
RCA	Receive CP Capabilities TP output.

Related Commands

Command	Description
debug appn all	Turns on all possible debugging messages for APPN.

debug appn scm

To display debugging information on Advanced Peer-to-Peer Networking (APPN) Session Connector Manager (SCM) component activity, use the**debugappnscm**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug appn scm

no debug appn scm

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

The SCM component is responsible for the activation and deactivation of the local resources that route an intermediate session through the router.

Examples

The following is sample output from the **debugappnscm** command. In this example an intermediate session traffic is being routed.

```
Router# debug appn scm
Turned on event 020000FF
Router#
APPN: ----- SCM-----SCM Deq a MU
APPN: ----- SCM-----SCM send ISR_INIT to SSI
APPN: ----- SCM-----(i05) Enter compare_fqpcid()
APPN: ----- SCM-----Adding new session_info table entry. addr=A93160
APPN: ----- SCM-----SCM Deq ISR_CINIT message
APPN: ----- SCM-----(i05) Enter compare_fqpcid()
APPN: ----- SCM-----SCM sends ASSIGN_LFSID to ASM
APPN: ----- SCM-----SCM Rcvd sync ASSIGN_LFSID from ASM
APPN: ----- SCM-----SCM PQenq a MU to ASM
APPN: ----- SCM-----SCM Deq a MU
APPN: ----- SCM-----(i05) Enter compare_fqpcid()
APPN: ----- SCM-----SCM PQenq BIND rsp to ASM
```

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

Table 19: debug appn scm Field Descriptions

Field	Description
APPN	APPN debugging output.
SCM	SCM component output.

Related Commands

Command	Description
debug appn all	Turns on all possible debugging messages for APPN.

debug appn ss

To display session services (SS) events, use the**debugappnss**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug appn ss

no debug appn ss

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

The SS component generates unique session identifiers, activates and deactivates control point-to-control point (CP-CP) sessions, and assists logical units (LUs) in initiating and activating LU-LU sessions.

Examples

The following is sample output from the **debugappnss** command. In this example CP-CP sessions between the router and another node are being activated.

```
Router# debug appn ss
Turned on event 100000FF
APPN: ----- SS ----- Deq ADJACENT_CP_CONTACTED message
APPN: ----- SS ----- Deq SESSST_SIGNAL message
APPN: ----- SS ----- Deq CP_CP_SESSION_ACT message
APPN: Sending ADJACENT_NN_1015 to SCM, adj_node_p=A6B980,cp_name=NETA.PATTY
APPN: ----- SS ----- Sending REQUEST_LAST_FRSN message to TRS
APPN: ----- SS ----- Receiving REQUEST_LAST_FRSN_RSP from TRS
APPN: ----- SS ----- Sending ACTIVE CP_STATUS CONLOSER message to DS
APPN: ----- SS ----- Sending ACTIVE CP_STATUS CONLOSER message to MS
APPN: ----- SS ----- Sending ACTIVE CP_STATUS CONLOSER message to TRS
APPN: ----- SS ----- Sending CP_CP_SESSION_ACT_RSP message to CCA TP
APPN: ----- SS ----- Sending PENDING_ACTIVE CP_STATUS CONWINNER message to DS
APPN: ----- SS ----- Sending REQUEST_LAST_FRSN message to TRS
APPN: ----- SS ----- Receiving REQUEST_LAST_FRSN_RSP from TRS
APPN: ----- SS ----- Sending ACT_CP_CP_SESSION message to RCA TP
APPN: ----- SS ----- Deq ASSIGN_PCID message
APPN: ----- SS ----- Sending ASSIGN_PCID_RSP message to someone
APPN: ----- SS ----- Deq INIT_SIGNAL message
APPN: ----- SS ----- Sending REQUEST_COS_TPF_VECTOR message to TRS
APPN: ----- SS ----- Receiving an REQUEST_COS_TPF_VECTOR_RSP from TRS
APPN: ----- SS ----- Sending REQUEST_SINGLE_HOP_ROUTE message to TRS
APPN: ----- SS ----- Receiving an REQUEST_SINGLE_HOP_ROUTE_RSP from TRS
APPN: ----- SS ----- Sending ACTIVATE_ROUTE message to CS
APPN: ----- SS ----- Deq ACTIVATE_ROUTE_RSP message
APPN: ----- SS ----- Sending CINIT_SIGNAL message to SM
APPN: ----- SS ----- Deq ACT_CP_CP_SESSION_RSP message
APPN: -- SS----SS ssp00, act_cp_cp_session_rsp received, sense_code=0, cgid=5C, ips@=A93790
APPN: Sending ADJACENT_NN_1015 to SCM, adj_node_p=A6B980,cp_name=18s
APPN: ----- SS ----- Sending ACTIVE CP_STATUS CONWINNER message to DS
APPN: ----- SS ----- Sending ACTIVE CP_STATUS CONWINNER message to MS
APPN: ----- SS ----- Sending ACTIVE CP_STATUS CONWINNER message to TRS
```

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

Table 20: debug appn ss Field Descriptions

Field	Description
APPN	APPN debugging output.
SS	SS component output.

Related Commands

Command	Description
debug appn all	Turns on all possible debugging messages for APPN.

debug appn trs

To display debugging information on Advanced Peer-to-Peer Networking (APPN) Topology and Routing Services (TRS) component activity, use the**debugappntrs** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug appn trs

no debug appn trs

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

The TRS component is responsible for creating and maintaining the topology database, creating and maintaining the class of service database, and computing and caching optimal routes through the network.

Examples

The following is sample output from the **debugappntrs** command:

```
Router# debug appn trs
Turned on event 400000FF
APPN: ----- TRS ----- Received a QUERY_CPNAME
APPN: ----- TRS ----- Received a REQUEST_ROUTE
APPN: ----- TRS ----- check_node node_name=NETA.LISA
APPN: ----- TRS ----- check_node node_index=0
APPN: ----- TRS ----- check_node node_weight=60
APPN: ----- TRS ----- add index 484 to origin description list
APPN: ----- TRS ----- add index 0 to dest description list
APPN: ----- TRS ----- origin tg_vector is NULL
APPN: ----- TRS ----- weight_to_origin = 0
APPN: ----- TRS ----- weight_to_dest = 0
APPN: ----- TRS ----- u_b_s_f weight = 30
APPN: ----- TRS ----- u_b_s_f prev_weight = 2147483647
APPN: ----- TRS ----- u_b_s_f origin_index = 484
APPN: ----- TRS ----- u_b_s_f dest_index = 0
APPN: ----- TRS ----- b_r_s_f weight = 30
APPN: ----- TRS ----- b_r_s_f origin_index = 484
APPN: ----- TRS ----- b_r_s_f dest_index = 0
APPN: ----- TRS ----- Received a REQUEST_ROUTE
APPN: ----- TRS ----- check_node node_name=NETA.LISA
APPN: ----- TRS ----- check_node node_index=0
APPN: ----- TRS ----- check_node node_weight=60
APPN: ----- TRS ----- check_node node_name=NETA.BART
APPN: ----- TRS ----- check_node node_index=484
APPN: ----- TRS ----- check_node node_weight=60
APPN: ----- TRS ----- add index 484 to origin description list
APPN: ----- TRS ----- add index 0 to dest description list
APPN: ----- TRS ----- origin_tg_weight to non-VN=30
APPN: ----- TRS ----- origin_node_weight to non-VN=60
APPN: ----- TRS ----- weight_to_origin = 90
APPN: ----- TRS ----- weight_to_dest = 0
APPN: ----- TRS ----- u_b_s_f weight = 120
APPN: ----- TRS ----- u_b_s_f prev_weight = 2147483647
APPN: ----- TRS ----- u_b_s_f origin_index = 484
APPN: ----- TRS ----- u_b_s_f dest_index = 0
APPN: ----- TRS ----- b_r_s_f weight = 120
APPN: ----- TRS ----- b_r_s_f origin_index = 484
APPN: ----- TRS ----- b_r_s_f dest_index = 0
```

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

Table 21: debug appn trs Field Descriptions

Field	Description
APPN	APPN debugging output.
TRS	TRS component output.

debug arap

To display AppleTalk Remote Access Protocol (ARAP) events, use the **debugarap** command in privileged EXEC mode. To disable debugging output, use the **no**form of this command.

debug arap {**internal**| **memory**| **mnp4**| **v42bis**} [*linenum* [**aux**| **console**| **tty**| **vty**]]

no debug arap {**internal**| **memory**| **mnp4**| **v42bis**} [*linenum* [**aux**| **console**| **tty**| **vty**]]

Syntax Description

internal	Debugs internal ARA packets.
memory	Debugs memory allocation for ARA.
mnp4	Debugs low-level asynchronous serial protocol.
v42bis	Debugs V.42bis compression.
linenum	(Optional) Line number. The number ranges from 0 to 999, depending on what type of line is selected.
aux	(Optional) Auxiliary line.
console	(Optional) Primary terminal line.
tty	(Optional) Physical terminal asynchronous line.
vty	(Optional) Virtual terminal line.

Command Modes

Privileged EXEC

Usage Guidelines

Use the **debugarap** command with the **debugcallback** command on access servers to debug dialin and callback events.

Use the **debugmodem** command to help catch problems related to ARAP autodetection (that is, **autoselectarap**). These problems are very common and are most often caused by modems, which are the most common cause of failure in ARAP connection and configuration sessions.

Examples

The following is sample output from the **debugarapinternal** command:

```
Router# debug arap internal
```

```
ARAP: ---------- SRVRVERSION ----------
ARAP: ---------- ACKing 0 ----------
ARAP: ---------- AUTH_CHALLENGE ----------
arapsec_local_account setting up callback
ARAP: ---------- ACKing 1 ----------
ARAP: ---------- AUTH_RESPONSE ----------
```

```
arap_startup initiating callback ARAP 2.0
ARAP: ---------- CALLBACK ----------
TTY7 Callback process initiated, user: dialback dialstring 40
TTY7 Callback forced wait = 4 seconds
TTY7 ARAP Callback Successful - await exec/autoselect pickup
TTY7: Callback in effect
ARAP: ---------- STARTINFOFROMSERVER ----------
ARAP: ---------- ACKing 0 ----------
ARAP: ---------- ZONELISTINFO ----------
ARAP: ---------- ZONELISTINFO ----------
ARAP: ---------- ZONELISTINFO ----------
ARAP: ---------- ZONELISTINFO ----------
ARAP: ---------- ZONELISTINFO ----------
```

Related Commands

Command	Description
debug callback	Displays callback events when the router is using a modem and a chat script to call back on a terminal line.
debug modem	Observes modem line activity on an access server.

debug archive config timestamp

To enable debugging of the processing time for each integral step of a configuration replace operation and the size of the configuration files being handled, use the **debug archive config timestamp** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug archive config timestamp

no debug archive config timestamp

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.3(7)T	This command was introduced.
12.2(25)S	This command was integrated into Cisco IOS Release 12.2(25)S.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(31)SB2	This command was implemented on the Cisco 10000 series.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.
12.2(33)SB	This command was integrated into Cisco IOS Release 12.2(33)SB and implemented on the Cisco 10000 series.
Cisco IOS XE Release 3.9S	This command was integrated into Cisco IOS XE Release 3.9S.

