Troubleshooting

This chapter describes some common problems and how to troubleshoot them and contains the following sections:

- Using Debug Commands, page 5-1
- Common Problems, page 5-5
- Accessing the ISR-WAAS Application, page 5-11
- Example of ISR-WAAS Running Configuration, page 5-11

Using Debug Commands

- AppNav-XE Debug Commands, page 5-1
- AppNav Service Node Auto Discovery Debug Commands, page 5-4
- Container Debug Commands, page 5-5
- EZConfig Debug Commands, page 5-5

AppNav-XE Debug Commands

- Clearing AppNav-XE Statistics, page 5-1
- Debugging the Cisco IOS-XE Control Plane, page 5-2
- Debugging the Cisco IOS-XE Infrastructure, page 5-2
- Debugging the Data Plane, page 5-4

Clearing AppNav-XE Statistics

To clear all the AppNav-XE statistics or just certain statistics, use the following command:

```
router# clear service-insertion statistics ?

all                         Clear all service-insertion statistics
appnav-controller           Clear appnav-controller statistics
appnav-controller-group     Clear appnav-controller-group statistics
service-context             Clear service-context statistics
service-node                Clear service-node statistics
service-node-group          Clear service-node-group statistics
```

Debugging the Cisco IOS-XE Control Plane

Use the following debug commands to trace control plane activities:

```
router# debug appnav-controller ?
   auto-discovery Debugging AppNav Controller service node auto discovery feature
   cm-reg         Debugging AppNav Controller CM registration with the WCM server
   cmm            Debugging AppNav Controller Cluster Management (CMM)
   fdm            Debugging AppNav Controller Flow Distribution Module (FDM)
   ha             Enable AppNav Controller high availability (HA) redundancy
                  checkpoint and ISSU infrastructure debugs
   vi             Debugging AppNav Controller Virtual Interface (VI), including the
                  status at the time of creation and links to the compress and
                  uncompress interface

router# debug appnav-controller cmm ?
  all      Enable all CMM debugs
  cli      Enable CMM CLI debugs
  events   Enable CMM state machine event debugs
  misc     Enable CMM misc debugs
  packets  Reception and transmission of packets (can be filtered based on IP address)
  shell    Enable CMM misc debugs
  timers   Enable CMM misc debugs

router# debug appnav-controller fdm ?
  all     Enable all FDM debugs
  events  Enable debugging for important events being handled by FDM
  infra   Enable debugging for FDM infrastructure events
```

The following debug commands are the most useful:

- **debug appnav-controller cmm events**
- **debug appnav-controller fdm events**
- **debug appnav-controller ha**

Debugging the Cisco IOS-XE Infrastructure

This section contains the following information:

- Showing Packet Drop Statistics, page 5-2
- Showing Data Path CPU Utilization, page 5-3
- Showing Data Path Memory Utilization, page 5-3

Showing Packet Drop Statistics

Use the following command to display unplanned packet drops:

```
router# show platform hardware qfp active statistics drop
-------------------------------------------------------------------------
Global Drop Stats                         Packets                 Octets
-------------------------------------------------------------------------
AppNavBadRoute                                 38                   2888
Ipv4AclLookupMiss                              42                   3034
Ipv4NoRoute                                  4408                1293334
UnconfiguredIpv4Fia                            19                   1710
```

The following are the reasons for which packets may drop:

- AppNavInvSNpkt—Malformed or unsupported packet from the service node.

- AppNavInternalErr—Logic error within the AppNav-XE component. Uncommon.
- AppNavBadRoute—A non-AppNav-XE packet appeared at the AppNav-XE virtual or tunnel interface. Very common when routing protocols are enabled.
- AppNavNoTunnel—There is no tunnel facility available for the service node-bound packet.
- AppNavNoSvcCtx—There is no service context matching the flows from the service node.
- AppNavInvFOState—The flow state is no longer valid. This is usually due to changes in the configuration.
- AppNavUnexpctdpkt—The AppNav-XE component did not expect to process more packets because it has been shut down.