Examples

The following is sample output from the **debug archive config timestamp** command:

```
Device# debug archive config timestamp
Device# configure replace disk0:myconfig force
Timing Debug Statistics for IOS Config Replace operation:
       Time to read file slot0:sample_2.cfg = 0 msec (0 sec)
       Number of lines read:55
       Size of file        :1054
Starting Pass 1
       Time to read file system:running-config = 0 msec (0 sec)
       Number of lines read:93
       Size of file        :2539
       Time taken for positive rollback pass = 320 msec (0 sec)
       Time taken for negative rollback pass = 0 msec (0 sec)
       Time taken for negative incremental diffs pass = 59 msec (0 sec)
       Time taken by PI to apply changes = 0 msec (0 sec)
       Time taken for Pass 1 = 380 msec (0 sec)
```

```
Starting Pass 2
       Time to read file system:running-config = 0 msec (0 sec)
       Number of lines read:55
       Size of file        :1054
       Time taken for positive rollback pass = 0 msec (0 sec)
       Time taken for negative rollback pass = 0 msec (0 sec)
       Time taken for Pass 2 = 0 msec (0 sec)
Total number of passes:1
Rollback Done
```

Related Commands

Command	Description
debug archive versioning	Enables debugging of the Cisco configuration archive activities.

debug archive log config persistent

To turn on debugging of configuration logging persistent events and display the results, use the **debugarchivelogconfigpersistent**command in privileged EXEC mode. To disable the debugging and display of the archive events, use the **no** form of this command.

debug archive log config persistent

no debug archive log config persistent

Syntax Description

This command has no arguments or keywords.

Command Default

If this command is not entered, there is no debugging or display of the configuration logging persistent events in the archive.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(33)SRA	This command was introduced.
12.4(11)T	This command was integrated into Cisco IOS Release 12.4(11)T.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.
12.2(33)SB	This command was integrated into Cisco IOS Release 12.2(33)SB.

Usage Guidelines

The configuration logger feature must be enabled in order for the debug capability to work.

Examples

The following example turns on the debugging feature and displays the configuration logging persistent events:

```
Router# debug archive config log persistent
Router# archive log config persistent save
Configuration logging persistent save triggered.
Saving the config log to disk0:IOS-Config-Logger-database'.
Command 'interface eth0' saved
Command 'ip address 10.1.1.1 255.255.255.0' saved
Command 'no shut' saved
Router#
```

Related Commands

Command	Description
archive log config persistent save	Saves the persisted commands in the configuration log to the Cisco IOS secure file system.

debug archive versioning

To enable debugging of the Cisco configuration archive activities, use the **debug archive versioning** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug archive versioning

no debug archive versioning

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.3(7)T	This command was introduced.
12.2(25)S	This command was integrated into Cisco IOS Release 12.2(25)S.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(31)SB2	This command was implemented on the Cisco 10000 series.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.
12.2SX	This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.
12.2(33)SB	This command was integrated into Cisco IOS Release 12.2(33)SB and implemented on the Cisco 10000 series.
12.2(33)SRE	This command was modified. It was integrated into Cisco IOS Release 12.2(33)SRE.
Cisco IOS XE Release 3.9S	This command was integrated into Cisco IOS XE Release 3.9S.

Examples

The following is sample output from the **debug archive versioning** command:

```
Device# debug archive versioning
Jan  9 06:46:28.419:backup_running_config
Jan  9 06:46:28.419:Current = 7
Jan  9 06:46:28.443:Writing backup file disk0:myconfig-7
Jan  9 06:46:29.547: backup worked
```

Related Commands

Command	Description
debug archive config timestamp	Enables debugging of the processing time for each integral step of a configuration replace operation and the size of the configuration files being handled.

debug arp

To enable debugging output for Address Resolution Protocol (ARP) transactions, use the **debug arp** command in privileged EXEC mode. To disable debugging, use the **no** form of this command.

debug arp [**vrf** *vrf-name*| **global**] [*arp-entry-event*| *arp-table-event*| **ha**| *interface-interaction*]

no debug arp [**vrf** *vrf-name*| **global**] [*arp-entry-event*| *arp-table-event*| **ha**| *interface-interaction*]

Syntax Description

vrf *vrf-name*	(Optional) Enables debug trace for a specific VPN routing and forwarding (VRF) instance.
global	(Optional) Enables global VRF debugging.
arp-entry-event	(Optional) ARP entry events for which a debug trace is enabled. Keywords are as follows: • **dynamic—**Enables debugging output for dynamic ARP entry events. • **interface—**Enables debugging output for interface ARP entry events. • **static—**Enables debugging output for static ARP entry events. • **subblocking—**Enables debugging output for ARP subblocking events.
arp-table-event	(Optional) ARP entry events for which a debug trace is enabled. Keywords are as follows: • **database—**Enables debugging output for ARP database operations. • **table—**Enables debugging output for ARP table operations. • **timer—**Enables debugging output for ARP timer operations.
ha	(Optional) Enables debug trace for ARP high availability (HA) events. **Note** This keyword is available only on HA-capable platforms (that is, Cisco networking devices that support dual Route Processors [RPs]).

interface-interaction	(Optional) ARP interface interaction for which a debug trace is enabled. Keywords are as follows: • **adjacency—**Enables debugging output for ARP interface events and Cisco Express Forwarding adjacency interface events. • **application—**Enables debugging output for ARP application interface events.

Command Default

Debugging output is disabled for ARP transactions. To enable ARP packet debugging, use this command without a keyword.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
10.3	This command was introduced.
12.4(11)T	The following keywords were added: **adjacency**, **application**, **dynamic**, **ha**, **interface**, **static**, **subblocking**, **table**, and **timer**.
12.2(33)SRE	This command was modified. The **database** keyword was added.
15.1(1)SY	This command was modified. The **vrf** *vrf-name* keyword-argument pair and **global** keyword were added.

Usage Guidelines

The debugging information shows whether the device is sending ARP packets and receiving ARP packets. Use this command when only some nodes on a TCP/IP network are responding.

The amount of debug information displayed is filtered based on an interface, an access list, or both, as specified by the **debug list** command.

To list the debugging options enabled on this device, use the **show debugging** command.

Examples

The following example shows how to enable ARP packet debugging filtered on ARP table entries for the host at 192.0.2.10:

```
Device(config)# access-list 10 permit host 192.0.2.10
Device(config)# exit
Device# debug list 10
Device# debug arp

ARP packet debugging is on
        for access list: 10
```

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

```
IP ARP: sent req src 192.0.2.7 0000.0c01.e117, dst 192.0.2.96 0000.0000.0000
IP ARP: rcvd rep src 192.0.2.96 0800.2010.b908, dst 192.0.2.7
IP ARP: rcvd req src 172.16.6.10 0000.0c00.6fa2, dst 192.0.2.62
IP ARP: rep filtered src 192.0.2.7 aa92.1b36.a456, dst 255.255.255.255 ffff.ffff.ffff
IP ARP: rep filtered src 192.0.2.240 0000.0c00.6b31, dst 192.0.2.7 0800.2010.b908
```

In the output, each line of output represents an ARP packet that the device has sent or received. Explanations for the individual lines of output follow.

The first line indicates that the device at IP address 192.0.2.7 and with the MAC address 0000.0c01.e117 sent an ARP request for the MAC address of the host at 192.0.2.96. The series of zeros (0000.0000.0000) following this address indicate that the device is currently unaware of the MAC address.

```
IP ARP: sent req src 192.0.2.7 0000.0c01.e117, dst 192.0.2.96 0000.0000.0000
```

The second line indicates that the device at IP address 192.0.2.7 receives a reply from the host at 192.0.2.96 indicating that the host MAC address is 0800.2010.b908:

```
IP ARP: rcvd rep src 192.0.2.96 0800.2010.b908, dst 192.0.2.7
```

The third line indicates that the device receives an ARP request from the host at 172.16.6.10 requesting the MAC address for the host at 192.0.2.62:

```
IP ARP: rcvd req src 172.16.6.10 0000.0c00.6fa2, dst 192.0.2.62
```

The fourth line indicates that another host on the network attempted to send the device an ARP reply for its own address. The device ignores meaningless replies. Usually, meaningless replies happen if a bridge is being run in parallel with the device and is allowing ARP to be bridged. This condition indicates a network misconfiguration.

```
IP ARP: rep filtered src 192.0.2.7 aa92.1b36.a456, dst 255.255.255.255 ffff.ffff.ffff
```

The fifth line indicates that another host on the network attempted to inform the device that it is on network 192.0.2.240, but the device does not know that the network is attached to a different device interface. The remote host (probably a PC or an X terminal) is misconfigured. If the device were to install this entry, the device would deny service to the real machine on the proper cable.

```
IP ARP: rep filtered src 192.0.2.240 0000.0c00.6b31, dst 192.0.2.7 0800.2010.b908
```

The following example shows that debugging information related to vpn1 on Ethernet interface 0/0 will be logged:

```
Device> enable
Device# configure terminal
Device(config)# interface ethernet0/0
Device(config-if)# vrf forwarding vpn1
Device(config-if)# end
Device# debug arp vrf vpn1
```

Related Commands

Command	Description
access-list (extended-ibm)	Configures the extended access list mechanism for filtering frames by both source and destination addresses and arbitrary bytes in the packet.
debug list	Enables filtering of debug trace on a per-interface or per-access list basis.
show debugging	Lists the debugging options enabled on this device.

debug ase

Note Effective with Cisco IOS Release 12.4(24), the **debugase** command is not available in Cisco IOS software.

To gather Automatic Signature Extraction (ASE) error, log, messaging, reporting, status, and timer information, use the **debugase** command in privileged EXEC mode. To disable error, log, messaging, reporting, status, and timer information, use the **no** form of this command.

debug ase {**errors**| **log**| **messages**| **reports**| **status**| **timing**}

no debug ase {**errors**| **log**| **messages**| **reports**| **status**| **timing**}

Syntax Description

errors	Displays ASE error information.
log	Displays ASE logging information.
messages	Displays ASE messaging information.
reports	Displays ASE reports.
status	Displays ASE status information.
timing	Displays ASE timer information.

Command Default

Disabled

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.4(15)T	This command was introduced.
12.4(24)	This command was removed.

Usage Guidelines

This command is used on the Cisco 1800, 2800, and 7200 series routers, Cisco 7301 router, and Integrated Services Routers (ISRs) as ASE sensors.

Related Commands

Command	Description
ase collector	Enters the ASE collector server IP address so that the ASE sensor has IP connectivity to the ASE collector.
ase enable	Enables the ASE feature on a specified interface.
ase group	Identifies the TIDP group number for the ASE feature.
ase signature extraction	Enables the ASE feature globally on the router.
clear ase signature	Clears ASE signatures that were detected on the router.
show ase	Displays the ASE run-time status, which includes the TIDP group number.

debug asnl events

To trace event logs in the Application Subscribe Notify Layer (ASNL), use the **debugasnlevents** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug asnl events

no debug asnl events

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(4)T	This command was introduced.

Usage Guidelines

This command traces the event logs in the ASNL, which serves as the interface layer between the application and protocol stacks. Event logs are generated during normal subscription processing, when the application responds to the notification request and when the session history table is updated.

Examples

The following example shows the ASNL subscription table being generated and the associated subscription timers as the application responds to the subscription request. The response timer is started to determine if the application responds to the notification request. If the application that made the subscription does not respond to the notification request within 5 seconds, the system automatically removes the subscription. The session-history-record deletion timer is also started. When the timer expires, the history record is removed from the active subscription table.

```
Router# debug asnl events

Application Subscribe Notify Layer Events debugging is on
*May  4 06:26:19.091://-1//ASNL:SUB-1:/asnl_process_is_up:Creating subscription table
*May  4 06:26:19.091://5//ASNL:SUB1:/asnl_subscribe:resp = ASNL_SUBCRIBE_PENDING[2]
*May  4 06:26:19.615://5//ASNL:SUB1:/asnl_start_timer:timer (0x63146C44)starts - delay
(5000)
*May  4 06:26:19.619://-1//ASNL:SUB1:/asnl_stop_timer:timer(0x63146C44) stops
*May  4 06:26:19.619://-1//ASNL:SUB1:/asnl_notify_ack:ret=0x0
c5300-5#
*May  4 06:26:24.631://5//ASNL:SUB1:/asnl_start_timer:timer (0x63146C44)starts - delay
(5000)
*May  4 06:26:24.631://-1//ASNL:SUB1:/asnl_stop_timer:timer(0x63146C44) stops
*May  4 06:26:24.635://-1//ASNL:SUB1:/asnl_notify_ack:ret=0x0
c5300-5#
*May  4 06:26:29.647://5//ASNL:SUB1:/asnl_start_timer:timer (0x63146C44)starts - delay
(5000)
*May  4 06:26:29.647://-1//ASNL:SUB1:/asnl_stop_timer:timer(0x63146C44) stops
```

```
*May  4 06:26:29.651://-1//ASNL:SUB1:/asnl_notify_ack:ret=0x0
*May  4 06:26:34.663://5//ASNL:SUB1:/asnl_start_timer:timer (0x63146C44)starts - delay
(5000)
*May  4 06:26:34.663://-1//ASNL:SUB1:/asnl_stop_timer:timer(0x63146C44) stops
*May  4 06:26:34.667://-1//ASNL:SUB-1:/asnl_create_session_history:Creating Session History

*May  4 06:26:34.667://-1//ASNL:SUB-1:/asnl_insert_session_history_record:starting history
 record deletion_timer of 15 minutes
*May  4 06:26:34.667://-1//ASNL:SUB1:/asnl_notify_ack:ret=0x0
```

Related Commands

Command	Description
clear subscription	Clears all active subscriptions or a specific subscription.
show subscription	Displays information about ASNL-based and non-ASNL-based SIP subscriptions.
subscription asnl session history	Specifies how long to keep ASNL subscription history records and how many history records to keep in memory.

debug asp packet

To display information on all asynchronous security protocols (ASPs) operating on the router, use the **debugasppacket**command in privileged EXEC mode. To disable debugging output, use the **no**form of this command.

debug asp packet

no debug asp packet

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Usage Guidelines

The router uses asynchronous security protocols from companies including ADT Security Systems, Inc., Adplex, and Diebold to transport alarm blocks between two devices (such as a security alarm system console and an alarm panel). The alarm blocks are transported in pass-through mode using BSTUN encapsulation.

Examples

The following is partial sample output from the **debugasppacket** command for asynchronous security protocols when packet debugging is enabled on an asynchronous line carrying Diebold alarm traffic. In this example, two polls are sent from the Diebold alarm console to two alarm panels that are multidropped from a single EIA/TIA-232 interface. The alarm panels have device addresses F0 and F1. The example trace indicates that F1 is responding and F0 is not responding. At this point, you need to examine the physical link and possibly use a datascope to determine why the device is not responding.

```
Router# debug asp packet
12:19:48: ASP: Serial5: ADI-Rx: Data (4 bytes): F1FF4C42
12:19:49: ASP: Serial5: ADI-Tx: Data (1 bytes): 88
12:19:49: ASP: Serial5: ADI-Rx: Data (4 bytes): F0FF9B94
12:20:47: ASP: Serial5: ADI-Rx: Data (4 bytes): F1FF757B
12:20:48: ASP: Serial5: ADI-Tx: Data (1 bytes): F3
12:20:48: ASP: Serial5: ADI-Rx: Data (4 bytes): F0FFB1BE
12:21:46: ASP: Serial5: ADI-Rx: Data (4 bytes): F1FFE6E8
12:21:46: ASP: Serial5: ADI-Tx: Data (1 bytes): 6F
12:21:46: ASP: Serial5: ADI-Rx: Data (4 bytes): F0FFC1CE
```

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

Table 22: debug asp packet Field Descriptions

Field	Description
ASP	Asyncronous security protocol packet.
Serial5	Interface receiving and sending the packet.
ADI-Rx	Packet is being received.
ADI-T	Packet is being sent.

Field	Description
Data (*n* bytes)	Type and size of the packet.
F1FF4c42	Alarm panel device address.

debug aspp event

To display asynchronous point of sale (APOS) event debug messages, use the **debugasppevent**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug aspp event

no debug aspp event

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(2)T	This command was introduced.

Usage Guidelines

The **debugasppevent**command should be used with the **debugaspppacket** command to display all available details of the APOS call flow.

Examples

The following is sample output from the **debugasppevent**command for a simple transaction:

```
Router# debug aspp event
ASPP event debugging is on
Router#
ASPP APOS: Serial0/1: Serial HayesAT: state = DISCONNECTED
ASPP APOS: Serial0/1: Received HayesAT DIAL: state = DISCONNECTED
ASPP APIP: Serial0/1: Serial ENABLE: state = CONNECTING
ASPP APIP: Serial0/1: Network ENABLE: state = CONNECTING
ASPP APOS: Serial0/1: Send HayesAT CONNECT 9600: state = CONNECTED
ASPP APOS: Serial0/1: Response timer expired: state = CONNECTED
ASPP APOS: Serial0/1: Response timer expired: state = CONNECTED
ASPP APOS: Serial0/1: Serial DATA: state = CONNECTED
ASPP APIP: Serial0/1: Serial DATA: state = CONNECTED
ASPP APIP: Serial0/1: Network DATA: state = CONNECTED
ASPP APOS: Serial0/1: Serial ACK: state = CONNECTED
ASPP APOS: Serial0/1: Disconnect timer expired: state = DISCONNECT WAIT
ASPP APIP: Serial0/1: Serial DISABLE: state = DISCONNECTING
ASPP APIP: Serial0/1: Network DISABLE: state = DISCONNECTING
```

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

Table 23: debug aspp event Field Descriptions

Field	Description
Serial ENABLE:	Enable event received from the serial interface.

Field	Description
Network ENABLE:	Enable event received from the network.
Send HayesAT CONNECT	Interpreted version of the Hayes AT command that is sent to the serial interface.
Response timer expired	The response timer has expired.
Serial DATA:	Data received from the serial interface.
Network DATA:	Data received from the network.
Disconnect timer expired	Hayes AT event is received by the serial interface.
Serial ACK:	Acknowledgment received from the serial interface.
Serial DISABLE:	Disable event received from the serial interface.
Network DISABLE:	Disable event received from the network.

Related Commands

Command	Description
debug aspp packet	Displays APOS packet debug messages.

debug aspp packet

To display asynchronous point of sale (APOS) packet debug messages, use the **debugaspppacket**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug aspp packet

no debug aspp packet

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(2)T	This command was introduced.

Usage Guidelines

The **debugaspppacket**command should be used with the **debugasppevent** command to display all available details of the APOS call flow.

Examples

The following is sample output from the **debugaspppacket**command for a simple transaction:

```
Router# debug aspp packet
ASPP event debugging is on
Router#
ASPP:Serial1/7:ADI-rx:Data (14 bytes): 415456302644325331313D35300D
ASPP:Serial1/7:ADI-tx:Data (2 bytes): 300D
ASPP:Serial1/7:ADI-rx:Data (27 bytes): 4154583453393D3153373D323444543138303039
ASPP:Serial1/7:ADI-tx:Data (3 bytes): 31320D
ASPP:Serial1/7:ADI-tx:Data (1 bytes): 05
ASPP:Serial1/7:ADI-rx:Data (5 bytes): 0212340325
ASPP:Serial1/7:ADI-tx:Data (5 bytes): 025678032D
ASPP:Serial1/7:ADI-rx:Data (1 bytes): 06
ASPP:Serial1/7:ADI-tx:Data (1 bytes): 04
```

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

Table 24: debug aspp packet Field Descriptions

Field	Description
ASPP	Indicates that this is an ASPP debug message.
Serial1/7:	The interface that received or transmitted the packet.

Field	Description
ADI-rx	Indicates a received packet.
ADI-tx	Indicates a transmitted packet.

Related Commands

Command	Description
debug aspp event	Displays APOS event debug messages.

debug async async-queue

To display debug messages for asynchronous rotary line queueing, use the **debugasyncasync-queue**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug async async-queue

no debug async async-queue

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.1(1)T	This command was introduced.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Examples

The following example starts the asynchronous rotary line queueing debugging display:

```
Router# debug async async-queue
*Mar  2 03:50:28.377: AsyncQ: First connection to be queued - starting the AsyncQ manager
*Mar  2 03:50:28.377: AsyncQ: Enabling the AsyncQ manager
*Mar  2 03:50:28.377: AsyncQ: Started the AsyncQ manager process with pid 98
*Mar  2 03:50:28.381: AsyncQ: Created a Waiting TTY on TTY66 with pid 99
*Mar  2 03:50:30.164: WaitingTTY66: Did Authentication on waiting TTY (VTY)
*Mar  2 03:50:30.168: AsyncQ: Received ASYNCQ_MSG_ADD
*Mar  2 03:50:30.168: AsyncQ: New queue, adding this connection as the first element
*Mar  2 03:50:34.920: AsyncQ: Created a Waiting TTY on TTY67 with pid 100
*Mar  2 03:50:36.783: WaitingTTY67: Did Authentication on waiting TTY (VTY)
*Mar  2 03:50:36.787: AsyncQ: Received ASYNCQ_MSG_ADD
*Mar  2 03:50:36.787: AsyncQ: Queue exists, adding this connection to the end of the queue
```

Related Commands

Command	Description
debug ip tcp transactions	Enables the IP TCP transactions debugging display to observe significant transactions such as state changes, retransmissions, and duplicate packets.
debug modem	Enables the modem debugging display to observe modem line activity on an access server.

debug atm autovc

To display information about autoprovisioned ATM permanent virtual circuit (PVC) events and errors, use the debug atm autovc command in privileged EXEC mode. To disable the display of information about autoprovisioned ATM PVC events and errors, use the no form of this command.

debug atm autovc {**event**| **error**| **all**}

no debug atm autovc {**event**| **error**| **all**}

Syntax Description

event	Displays all autoprovisioned PVC events.
error	Displays all autoprovisioned PVC errors.
all	Displays all autoprovisioned PVC events and errors.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(15)B	This command was introduced.
15.0(1)M	This command was integrated into Cisco IOS Release 15.0(1)M.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(33)SRE	This command was integrated into Cisco IOS Release 12.2(33)SRE.

Examples

The following example shows output for the debug atm autovc command for all autoprovisioned PVC events and errors:

```
Router# debug atm autovc all
AutoVC all debugging is on
00:09:03:AutoVC(ATM1/0):1/101 enqueued
```

This message indicates that there is incoming traffic on PVC 1/101 and the PVC is enqueued to be processed.

```
00:09:03:AutoVC(ATM1/0):process VC 1/101
```

This message indicates that PVC 1/101 is in the process of being autoprovisioned.

```
00:09:03:AutoVC(ATM1/0.1):bring up vc 1/101
```

This message indicates that PVC 1/101 is being brought up.

```
00:09:03:%ATM-5-UPDOWN:Interface ATM1/0.1, Changing autovc 1/101 to UP
```

This message indicates that the PVC was brought up successfully.

Related Commands

Command	Description
create on-demand	Configures ATM PVC autoprovisioning, which enables a PVC or range of PVCs to be created automatically on demand.

debug atm bundle error

To display debug messages for switched virtual circuit (SVC) bundle errors, use the **debugatmbundleerror**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug atm bundle error

no debug atm bundle error

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(4)T	This command was introduced.

Examples

The following example provides output for the **debugatmbundleerror**command:

```
Router#
debug atm bundle error
```

Related Commands

Command	Description
debug atm bundle events	Displays SVC bundle events.

debug atm bundle events

To display switched virtual circuit (SVC) bundle events, use the **debugatmbundleevents**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug atm bundle events

no debug atm bundle events

Syntax Description

This command has no arguments or keywords.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(4)T	This command was introduced.

Examples

The following example provides output for the **debugatmbundleevents**command:

```
Router# debug atm bundle events
01:14:35:BUNDLE EVENT(test):b_update_vc for four with bstate 1, vc_state4
01:14:35:BUNDLE EVENT(test):bmupdate active precedence 0x01 0
01:14:35:BUNDLE EVENT(test):bmupdate active precedence 0x02 0
01:14:35:BUNDLE EVENT(test):bmupdate active precedence 0x04 0
01:14:35:BUNDLE EVENT(test):bmupdate active precedence 0x08 0
01:14:35:BUNDLE EVENT(test):bmupdate active precedence 0x10 0
01:14:35:BUNDLE EVENT(test):bmupdate active precedence 0x20 0
01:14:35:BUNDLE EVENT(test):bmupdate active precedence 0x40 0
01:14:35:BUNDLE EVENT(test):bmupdate active precedence 0x80 0 -
01:14:35:BUNDLE EVENT(test):bundle precedence updated
```

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

Table 25: debug atm bundle events Field Descriptions

Field	Description
01:14:35	Local time on the router in hours:minutes:seconds.
BUNDLE EVENT(test)	Bundle event for bundle by that name.
b_update_vc for four with bstate 1, vc_state 1	Test describing the bundle event.