Showing Data Path CPU Utilization

To display the data path CPU utilization, use the following command:

```
router# show platform hardware qfp active datapath utilization

    CPP 0: Subdev 0      5 secs       1 min      5 min      60 min
Input: Priority (pps)    0            0          0          0
               (bps)     0            72         88         48

Non-Priority (pps)       226455       225968     198785     72441
             (bps)       1879325304   1875408168 1648044616 599951168

        Total (pps)      226455       225968     198785     72441
             (bps)       1879325304   1875408240 1648044704 599951216

Output: Priority (pps)   229023       228474     245267     90057
               (bps)     1619093520   1641710256 2389617496 949076160

Non-Priority (pps)       209522       208053     293300     104501
               (bps)     180090080    178161632  3124680344 1191566064

            Total (pps)  438545       436527     538567     194558
                 (bps)   1799183600   1819871888 5514297840 2140642224

Processing: Load (pct)   26           26         19         8
```

Showing Data Path Memory Utilization

Use the following command to show statistics about the data path memory use.

Note The value for **In Use DRAM memory** must be less than 90 percent of the value for **Total DRAM memory**; otherwise, the AppNav-XE component stops optimizing new flows.

```
router# show platform hardware qfp active infrastructure exmem statistics

QFP exmem statistics
Type: Name: DRAM, QFP: 0
    Total: 268435456
    InUse: 99933184
    Free: 168502272
    Lowest free water mark: 168502272

Type: Name: IRAM, QFP: 0
    Total: 134217728
    InUse: 8087552
    Free: 126130176
    Lowest free water mark: 126130176

Type: Name: SRAM, QFP: 0
    Total: 32768
    InUse: 15088
    Free: 17680
    Lowest free water mark: 17680
```

Debugging the Data Plane

The output of the following debug command is displayed as a log file named /tmp/fp/trace/cpp_cp_*Fx*-0.log under the FP shell, where *Fx* is either F0 or F1 depending on the active FP module. You need a shell license to access the FP shell.

If you do not have shell access, you can use the **test platform software trace slot fp act cpp-control-process rotate** command to force the log to flush to bootflash:tracelogs.

```
router# debug platform hardware qfp active feature appnav-controller datapath ?
classify    Debug QFP flow classification such as traces, policy, peer ID, and
               classification action (which service node group)
drop        Enable drop debugging and shows traces of packet drop due to errors
fdl         Debug QFP flow distribution such as selecting a service node within a service
               node group
ha          Debug QFP high availability (HA) and AppNav Controller issues. Shows traces
               related to syncing flows between AppNav Controllers and between active and
               standby FPs
interop     Debug QFP feature interoperations such as FNF, NBAR, and NAT
pkt-path    Debug QFP packet processing and packet interception
proxy       Debug QFP proxy issues related to interface with the control place, such as
               statistics reporting and configuration
```

Each of the above categories (other than **drop**, which has no level) has the following four levels:

- Error—Displays error level debugs and detects potential issues.
- Warn—Displays warnings and errors.
- Info—Displays information, warnings, and errors.
- All—The lowest level of debugging. Displays all debugs.

To limit the number of debug messages, we recommend that you only enable the error debug level first and then slowly reduce the debug level.

You can also use the following command to check on packets dropped by the router. The command lists all the packets that were dropped with a reason. If you see AppNav drop reasons, you can enable the debug drop command to see the actual packet drops inside the trace logs.

```
router# show platform hardware qfp active statistics drop

Global Drop Stats         Packets        Octets
-------------------------------------------------------
The Global drop stats were all zero
```

AppNav Service Node Auto Discovery Debug Commands

Use the following debug commands to trace the AppNav service node auto discovery feature:

- **debug appnav-controller auto-discovery**
- **debug mdns all**

- **debug mdns packet**

Container Debug Commands

The following debug commands are available to debug the application configuration, installation, activation, and status:

- **debug virtual-service virtualPortGroup**
- **debug virtual-service timeout**
- **debug virtual-service messaging**
- **debug virtual-service all**

EZConfig Debug Commands

Use the following commands to debug the operation of the EZConfig program:

- **debug service-integration api**
- **debug service-integration configuration**
- **debug service-integration menu**

Common Problems

Traffic Not Redirected

If traffic is not redirected properly, ensure that "service-insertion waas" is present on interfaces on which the traffic is supposed to be intercepted. Issue the **show service-insertion status** command to verify this.

Traffic Passed Through Instead of Redirected

The **show service-insertion statistics connection** command indicates whether traffic is passed through or redirected. If traffic is passed through instead of being redirected, use the **show policy-map target service-context** *context_name* **passthru-reason** command to find out the reason. For details, see the "Displaying Pass Through Reason Statistics" section on page 4-10.

You can also monitor the service node counters. See the "Displaying Per Service Node and Service Node Group Statistics" section on page 4-6.

The term "Initial Redirect" indicates that flows are being redirected to the service nodes. If the flows are not being redirected to the service nodes, maybe the policy did not cover the traffic type.