Related Commands

Command	Description
debug atm bundle error	Displays debug messages for SVC bundle errors.

debug atm cell-packing

To enable the display of ATM cell relay cell-packing debugging information, use the debug atm cell-packing command in privileged EXEC mode. To disable the display of debugging information, use the no form of this command.

debug atm cell-packing

no debug atm cell-packing

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(25)S	This command was introduced.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Examples

The following example enables debugging for ATM virtual circuits (VCs) that have been configured with cell packing:

```
Router# debug atm cell-packing
ATM Cell Packing debugging is on
00:09:04: ATM Cell Packing: vc 1/100 remote mncp 22 validated
```

The following example enables debugging for permanent virtual paths (PVPs) that have been configured with cell packing:

```
Router# debug atm cell-packing
ATM Cell Packing debugging is on
00:12:33: ATM Cell Packing: vp 1 remote mncp 22 validated
```

The output indicates that the router received the MNCP information from the remote PE router.

Related Commands

Command	Description
atm mcpt-timers	Creates cell-packing timers that specify how long the PE router can wait for cells to be packed into an MPLS or L2TPv3 packet.
cell-packing	Enables the packing of multiple ATM cells into a single MPLS or L2TPv3 packet.

Command	Description
show atm cell-packing	Displays information about the VCs and VPs that have ATM cell relay over MPLS or L2TPv3 cell packing enabled.

debug atm events

To display ATM events, use the **debugatmevents**command in privileged EXEC mode. To disable event debugging output, use the **no** form of this command.

debug atm events

no debug atm events

Syntax Description

This command has no arguments or keywords.

Command Default

ATM event debugging is disabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.1(3)XJ	This command was introduced on the Cisco 1700 series routers.
12.1(5)XR1	This command was implemented on the Cisco IAD2420 Series.
12.2(4)T	This command was integrated into Cisco IOS Release 12.2(4)T.

Usage Guidelines

This command displays ATM events that occur on the ATM interface processor and is useful for diagnosing problems in an ATM network. It provides an overall picture of the stability of the network. In a stable network, the **debugatmevents** command does not return any information. If the command generates numerous messages, the messages can indicate the possible source of problems.

When configuring or making changes to a router or interface for ATM, enable the **debugatmevents**command. Doing so alerts you to the progress of the changes or to any errors that might result. Also use this command periodically when you suspect network problems.

Examples

The following is sample output from the **debugatmevents** command:

```
Router# debug atm events

RESET(ATM4/0): PLIM type is 1, Rate is 100Mbps
aip_disable(ATM4/0): state=1
config(ATM4/0)
aip_love_note(ATM4/0): asr=0x201
aip_enable(ATM4/0)
aip_love_note(ATM4/0): asr=0x4000
aip_enable(ATM4/0): restarting VCs: 7
aip_setup_vc(ATM4/0): vc:1 vpi:1 vci:1
aip_love_note(ATM4/0): asr=0x200
aip_setup_vc(ATM4/0): vc:2 vpi:2 vci:2
aip_love_note(ATM4/0): asr=0x200
aip_setup_vc(ATM4/0): vc:3 vpi:3 vci:3
```

```
aip_love_note(ATM4/0): asr=0x200
aip_setup_vc(ATM4/0): vc:4 vpi:4 vci:4
aip_love_note(ATM4/0): asr=0x200
aip_setup_vc(ATM4/0): vc:6 vpi:6 vci:6
aip_love_note(ATM4/0): asr=0x200
aip_setup_vc(ATM4/0): vc:7 vpi:7 vci:7
aip_love_note(ATM4/0): asr=0x200
aip_setup_vc(ATM4/0): vc:11 vpi:11 vci:11
aip_love_note(ATM4/0): asr=0x200
```

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

Table 26: debug atm events Field Descriptions

Field	**Description**
PLIM type	Indicates the interface rate in megabits per second (Mbps). Possible values are: • 1 = TAXI(4B5B) 100 Mbps • 2 = SONET 155 Mbps • 3 = E3 34 Mbps
state	Indicates current state of the ATM Interface Processor (AIP). Possible values are: • 1 = An ENABLE will be issued soon. • 0 = The AIP will remain shut down.
asr	Defines a bitmask, which indicates actions or completions to commands. Valid bitmask values are: • 0x0800 = AIP crashed, reload may be required. • 0x0400 = AIP detected a carrier state change. • 0x0n00 = Command completion status. Command completion status codes are: • n = 8 Invalid Physical Layer Interface Module (PLIM) detected • n = 4 Command failed • n = 2 Command completed successfully • n = 1 CONFIG request failed • n = 0 Invalid value

The following line indicates that the AIP was reset. The PLIM TYPE detected was 1, so the maximum rate is set to 100 Mbps.

```
RESET(ATM4/0): PLIM type is 1, Rate is 100Mbps
```

The following line indicates that the AIP was given a shutdown command, but the current configuration indicates that the AIP should be up:

```
aip_disable(ATM4/0): state=1
```

The following line indicates that a configuration command has been completed by the AIP:

```
aip_love_note(ATM4/0): asr=0x201
```

The following line indicates that the AIP was given a no shutdown command to take it out of shutdown:

```
aip_enable(ATM4/0)
```

The following line indicates that the AIP detected a carrier state change. It does not indicate that the carrier is down or up, only that it has changed.

```
aip_love_note(ATM4/0): asr=0x4000
```

The following line of output indicates that the AIP enable function is restarting all permanent virtual circuits (PVCs) automatically:

```
aip_enable(ATM4/0): restarting VCs: 7
```

The following lines of output indicate that PVC 1 was set up and a successful completion code was returned:

```
aip_setup_vc(ATM4/0): vc:1 vpi:1 vci:1
aip_love_note(ATM4/0): asr=0x200
```

debug atm ha-error

To debug ATM) high-availability (HA errors on a networking device, use the **debugatmha-error** command in privileged EXEC mode. To disable the display of debugging output, use the **no** form of this command.

debug atm ha-error

no debug atm ha-error

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging is disabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(22)S	This command was introduced on Cisco 7500, 10000, and 12000 series Internet routers.
12.2(18)S	This command was integrated into Cisco IOS Release 12.2(18)S on Cisco 7500 series routers.
12.2(20)S	Support was added for the Cisco 7304 router. The Cisco 7500 series router is not supported in Cisco IOS Release 12.2(20)S.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.

Examples

The following example displays debug messages regarding ATM HA errors on the networking device:

```
Router# debug atm ha-error
```

Related Commands

Command	Description
debug atm ha-events	Debugs ATM HA events on the networking device.
debug atm ha-state	Debugs ATM HA state information on the networking device.

debug atm ha-events

To debug ATM high-availability (HA) events on the networking device, use the **debugatmha-events** command in privileged EXEC mode. To disable the display of debugging output, use the **no** form of this command.

debug atm ha-events

no debug atm ha-events

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging is disabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(22)S	This command was introduced on Cisco 7500, 10000, and 12000 series Internet routers.
12.2(18)S	This command was integrated into Cisco IOS Release 12.2(18)S on Cisco 7500 series routers.
12.2(20)S	Support was added for the Cisco 7304 router. The Cisco 7500 series router is not supported in Cisco IOS Release 12.2(20)S.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.

Examples

The following example displays debug messages regarding ATM HA events on the networking device:

```
Router# debug atm ha-events
```

Related Commands

Command	Description
debug atm ha-error	Debugs ATM HA errors on the networking device.
debug atm ha-state	Debugs ATM HA state information on the networking device.

debug atm ha-state

To debug ATM high-availability (HA) state information on the networking device, use the **debugatmha-state** command in privileged EXEC mode. To disable the display of debugging output, use the **no** form of this command.

debug atm ha-state

no debug atm ha-state

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(22)S	This command was introduced on Cisco 7500, 10000, and 12000 series Internet routers.
12.2(18)S	This command was integrated into Cisco IOS Release 12.2(18)S on Cisco 7500 series routers.
12.2(20)S	Support was added for the Cisco 7304 router. The Cisco 7500 series router is not supported in Cisco IOS Release 12.2(20)S.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.

Examples

The following example displays debug messages regarding the ATM HA state on the networking device:

```
Router# debug atm ha-state
```

Related Commands

Command	Description
debug atm ha-error	Debugs ATM HA errors on the networking device.
debug atm ha-events	Debugs ATM HA events on the networking device.

debug atm l2transport

To enable the display of debugging information related to ATM over MPLS, use the debug atm l2transport command in privileged EXEC mode. To disable the display of debugging information, use the no form of this command.

debug atm l2transport

no debug atm l2transport

Syntax Description

This command has no arguments or keywords.

Command Default

Debugging of ATM over MPLS is not enabled.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.0(25)S	This command was introduced.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Examples

The following example shows the events and messages when configuring ATM Cell Relay over MPLS in VP mode.

```
Router# debug atm l2transport
ATM L2transport Events and Errors debugging is on
Router# show debug
ATM L2transport:
  ATM L2transport Events and Errors debugging is on
Router(config-if)# atm pvp 24 l2transport
Router(cfg-if-atm-l2trans-pvp)#  xconnect 11.11.11.11 700 pw-class vp
Router(cfg-if-atm-l2trans-pvp)# end
00:14:51: ATM L2trans(ATM1/0): VP 24 is created
00:14:51: ATM L2trans(ATM1/0): ckt_type 10, ckt_id 1000024 UP
00:14:51: ATM L2trans(ATM1/0): VP 24, response is connect forwarded
```

The following example shows the events and messages when deleting a PVP.

```
Router(config-if)# no atm pvp 24 l2transport
00:14:37: ATM L2trans(ATM1/0): ckt_type 10, ckt_id 1000024 DOWN
00:14:37: ATM L2trans(ATM1/0): ckt_type 10, ckt_id 1000024 DOWN
00:14:37: ATM L2trans(ATM1/0): remove xconnect circuit_type=10,
circuit_id=1000024
00:14:37: ATM L2trans(ATM1/0): ckt_type 10, ckt_id 1000024 DOWN
```

Related Commands

Command	Description
show mpls l2transport vc	Displays information about AToM circuits that have been enabled to route Layer 2 packets on a router.

debug atm lfi

To display multilink PPP (MLP) over ATM link fragmentation and interleaving (LFI) debug information, use the **debugatmlfi** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug atm lfi

no debug atm lfi

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.3(7)T	This command was introduced.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.

Examples

The following examples show output from the **debugatmlfi** command. Each example is preceded by an explanation of the output.

- The following output indicates that the packet has dequeued from the per-VC queue that is associated with the virtual circuit (VC):

```
00:17:27: MLP-ATM(Virtual-Access3) pak dequeued from per VC Q 15/200,qcount:0
```

- The following output indicates that the packet is enqueued on the per-VC queue associated with the VC:

```
00:17:27: MLP-ATM(Virtual-Access3) pak enqueued to per VC Q 15/200, qcount:0
```

- The following output indicates that the packet has dequeued from the MLP bundle queue:

```
00:17:27: MLP-ATM(Virtual-Access3) pak dequeued from MP Bundle 15/200, qcount:0
```

- The following output indicates that PPP over ATM (PPPoA) encapsulation cannot be added to the packet for some reason:

```
00:17:27: MLP-ATM(Virtual-Access3) encapsulation failure - dropping packet
```

- The following output indicates that the VC could not be found on the virtual access interface associated with the PPPoA session:

```
00:17:27: MLP-ATM(Virtual-Access3) No VC to transmit- dropping packet
```

- When a permanent virtual circuit (PVC) has been deleted, the following output indicates that MLP has been deconfigured successfully:

```
00:17:27: MLP-ATM(Virtual-Access3) mlp de-configured for PVC 15/200
```

- If the changing of any PVC parameters requires re-creation of the PVC, the following output is generated during the re-creation of the PVC:

```
00:17:27: MLP-ATM(Virtual-Access3) MLPoATM re-configured for PVC 15/200
```

- The following output indicates that the MLP over ATM structure associated with a VC has failed to allocate memory:

```
00:17:27: MLP-ATM(Virtual-Access3) Memory allocation error
```

- The following output is generated when MLP over ATM is first configured on a PVC:

```
00:17:27: MLP-ATM(Virtual-Access3) MLPoATM configured for PVC 15/200
```

Related Commands

Command	Description
show multilink ppp	Displays bundle information for MLP bundles.

debug atm native

To display ATM switched virtual circuit (SVC) signaling events, use the **debugatmnative** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug atm native {**[api]**| **[conn]**| **[error]**| **[filter]**}

no debug atm native

Syntax Description

api	(Optional) Native ATM application programming interface (API). Displays events that occur as a result of the exchange between the native ATM API and the signaling API.
conn	(Optional) Native ATM connection manager. Displays internal connection manager events for the native ATM API.
error	(Optional) Native ATM error. Displays errors that occur during the setup of an ATM SVC.
filter	(Optional) Native ATM filter. Displays the internal network service access point (NSAP) filter events of the native ATM API.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(8)T	This command was introduced.

Usage Guidelines

Native ATM API is the layer above the signaling API. Static map and Resource Reservation Protocol (RSVP) clients use the native ATM API to interact with the signaling API to create ATM SVCs.

Use the **debugatmnative** command to diagnose problems in the creation of static map and RSVP ATM SVCs.

Examples

The following is sample output for the **debugatmnative** command with the **api** keyword:

```
Router# debug atm native api
0:24:59:NATIVE ATM :associate endpoint
```

```
00:24:59:NATIVE ATM :ID (3) prep outgoing call, conn_type  0
00:24:59:NATIVE ATM :ID (3) set connection attribute for 5
00:24:59:NATIVE ATM :ID (3) query connection attribute 8
00:24:59:NATIVE ATM :ID (3) set connection attribute for 8
00:24:59:NATIVE ATM :ID (3) set connection attribute for 9
00:24:59:NATIVE ATM :ID (3) set connection attribute for 10
00:24:59:NATIVE ATM :ID (3) set connection attribute for 7
00:24:59:NATIVE ATM :ID (3) set connection attribute for 6
00:24:59:NATIVE ATM :ID (3) set connection attribute for 2
00:24:59:NATIVE ATM :ID (3) set connection attribute for 0
00:24:59:NATIVE ATM :ID (3) query connection attribute 12
00:24:59:NATIVE ATM :ID (3) set connection attribute for 12
00:24:59:NATIVE ATM :ID (3) query connection attribute 13
00:24:59:NATIVE ATM :ID (3) set connection attribute for 13
00:24:59:NATIVE ATM :ID (3) connect outgoing call
00:24:59:NATIVE ATM :ID (3) callback, CONNECT received
```

debug atm nbma

To display setup and teardown events for ATM switched virtual circuits (SVCs) configured using the Resource Reservation Protocol (RSVP), use the **debugatmnbma**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug atm nbma [api]

no debug atm nbma

Syntax Description

api	(Optional) Nonbroadcast multiaccess (NBMA) ATM application programming interface (API). Displays events that occur as a result of the exchange between RSVP and the NBMA API.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(8)T	This command was introduced.

Usage Guidelines

Use the **debugatmnbma** command to diagnose problems in the creation of RSVP SVCs.

The RSVP application creates SVCs by using the NBMA API. The **debugatmnbma** command with the**api**keyword displays events that occur as a result of the exchange between RSVP and the NBMA API.

Examples

The following is sample output for the **debugatmnbma** command:

```
Router# debug atm nbma api
00:52:50:NBMA-ATM-API - atm_setup_req
00:52:50:NBMA_ATM-API - nbma_atm_fill_blli
00:52:50:NBMA_ATM-API - nbma_atm_fill_bhli
00:52:50:NBMA_ATM-API - nbma_atm_callbackMsg - NATIVE_ATM_OUTGOING_CALL_ACTIVE
00:52:50:NBMA_ATM-API - rcv_outgoing_call_active
00:52:50:NBMA_ATM-API - nbma_svc_lookup
```

debug atm oam cc

To display ATM operation, administration, and maintenance (OAM) F5 continuity check (CC) management activity, use the **debugatmoamcc**command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug atm oam cc [**interface atm** *number*]

no debug atm oam cc [**interface atm** *number*]

Syntax Description

interface atm *number*	(Optional) Number of the ATM interface.

Command Default

No default behavior or values

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(2)XB	This command was introduced.
12.2(13)T	This command was integrated into Cisco IOS Release 12.2(13)T.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.

Examples

The following sample output for the **debugatmoamcc** command records activity beginning with the entry of the**oam-pvcmanagecc** command and ending with the entry of the**nooam-pvcmanagecc** command. The ATM 0 interface is specified, and the "both" segment direction is specified. The output shows an activation request sent and confirmed, a series of CC cells sent by the routers on each end of the segment, and a deactivation request and confirmation.

```
Router# debug atm oam cc interface atm0
Generic ATM:
  ATM OAM CC cells debugging is on
Router#
00:15:05: CC ACTIVATE MSG (ATM0) I:VCD#1 VC 1/40 OAM Cell Type:4 OAM
Type:8 OAM Func:1 Direction:3 CTag:5
00:15:05: CC ACTIVATE CONFIRM MSG (ATM0) O:VCD#1 VC 1/40 OAM Cell
Type:4 OAM Type:8 OAM Func:1 Direction:3 CTag:5
00:15:06: CC CELL (ATM0) O:VCD#1 VC 1/40 OAM Cell Type:4 OAM Type:1
00:15:07: CC CELL (ATM0) I:VCD#1 VC 1/40 OAM Cell Type:4 OAM Type:1 OAM Func:4
00:15:08: CC CELL (ATM0) O:VCD#1 VC 1/40 OAM Cell Type:4 OAM Type:1 OAM Func:4
00:15:09: CC CELL (ATM0) I:VCD#1 VC 1/40 OAM Cell Type:4 OAM Type:1 OAM Func:4
00:15:10: CC CELL (ATM0) O:VCD#1 VC 1/40 OAM Cell Type:4 OAM Type:1 OAM Func:4
00:15:11: CC CELL (ATM0) I:VCD#1 VC 1/40 OAM Cell Type:4 OAM Type:1 OAM Func:4
00:15:12: CC CELL (ATM0) O:VCD#1 VC 1/40 OAM Cell Type:4 OAM Type:1 OAM Func:4
00:15:13: CC CELL (ATM0) I:VCD#1 VC 1/40 OAM Cell Type:4 OAM Type:1 OAM Func:4
00:15:14: CC CELL (ATM0) O:VCD#1 VC 1/40 OAM Cell Type:4 OAM Type:1 OAM Func:4
```

```
00:15:15: CC CELL (ATM0) I:VCD#1 VC 1/40 OAM Cell Type:4 OAM Type:1 OAM Func:4
00:15:16: CC CELL (ATM0) O:VCD#1 VC 1/40 OAM Cell Type:4 OAM Type:1 OAM Func:4
00:15:17: CC CELL (ATM0) I:VCD#1 VC 1/40 OAM Cell Type:4 OAM Type:1 OAM Func:4
00:15:18: CC CELL (ATM0) O:VCD#1 VC 1/40 OAM Cell Type:4 OAM Type:1 OAM Func:4
00:15:19: CC CELL (ATM0) I:VCD#1 VC 1/40 OAM Cell Type:4 OAM Type:1 OAM Func:4
00:15:19: CC DEACTIVATE MSG (ATM0) I:VCD#1 VC 1/40 OAM Cell Type:4 OAM
Type:8 OAM Func:1 Direction:3 CTag:6
00:15:19: CC DEACTIVATE CONFIRM MSG (ATM0) O:VCD#1 VC 1/40 OAM Cell
Type:4 OAM Type:8 OAM Func:1 Direction:3 CTag:6
```

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

Table 27: debug atm oam cc Field Descriptions

Field	Description
00:15:05	Time stamp.
CC ACTIVATE MSG (ATM0)	Message type and interface.
0	Source.
1	Sink.
VC 1/40	Virtual circuit identifier.
Direction:3	Direction in which the cells are traveling. May be one of the following values: 1-- local router is the sink. 2-- local router is the source. 3-- both routers operate as the source and sink.

Related Commands

Command	Description
oam-pvc manage cc	Configures ATM OAM F5 CC management.
show atm pvc	Displays all ATM PVCs and traffic information.

debug atm oc3 pom

To display debug messages for ATM-OC3 Provisioning Object Manager (POM) network modules, use the **debugatmoc3pom** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug atm oc3 pom {**data**| **flow**| **pa**| **sar**| **sfp**| **trace**}

no debug atm oc3 pom {**data**| **flow**| **pa**| **sar**| **sfp**| **trace**}

Syntax Description

data	Displays debug messages for incoming packet indications.
flow	Displays debug messages for flow control indications.
pa	Displays debug messages for online insertion or removal (OIR) of the ATM-OC3 POM network module.
sar	Displays debug messages for blocking commands sent to the segmentation and reassembly (SAR) and their acknowledgments.
sfp	Displays debug messages for OIR of modules in the SFP port of the network module.
trace	Displays debug messages that give the hexadecimal representation of commands sent to the SAR and their acknowledgments.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.4(2)T	This command was introduced.

Usage Guidelines

debug atm oc3 pom data command

Use the **debugatmoc3pomdata** command to display the incoming packet indications. Each incoming packet transferred by direct memory access (DMA) to the host memory by the SAR will cause a packet indication.

debug atm oc3 pom flow command

Use the **debugatmoc3pomflow** command to display flow control indications.

When traffic sent to the SAR exceeds the peak cell rate for a particular virtual circuit (VC), the SAR indicates this to the host by sending flow control indications. These indications inform the host that either the high watermark or the low watermark has been reached for that VC queue.

When a high watermark is received from the SAR, indicating that the VC queue is full, the host will stop sending packets to the SAR until a low watermark indication is received. A low watermark indicates that the VC queue has been drained sufficiently to receive additional packets.

debug atm oc3 pom pa command

Use the **debugatmoc3pompa**command on those platforms supporting OIR to display the indications generated when the port adapter (the ATM-OC3 POM network module) is subjected to OIR. This command is used principally during the port adapter initialization phase.

debug atm oc3 pom sar command

Use the **debugatmoc3pomsar** command to display blocking commands or indications sent to or received from the SAR. This includes commands or indications of the creation or deletion of virtual circuits or virtual paths.

debug atm oc3 pom sfp command

Use the **debugatmoc3pomsfp** command to display the indications generated when a module in the SFP port is subjected to OIR.

debug atm oc3 pom trace command

Use the **debugatmoc3pomtrace** command to display the hexadecimal representation of commands sent to or received from the SAR. To facilitate debugging, use this command in conjunction with the **debugatmoc3pomsar** command.

Examples

Examples

The following is sample output from the **debugatmoc3pomdata** command:

```
Router# debug atm oc3 pom data
DATA debugging is on
Router#
*Jun 27 22:03:17.996: Packet Indication:
*Jun 27 22:03:17.996:   word 0: 0x00007D24
*Jun 27 22:03:17.996:   word 1: 0x00002F02
*Jun 27 22:03:17.996:   word 2: 0xEE323464
*Jun 27 22:03:17.996:   word 3: 0x006C006D
```

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

Table 28: debug atm oc3 pom data Field Descriptions

Field	Description
Jun 27 22:03:17.996:	Date or time stamp of packet DMA transfer.
word [0 - 3]: 0xXXXXXXXX	Hexadecimal representation of four-word acknowledgment from the SAR when a packet is transferred by DMA to the host memory by the SAR.

Examples

The following example illustrates the output from the **debugatmoc3pomflow** command:

```
Router# debug atm oc3 pom flow
FLOW CNTL INDICATION debugging is on
Router#
*Jun 27 15:14:13.123: Flow Indication:
*Jun 27 15:14:13.123:   word 0: 0x00000001
*Jun 27 15:14:13.123:   word 1: 0x300012C0
*Jun 27 15:14:13.123:   word 2: 0x18001060
*Jun 27 15:14:13.123:   word 3: 0x00080021
*Jun 27 15:14:13.456: Flow Indication:
*Jun 27 15:14:13.456:   word 0: 0x00000001
*Jun 27 15:14:13.456:   word 1: 0x300012C0
*Jun 27 15:14:13.456:   word 2: 0x18001060
*Jun 27 15:14:13.456:   word 3: 0x00090022
```

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

Table 29: debug atm oc3 pom flow Field Descriptions

Field	Description
Jun 27 15:14:13.456:	Date or time stamp of flow indication
word [0 - 3]: 0xXXXXXXXX	Hexadecimal representation of four-word indication sent by the SAR to the host that a high watermark or low watermark event has occurred.
word 3: 0x00XXYYYY	When XX is 08, a high watermark has been received by the host. The host will stop queueing packets for the VC. When XX is 09, a low watermark has been received by the host. The host will resume sending packets to the VC. YYYY is the running count of flow indication events sent to the host.

Examples

The following examples illustrate the output from the **debugatmoc3pompa** command.

The first example gives the output when the network module is removed:

```
Router# debug atm oc3 pom pa
PA debugging is on
*Jun 27 22:40:56.110: %OIR-6-REMCARD: Card removed from slot 2, interfaces disabled
*Jun 27 22:40:56.122: *** Freed 6146 buffers
```

The second example gives the output when the network module is inserted, and gives the values of internal registers of the module:

```
*Jun 27 22:41:08.654: %OIR-6-INSCARD: Card inserted in slot 2, interfaces administratively
 shut down
*Jun 27 22:41:11.402: sar_base_addr 0x5C800000
*Jun 27 22:41:11.402: PCI_MEMBAR2_REG after configuring:0x5E000008
*Jun 27 22:41:11.402: PCI_MEMBAR3_REG after configuring:0x5F000000
*Jun 27 22:41:11.402: PCI_COMMAND_REG: Offset= 0x4; value= 0x2A00006
```

```
*Jun 27 22:41:11.402: FPGA Base address is 0x5C900000
*Jun 27 22:41:11.402: FPGA PCI config Reg is 0x02200002
```

Examples

The following examples illustrate the output from the **debugatmoc3pomsar** command.

The first example displays command indications for setting up a VC and opening the reassembly channel and the segmentation channel in the SAR:

```
Router# debug atm oc3 pom sar
SAR debugging is on
Router# configure terminal
Router(config)# interface atm 2/0
Router(config-if)# pvc 2/2
Router(config-if-atm-vc)# exit
Router(config-if)#
*Jun 27 22:12:28.816: ATM2/0: Setup_VC: vc:3 vpi:2 vci:2
*Jun 27 22:12:28.816: ATM2/0: Open_Channel(RSY): CH (1), VPI (2), VCI (2)
*Jun 27 22:12:28.816: ATM2/0: HI/LO watermarks: 526/263; PeakRate: 149760
*Jun 27 22:12:28.816: ATM2/0: Open_Channel(SEG): CH (1), VPI (2), VCI (2)
*Jun 27 22:12:28.820: ATM2/0: Setup_Cos: vc:3 wred_name:- max_q:0
```

The second example displays the commands sent to the SAR and the acknowledgements returned when the VC is deleted and the segmentation and reassembly channels are closed:

```
Router(config-if)# no pvc 2/2
Router(config-if)#
*Jun 27 22:12:59.016: ATM2/0: Sent pending EOP successfully
*Jun 27 22:12:59.016: ATM2/0: Close_Channel(RSY): Chan_ID (0x104)
*Jun 27 22:12:59.016: ATM2/0: Close_Channel(RSY): Chan_ID (0x104) CLOSE
*Jun 27 22:12:59.016: ATM2/0: Close_Channel: CLOSE_PENDING
*Jun 27 22:12:59.016: ATM2/0: Close_Channel(SEG): Chan_ID (0x105)
*Jun 27 22:12:59.016: ATM2/0: Close_Channel: CLOSE
```

Examples

The following examples illustrate the output from the **debugatmoc3pomsfp** command.

The first example gives the output when the module is removed from the SFP port:

```
Router# debug atm oc3 pom sfp
SFP debugging is on
*Jun 27 22:27:40.792: SFP TX FAULT detected
*Jun 27 22:27:40.808: SFP LOS detected
*Jun 27 22:27:40.812: SFP removal detected
*Jun 27 22:27:41.464: NM-1A-OC3-POM: SFP 2/0 - Removed unique
*Jun 27 22:27:43.464: %LINK-3-UPDOWN: Interface ATM2/0, changed state to down
*Jun 27 22:27:44.464: %LINEPROTO-5-UPDOWN: Line protocol on Interface ATM2/0, changed state
 to down
```

The second example gives the output when the module is inserted in the SFP port.

```
*Jun 27 22:27:47.776: SFP LOS cleared
*Jun 27 22:27:47.776: SFP TX FAULT detected
*Jun 27 22:27:48.276: SFP present detected
*Jun 27 22:27:48.276: SFP TX FAULT cleared
*Jun 27 22:27:48.496:  Set the Container_id to 17
*Jun 27 22:27:50.496: %LINK-3-UPDOWN: Interface ATM2/0, changed state to up
*Jun 27 22:27:51.496: %LINEPROTO-5-UPDOWN: Line protocol on Interface ATM2/0, changed state
 to up
```

Examples

The first example illustrates the output from the **debugatmoc3pomtrace** command when it is run without the **debugatmoc3sar** command being activated:

```
Router# debug atm oc3 pom trace
SAR CMD/ACK debugging is on
```

```
Router# configure terminal
Router(config)# interface atm 2/0
router(config-if)# pvc 2/2
Router(config-if-atm-vc)# exit
Router(config-if)#
*Jun 27 22:15:09.284: Command Sent:
*Jun 27 22:15:09.284:   word 0: 0x00000480
*Jun 27 22:15:09.284:   word 1: 0x00012010
*Jun 27 22:15:09.284:   word 2: 0x00000000
*Jun 27 22:15:09.284:   word 3: 0x00000000
*Jun 27 22:15:09.284:   word 4: 0x00200020
*Jun 27 22:15:09.284:   word 5: 0x00000000
*Jun 27 22:15:09.284:   word 6: 0x00000000
*Jun 27 22:15:09.284:   word 7: 0x00000000
*Jun 27 22:15:09.284:   word 8: 0x00000000
*Jun 27 22:15:09.284: Command Indication:
*Jun 27 22:15:09.284:   word 0: 0x00000000
*Jun 27 22:15:09.284:   word 1: 0x01042110
*Jun 27 22:15:09.284:   word 2: 0x01050000
*Jun 27 22:15:09.284:   word 3: 0x0000003B
*Jun 27 22:15:09.284: ACK received = 200 usecs
*Jun 27 22:15:09.284: Command Sent:
*Jun 27 22:15:09.284:   word 0: 0x01050480
*Jun 27 22:15:09.284:   word 1: 0x00011010
*Jun 27 22:15:09.284:   word 2: 0x02000000
*Jun 27 22:15:09.284:   word 3: 0x00010003
*Jun 27 22:15:09.284:   word 4: 0x00200020
*Jun 27 22:15:09.284:   word 5: 0x64B30000
*Jun 27 22:15:09.284:   word 6: 0x10C00000
*Jun 27 22:15:09.284:   word 7: 0x86850000
*Jun 27 22:15:09.284:   word 8: 0x00010040
*Jun 27 22:15:09.284:   word 9: 0x00000000
*Jun 27 22:15:09.284: Command Indication:
*Jun 27 22:15:09.284:   word 0: 0x00010000
*Jun 27 22:15:09.284:   word 1: 0x00011110
*Jun 27 22:15:09.284:   word 2: 0x02000000
*Jun 27 22:15:09.284:   word 3: 0x0001003D
*Jun 27 22:15:09.284: ACK received = 200 usecs
```

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

Table 30: debug atm oc3 pom trace Field Descriptions

Field	Description
Jun 27 22:15:09.284:	Date or time stamp for the command dialog.
word [0 - n]: 0xXXXXXXXX	Hexadecimal representation of the **n-word** command sent to the SAR (under Command Sent:) and the four-word acknowledgment returned by the SAR (under Command Indication:).
ACK received	Time (in microseconds) between sending the command to the SAR and receiving the acknowledgment.

The second example illustrates the output from the **debugatmoc3pomtrace** command run in conjunction with the **debugatmoc3pomsar** command.

In this example, each command sent to the SAR is displayed by the **debugatmoc3pomsar** command. Then the hexadecimal representation of the command and its acknowledgement are displayed by the **debugatmoc3pomtrace** command.

```
Router# debug atm oc3 pom trace
SAR CMD/ACK debugging is on
Router# debug atm oc3 pom sar
SAR debugging is on
Router# configure terminal
Router(config)# interface atm 2/0
router(config-if)# pvc 2/2
Router(config-if-atm-vc)# exit
Router(config-if)#
*Jun 27 22:15:09.284: ATM2/0: Setup_VC: vc:4 vpi:2 vci:2
*Jun 27 22:15:09.284: ATM2/0: Open_Channel(RSY): CH (1), VPI (2), VCI (2)
*Jun 27 22:15:09.284: Command Sent:
*Jun 27 22:15:09.284:   word 0: 0x00000480
*Jun 27 22:15:09.284:   word 1: 0x00012010
*Jun 27 22:15:09.284:   word 2: 0x00000000
*Jun 27 22:15:09.284:   word 3: 0x00000000
*Jun 27 22:15:09.284:   word 4: 0x00200020
*Jun 27 22:15:09.284:   word 5: 0x00000000
*Jun 27 22:15:09.284:   word 6: 0x00000000
*Jun 27 22:15:09.284:   word 7: 0x00000000
*Jun 27 22:15:09.284:   word 8: 0x00000000
*Jun 27 22:15:09.284: Command Indication:
*Jun 27 22:15:09.284:   word 0: 0x00000000
*Jun 27 22:15:09.284:   word 1: 0x01042110
*Jun 27 22:15:09.284:   word 2: 0x01050000
*Jun 27 22:15:09.284:   word 3: 0x0000003B
*Jun 27 22:15:09.284: ACK received = 200 usecs
*Jun 27 22:15:09.284: ATM2/0: HI/LO watermarks: 526/263; PeakRate: 149760
*Jun 27 22:15:09.284: ATM2/0: Open_Channel(SEG): CH (1), VPI (2), VCI (2)
*Jun 27 22:15:09.284: Command Sent:
*Jun 27 22:15:09.284:   word 0: 0x01050480
*Jun 27 22:15:09.284:   word 1: 0x00011010
*Jun 27 22:15:09.284:   word 2: 0x02000000
*Jun 27 22:15:09.284:   word 3: 0x00010003
*Jun 27 22:15:09.284:   word 4: 0x00200020
*Jun 27 22:15:09.284:   word 5: 0x64B30000
*Jun 27 22:15:09.284:   word 6: 0x10C00000
*Jun 27 22:15:09.284:   word 7: 0x86850000
*Jun 27 22:15:09.284:   word 8: 0x00010040
*Jun 27 22:15:09.284:   word 9: 0x00000000
*Jun 27 22:15:09.284: Command Indication:
*Jun 27 22:15:09.284:   word 0: 0x00010000
*Jun 27 22:15:09.284:   word 1: 0x00011110
*Jun 27 22:15:09.284:   word 2: 0x02000000
*Jun 27 22:15:09.284:   word 3: 0x0001003D
*Jun 27 22:15:09.284: ACK received = 200 usecs
*Jun 27 22:15:09.284: ATM2/0: Setup_Cos: vc:4 wred_name:- max_q:0
```

debug atm t3e3

To display debug messages for ATM T3/E3 network modules, use the **debugatmt3e3** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug atm t3e3 {**data**| **flow**| **pa**| **sar**| **trace**}

no debug atm t3e3 {**data**| **flow**| **pa**| **sar**| **trace**}

Syntax Description

data	Displays debug messages for incoming packet indications.
flow	Displays debug messages for flow control indications.
pa	Displays debug messages for online insertion or removal (OIR) of the ATM T3/E3 network module.
sar	Displays debug messages for blocking commands sent to the segmentation and reassembly (SAR) and their acknowledgments.
trace	Displays debug messages that give the hexadecimal representation of commands sent to the SAR and their acknowledgments.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.4(15)T	This command was introduced.

Usage Guidelines

debug atm t3e3 data command

Use the **debugatmt3e3data** command to display the incoming packet indications. Each incoming packet transferred by direct memory access (DMA) to the host memory by the SAR will cause a packet indication.

debug atm t3e3 flow command

Use the **debugatmt3e3flow** command to display flow control indications.

When traffic sent to the SAR exceeds the peak cell rate for a particular virtual circuit (VC), the SAR indicates this to the host by sending flow control indications. These indications inform the host that either the high watermark or the low watermark has been reached for that VC queue.

When a high watermark is received from the SAR, indicating that the VC queue is full, the host will stop sending packets to the SAR until a low watermark indication is received. A low watermark indicates that the VC queue has been drained sufficiently to receive additional packets.

debug atm t3e3 pa command

Use the **debugatmt3e3pa**command on those platforms supporting OIR to display the indications generated when the port adapter (the ATM T3/E3 network module) is subjected to OIR. This command is used principally during the port adapter initialization phase.

debug atm t3e3 sar command

Use the **debugatmt3e3sar** command to display blocking commands or indications sent to or received from the SAR. This includes commands or indications of the creation or deletion of virtual circuits or virtual paths.

debug atm t3e3 trace command

Use the **debugatmt3e3trace** command to display the hexadecimal representation of commands sent to or received from the SAR. To facilitate debugging, use this command in conjunction with the **debugatmt3e3sar** command.

Examples

Examples

The following is sample output from the **debugatmt3e3data** command:

```
Router# debug atm t3e3 data
DATA debugging is on
Router#
*Jun 27 22:03:17.996: Packet Indication:
*Jun 27 22:03:17.996:   word 0: 0x00007D24
*Jun 27 22:03:17.996:   word 1: 0x00002F02
*Jun 27 22:03:17.996:   word 2: 0xEE323464
*Jun 27 22:03:17.996:   word 3: 0x006C006D
```

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

Table 31: debug atm t3e3 data Field Descriptions

Field	Description
Jun 27 22:03:17.996:	Date or time stamp of packet DMA transfer.
word [0 - 3]: 0xXXXXXXXX	Hexadecimal representation of four-word acknowledgment from the SAR when a packet is transferred by DMA to the host memory by the SAR.

Examples

The following example illustrates the output from the **debugatmt3e3flow** command:

```
Router# debug atm t3e3 flow
FLOW CNTL INDICATION debugging is on
Router#
*Jun 27 15:14:13.123: Flow Indication:
*Jun 27 15:14:13.123:   word 0: 0x00000001
*Jun 27 15:14:13.123:   word 1: 0x300012C0
*Jun 27 15:14:13.123:   word 2: 0x18001060
*Jun 27 15:14:13.123:   word 3: 0x00080021
*Jun 27 15:14:13.456: Flow Indication:
*Jun 27 15:14:13.456:   word 0: 0x00000001
```

```
*Jun 27 15:14:13.456:   word 1: 0x300012C0
*Jun 27 15:14:13.456:   word 2: 0x18001060
*Jun 27 15:14:13.456:   word 3: 0x00090022
```

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

Table 32: debug atm t3e3 flow Field Descriptions

Field	Description
Jun 27 15:14:13.456:	Date or time stamp of flow indication
word [0 - 3]: 0xXXXXXXXX	Hexadecimal representation of four-word indication sent by the SAR to the host that a high watermark or low watermark event has occurred.
word 3: 0x00XXYYYY	When XX is 08, a high watermark has been received by the host. The host will stop queueing packets for the VC. When XX is 09, a low watermark has been received by the host. The host will resume sending packets to the VC. YYYY is the running count of flow indication events sent to the host.

Examples

The following examples illustrate the output from the **debugatmt3e3pa** command.

The first example gives the output when the network module is removed:

```
Router# debug atm t3e3 pa
PA debugging is on
*Jun 27 22:40:56.110: %OIR-6-REMCARD: Card removed from slot 2, interfaces disabled
*Jun 27 22:40:56.122: *** Freed 6146 buffers
```

The second example gives the output when the network module is inserted, and gives the values of internal registers of the module:

```
*Jun 27 22:41:08.654: %OIR-6-INSCARD: Card inserted in slot 2, interfaces administratively
 shut down
*Jun 27 22:41:11.402: sar_base_addr 0x5C800000
*Jun 27 22:41:11.402: PCI_MEMBAR2_REG after configuring:0x5E000008
*Jun 27 22:41:11.402: PCI_MEMBAR3_REG after configuring:0x5F000000
*Jun 27 22:41:11.402: PCI_COMMAND_REG: Offset= 0x4; value= 0x2A00006
*Jun 27 22:41:11.402: FPGA Base address is 0x5C900000
*Jun 27 22:41:11.402: FPGA PCI config Reg is 0x02200002
```

Examples

The following examples illustrate the output from the **debugatmt3e3sar** command.

The first example displays command indications for setting up a VC and opening the reassembly channel and the segmentation channel in the SAR:

```
Router# debug atm t3e3 sar
SAR debugging is on
Router# configure terminal
Router(config)# interface atm 2/0
Router(config-if)# pvc 2/2
Router(config-if-atm-vc)# exit
Router(config-if)#
*Jun 27 22:12:28.816: ATM2/0: Setup_VC: vc:3 vpi:2 vci:2
*Jun 27 22:12:28.816: ATM2/0: Open_Channel(RSY): CH (1), VPI (2), VCI (2)
*Jun 27 22:12:28.816: ATM2/0: HI/LO watermarks: 526/263; PeakRate: 149760
*Jun 27 22:12:28.816: ATM2/0: Open_Channel(SEG): CH (1), VPI (2), VCI (2)
*Jun 27 22:12:28.820: ATM2/0: Setup_Cos: vc:3 wred_name:- max_q:0
```

The second example displays the commands sent to the SAR and the acknowledgements returned when the VC is deleted and the segmentation and reassembly channels are closed:

```
Router(config-if)# no pvc 2/2
Router(config-if)#
*Jun 27 22:12:59.016: ATM2/0: Sent pending EOP successfully
*Jun 27 22:12:59.016: ATM2/0: Close_Channel(RSY): Chan_ID (0x104)
*Jun 27 22:12:59.016: ATM2/0: Close_Channel(RSY): Chan_ID (0x104) CLOSE
*Jun 27 22:12:59.016: ATM2/0: Close_Channel: CLOSE_PENDING
*Jun 27 22:12:59.016: ATM2/0: Close_Channel(SEG): Chan_ID (0x105)
*Jun 27 22:12:59.016: ATM2/0: Close_Channel: CLOSE
```

Examples

The first example illustrates the output from the **debugatmt3e3trace** command when it is run without the **debugatmt3e3sar** command being activated:

```
Router# debug atm t3e3 trace
SAR CMD/ACK debugging is on
Router# configure terminal
Router(config)# interface atm 2/0
router(config-if)# pvc 2/2
Router(config-if-atm-vc)# exit
Router(config-if)#
*Jun 27 22:15:09.284: Command Sent:
*Jun 27 22:15:09.284:   word 0: 0x00000480
*Jun 27 22:15:09.284:   word 1: 0x00012010
*Jun 27 22:15:09.284:   word 2: 0x00000000
*Jun 27 22:15:09.284:   word 3: 0x00000000
*Jun 27 22:15:09.284:   word 4: 0x00200020
*Jun 27 22:15:09.284:   word 5: 0x00000000
*Jun 27 22:15:09.284:   word 6: 0x00000000
*Jun 27 22:15:09.284:   word 7: 0x00000000
*Jun 27 22:15:09.284:   word 8: 0x00000000
*Jun 27 22:15:09.284: Command Indication:
*Jun 27 22:15:09.284:   word 0: 0x00000000
*Jun 27 22:15:09.284:   word 1: 0x01042110
*Jun 27 22:15:09.284:   word 2: 0x01050000
*Jun 27 22:15:09.284:   word 3: 0x0000003B
*Jun 27 22:15:09.284: ACK received = 200 usecs
*Jun 27 22:15:09.284: Command Sent:
*Jun 27 22:15:09.284:   word 0: 0x01050480
*Jun 27 22:15:09.284:   word 1: 0x00011010
*Jun 27 22:15:09.284:   word 2: 0x02000000
*Jun 27 22:15:09.284:   word 3: 0x00010003
*Jun 27 22:15:09.284:   word 4: 0x00200020
*Jun 27 22:15:09.284:   word 5: 0x64B30000
*Jun 27 22:15:09.284:   word 6: 0x10C00000
*Jun 27 22:15:09.284:   word 7: 0x86850000
*Jun 27 22:15:09.284:   word 8: 0x00010040
*Jun 27 22:15:09.284:   word 9: 0x00000000
*Jun 27 22:15:09.284: Command Indication:
*Jun 27 22:15:09.284:   word 0: 0x00010000
*Jun 27 22:15:09.284:   word 1: 0x00011110
*Jun 27 22:15:09.284:   word 2: 0x02000000
*Jun 27 22:15:09.284:   word 3: 0x0001003D
*Jun 27 22:15:09.284: ACK received = 200 usecs
```

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

Table 33: debug atm t3e3 trace Field Descriptions

Field	Description
Jun 27 22:15:09.284:	Date or time stamp for the command dialog.
word [0 - n]: 0xXXXXXXXX	Hexadecimal representation of the **n-word** command sent to the SAR (under Command Sent:) and the four-word acknowledgment returned by the SAR (under Command Indication:).
ACK received	Time (in microseconds) between sending the command to the SAR and receiving the acknowledgment.

The second example illustrates the output from the **debugatmt3e3trace** command run in conjunction with the **debugatmt3e3sar** command.

In this example, each command sent to the SAR is displayed by the **debugatmt3e3sar** command. Then the hexadecimal representation of the command and its acknowledgement are displayed by the **debugatmt3e3trace** command.

```
Router# debug atm t3e3 trace
SAR CMD/ACK debugging is on
Router# debug atm t3e3 sar
SAR debugging is on
Router# configure terminal
Router(config)# interface atm 2/0
router(config-if)# pvc 2/2
Router(config-if-atm-vc)# exit
Router(config-if)#
*Jun 27 22:15:09.284: ATM2/0: Setup_VC: vc:4 vpi:2 vci:2
*Jun 27 22:15:09.284: ATM2/0: Open_Channel(RSY): CH (1), VPI (2), VCI (2)
*Jun 27 22:15:09.284: Command Sent:
*Jun 27 22:15:09.284:   word 0: 0x00000480
*Jun 27 22:15:09.284:   word 1: 0x00012010
*Jun 27 22:15:09.284:   word 2: 0x00000000
*Jun 27 22:15:09.284:   word 3: 0x00000000
*Jun 27 22:15:09.284:   word 4: 0x00200020
*Jun 27 22:15:09.284:   word 5: 0x00000000
*Jun 27 22:15:09.284:   word 6: 0x00000000
*Jun 27 22:15:09.284:   word 7: 0x00000000
*Jun 27 22:15:09.284:   word 8: 0x00000000
*Jun 27 22:15:09.284: Command Indication:
*Jun 27 22:15:09.284:   word 0: 0x00000000
*Jun 27 22:15:09.284:   word 1: 0x01042110
*Jun 27 22:15:09.284:   word 2: 0x01050000
*Jun 27 22:15:09.284:   word 3: 0x0000003B
*Jun 27 22:15:09.284: ACK received = 200 usecs
*Jun 27 22:15:09.284: ATM2/0: HI/LO watermarks: 526/263; PeakRate: 149760
*Jun 27 22:15:09.284: ATM2/0: Open_Channel(SEG): CH (1), VPI (2), VCI (2)
*Jun 27 22:15:09.284: Command Sent:
*Jun 27 22:15:09.284:   word 0: 0x01050480
*Jun 27 22:15:09.284:   word 1: 0x00011010
*Jun 27 22:15:09.284:   word 2: 0x02000000
*Jun 27 22:15:09.284:   word 3: 0x00010003
*Jun 27 22:15:09.284:   word 4: 0x00200020
*Jun 27 22:15:09.284:   word 5: 0x64B30000
*Jun 27 22:15:09.284:   word 6: 0x10C00000
*Jun 27 22:15:09.284:   word 7: 0x86850000
*Jun 27 22:15:09.284:   word 8: 0x00010040
*Jun 27 22:15:09.284:   word 9: 0x00000000
```

```
*Jun 27 22:15:09.284: Command Indication:
*Jun 27 22:15:09.284:   word 0: 0x00010000
*Jun 27 22:15:09.284:   word 1: 0x00011110
*Jun 27 22:15:09.284:   word 2: 0x02000000
*Jun 27 22:15:09.284:   word 3: 0x0001003D
*Jun 27 22:15:09.284: ACK received = 200 usecs
*Jun 27 22:15:09.284: ATM2/0: Setup_Cos: vc:4 wred_name:- max_q:0
```

debug audit

To display debug messages for the audit subsystem, use the **debugaudit**command in privileged EXEC mode. To disable debugging for the audit subsystem, use the **no** form of this command.

debug audit

no debug audit

Syntax Description

This command has no arguments or keywords.

Command Modes

Privileged EXEC

Command History

Release	Modification
12.2(18)S	This command was introduced.
12.0(27)S	This feature was integrated into Cisco IOS Release 12.0(27)S.
12.2(27)SBC	This command was integrated into Cisco IOS Release 12.2(27)SBC.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.

Usage Guidelines

Audit files allow you to track changes that have been made to your router. Each change is logged as a syslog message, and all syslog messages are kept in the audit file, which is kept in the audit subsystem.

Examples

The following example is sample output from the **debugaudit**command:

```
Router# debug audit

*Sep 14 18:37:31.535:disk0:/forensics.log -> File not found
*Sep 14 18:37:31.535:%AUDIT-1-RUN_VERSION:Hash:
24D98B13B87D106E7E6A7E5D1B3CE0AD User:
*Sep 14 18:37:31.583:%AUDIT-1-RUN_CONFIG:Hash:
4AC2D776AA6FCA8FD7653CEB8969B695 User:
*Sep 14 18:37:31.587:Audit:Trying to hash nvram:startup-config
*Sep 14 18:37:31.587:Audit:nvram:startup-config Done.
*Sep 14 18:37:31.587:Audit:Trying to hash nvram:private-config
*Sep 14 18:37:31.591:Audit:nvram:private-config Done.
*Sep 14 18:37:31.591:Audit:Trying to hash nvram:underlying-config
*Sep 14 18:37:31.591:Audit:nvram:underlying-config Done.
*Sep 14 18:37:31.591:Audit:Trying to hash nvram:persistent-data
*Sep 14 18:37:31.591:Audit:nvram:persistent-data Done.
*Sep 14 18:37:31.595:Audit:Trying to hash nvram:ifIndex-table
*Sep 14 18:37:31.595:Audit:Skipping nvram:ifIndex-table
*Sep 14 18:37:31.595:%AUDIT-1-STARTUP_CONFIG:Hash:
95DD497B1BB61AB33A629124CBFEC0FC User:
*Sep 14 18:37:31.595:Audit:Trying to hash filesystem disk0:
*Sep 14 18:37:31.775:Audit:Trying to hash attributes of
disk0:c7200-p-mz.120-23.S
*Sep 14 18:37:32.103:Audit:disk0:c7200-p-mz.120-23.S DONE
```

```
*Sep 14 18:37:32.103:Audit:disk0:DONE
*Sep 14 18:37:32.103:Audit:Trying to hash filesystem bootflash:
*Sep 14 18:37:32.103:Audit:Trying to hash attributes of
bootflash:c7200-kboot-mz.121-8a.E
*Sep 14 18:37:32.107:Audit:bootflash:c7200-kboot-mz.121-8a.E DONE
*Sep 14 18:37:32.107:Audit:Trying to hash attributes of
bootflash:crashinfo_20030115-182547
*Sep 14 18:37:32.107:Audit:bootflash:crashinfo_20030115-182547 DONE
*Sep 14 18:37:32.107:Audit:Trying to hash attributes of
bootflash:crashinfo_20030115-212157
*Sep 14 18:37:32.107:Audit:bootflash:crashinfo_20030115-212157 DONE
*Sep 14 18:37:32.107:Audit:Trying to hash attributes of
bootflash:crashinfo_20030603-155534
*Sep 14 18:37:32.107:Audit:bootflash:crashinfo_20030603-155534 DONE
*Sep 14 18:37:32.107:Audit:bootflash:DONE
*Sep 14 18:37:32.107:%AUDIT-1-FILESYSTEM:Hash:
330E7111F2B526F0B850C24ED5774EDE User:
*Sep 14 18:37:32.107:Audit:Hashing entitymib entry for 7206VXR chassis,
Hw Serial#:28710795, Hw Revision:A
*Sep 14 18:37:32.107:Audit:Hashing entitymib entry for NPE 400 Card, Hw
Serial#:28710795, Hw Revision:A
*Sep 14 18:37:32.107:Audit:Hashing entitymib entry for Chassis Slot
*Sep 14 18:37:32.107:Audit:Hashing entitymib entry for I/O Dual
FastEthernet Controller
*Sep 14 18:37:32.107:Audit:Hashing entitymib entry for i82543
(Livengood)
*Sep 14 18:37:32.107:Audit:Hashing entitymib entry for i82543
(Livengood)
*Sep 14 18:37:32.107:Audit:Hashing entitymib entry for Chassis Slot
*Sep 14 18:37:32.107:Audit:Hashing entitymib entry for Chassis Slot
*Sep 14 18:37:32.107:Audit:Hashing entitymib entry for Chassis Slot
*Sep 14 18:37:32.107:Audit:Hashing entitymib entry for Chassis Slot
*Sep 14 18:37:32.107:Audit:Hashing entitymib entry for Chassis Slot
*Sep 14 18:37:32.107:Audit:Hashing entitymib entry for Chassis Slot
*Sep 14 18:37:32.107:%AUDIT-1-HARDWARE_CONFIG:Hash:
32F66463DDA802CC9171AF6386663D20 User:
```

Related Commands

Command	Description
audit filesize	Changes the size of the audit file.
audit interval	Changes the time interval that is used for calculating hashes.

debug authentication

To display debugging information about the Authentication Manager, use the **debugauthentication** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug authentication {[**feature** *feature-name*]| {**all**| **detail**| **errors**| **events**| **sync**}}

no debug authentication {[**feature** *feature-name*]| {**all**| **detail**| **errors**| **events**| **sync**}}

Syntax Description

feature *feature-name*	Displays debugging information about specific features. To display the valid feature names, use the question mark (**?**) online help function.
all	Displays all debugging information about the Authentication Manager and all features.
detail	Displays detailed debugging information.
errors	Displays debugging information about Authentication Manager errors.
events	Displays debugging information about Authentication Manager events.
sync	Displays debugging information about Authentication Manager stateful switchovers (SSOs) or In Service Software Upgrades (ISSUs).

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.2(33)SXI	This command was introduced.
15.2(2)T	This command was integrated into Cisco IOS Release 15.2(2)T.
Cisco IOS XE Release 3.2SE	This command was modified. The **detail** keyword was added.

Usage Guidelines

Use the **debug authentication** command to troubleshoot the Authentication Manager.

Examples

The following example shows sample output from the **debug authentication** command when the **feature** and **events** keywords are configured:

```
Device# debug authentication feature mda events
Auth Manager mda events debugging is on
```

Related Commands

Command	Description
debug access-session	Displays debugging information about Session Aware Networking sessions.
debug dot1x	Displays 802.1x debugging information.

debug auto-config

To enable debugging for autoconfiguration applications, use the **debugauto-config** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug auto-config {**all**| **errors**| **events**| **parser**}

no debug auto-config {**all**| **errors**| **events**| **parser**}

Syntax Description

all	Displays all autoconfiguration debug trace.
errors	Displays autoconfiguration errors.
events	Displays autoconfiguration events.
parser	Displays autoconfiguration parser.

Command Default

Disabled

Command Modes

Privileged EXEC

Command History

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

Examples

The following example shows the **debugauto-config** command used to enable debugging for autoconfiguration applications and to display autoconfiguration events:

```
Router# debug auto-config events
.
.
.
Feb  8 02:17:31.119: dnld_app_check_state(0x628C8164)...
Feb  8 02:17:31.123: dnld_chk_app_handle(0x628C8164)
Feb  8 02:17:31.123: dnld_app_check_state: appl = 0x628C8164, state = 0x11
.
.
.
```

The table below describes significant fields shown in the display.

Table 34: debug auto-config Field Descriptions

Field	Description
0x628C8164	Identifies the application handle, an auto-generated number for debugging.
0x11	Displays the state of the application. State values are as follows: 0x11--Registered and enabled. 0x1--Download application enabled. 0x10--Download application registered.

Related Commands

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

debug autoupgrade

To display the debug output of the Cisco Auto-Upgrade Manager (AUM), use the **debug autoupgrade** command in privileged EXEC mode. To disable debug output, use the **no** form of this command.

debug autoupgrade

no debug autoupgrade

Syntax Description

This command has no arguments or keywords.

Command Default

The debug output is disabled.

Command Modes

Privileged EXEC (#)

Command History

Release	Modification
12.4(15)T	This command was introduced.
Cisco IOS XE Release 3.9S	This command was integrated into Cisco IOS XE Release 3.9S.

Usage Guidelines

Use the **debug autoupgrade** command when you encounter a problem with AUM and provide the output to TAC. Run the **debug autoupgrade** command and then run AUM to view the debug messages.

Examples

The following example shows how to enable the debugging of Cisco Auto-Upgrade Manager:

```
Device# debug autoupgrade
Auto Upgrade Manager debugging ON
Device#
Device# upgrade automatic getversion tftp://10.1.0.1/username/aaa

Image not found.
Device#
Jun 14 14:23:08.251 IST: AUM: Currently running software:
flash:c3825-adventerprisek9-mz.CALVIN_AUM_EFT1

Jun 14 14:23:08.251 IST: AUM: Reload type:2 hour:0 min:0

Jun 14 14:23:08.251 IST: AUM: Disk management: 1
Jun 14 14:23:08.251 IST: AUM: Get image tftp://10.1.0.1/username/aaa from local server and
 upgrade:
Jun 14 14:23:08.251 IST: AUM: Extracted image name: aaa
Jun 14 14:23:08.339 IST: AUM: get image info: failed to open url
Jun 14 14:23:08.339 IST: AUM: get image info: image size unknown
```

Related Commands

Command	Description
upgrade automatic getversion	Downloads a Cisco software image directly from www.cisco.com or from a non-Cisco server.