The "Initial Redirect -> Passthrough" counter indicates that the service node has decided to pass-through the flow. This is likely due to policies on the service node.

The "Redirect -> Passthrough" counter indicates that the service node later decided to pass-through the flow. This is likely due to lack of a peer WAAS device. Two WAAS devices are needed along the path to optimize a flow.

Traffic Not Optimized

The following can cause traffic to not be optimized:

- One or both disks failed
- Configuration mismatch between the AppNav-XE component and the WAAS policy
- The AppNav-XE component not redirecting the traffic
- A resource is not available
- ISR-WAAS configuration
- ISR-WAAS crashes

Degraded Cluster

If connections are passed through and you are using an AppNav Controller group that has two or more AppNav Controllers, it is possible that the cluster state is degraded instead of operational. This means that the AppNav Controller view is not the same on each of the AppNav Controllers.

To check the cluster state and the stable AppNav Controller view on each of the AppNav Controllers, use the following command:

```
router# show service-insertion service-context

Service Context                      : waas/1
Cluster protocol ICIMP version       : 1.1
Cluster protocol DMP version         : 1.1
Time service context was enabled     : Fri Dec 7 19:28:11 2012
Current FSM state                    : Degraded
```

```
Time FSM entered current stat         : Fri Dec 7 21:58:29 2012
Last FSM state                        : Converging
Time FSM entered last state           : Fri Dec 7 21:58:19 2012
Cluster operational state             : Degraded

Stable AppNav controller View:
    21.0.0.145
    21.0.0.160

Stable SN View:
    21.0.0.149
```

The reason for the difference in AppNav Controller views on the AppNav Controllers may be due to a mismatch in the AppNav Controller group configuration on the AppNav Controllers or due to a connectivity problem between the AppNav Controllers.

It is also useful to check the alarms on each of the AppNav Controllers by using the following command that also suggests corrective actions:

```
router# show service-insertion alarms detail support

Critical Alarms:
----------------
    Alarm Instance     Alm ID  Module      AC/SN IP Addr  AO      SNG
    1 degraded_cluster 29002   cmm         N/A            N/A     N/A

    Cluster protocol detected inconsistency in AC view of peer ACs. Device will
pass-through all new connections.

    Cluster view is degraded.

        Explanation:
                Cluster membership manager has detected a discrepancy in the AC view of peer
ACs. Optimization will be turned off on this device for cluster consistency.
        Action:
                Check the network for partial connectivity issues.

Major Alarms:
-------------
    Alarm Instance     Alm ID  Module      AC/SN IP Addr  AO      SNG
    1 ac_unreachable   29006   cmm         192.168.1.11   N/A     N/A

    Cluster protocol on device cannot communicate with peer AC ("192.168.1.11").

        AppNav controller is unreachable.


        Explanation:
                Cluster protocol detected failure of the peer AC. This could happen due to
several reasons - configuration mismatch or network issues preventing communication between
the ACs or the AC actually being down.
    Action:
        Other alarms will indicate if this is a configuration issue. If so, correcting the
configuration mismatch will cause this alarm to go away. Otherwise, check the network to
see if the devices are able to communicate with each other.

Minor Alarms:
-------------
None
```

Service Node Excluded

If no traffic is redirected to a particular service node and you are using an AppNav Controller group with two or more AppNav Controllers, it is possible that the service node is excluded. This happens when the service node view is not the same on each of the AppNav Controllers.

To check the stable service node view on each of the AppNav Controllers, use the **show service-insertion service-context** command.

The reason for the difference in service node views could be due to a mismatch in the service node group configuration on the AppNav Controllers or due to a connectivity problem between one or more of the AppNav Controllers and the excluded service node.

To check if any service nodes are excluded or unreachable, look for the SN_excluded and SN_unreachable alarms by using the **show service-insertion alarms detail support** command on each of the AppNav Controllers.

Flows Not Synced Between AppNav Controllers

This could be due to a mismatch in the VRF names for the traffic seen by the AppNav Controllers in the ACG.

Check the output of the **show service-insertion statistics connection summary** command for the counter for Flow Sync Failures due to vrf mismatch.

```
router# show service-insertion statistics connection summary
Number of 2T entries=0
Number of 3T entries=0
Number of optimized flows=0
Number of pass-through flows=0
Flow sync failures due to vrf-mismatch=3
```

Connection Hangs

A connection might be considered “hung” for various reasons. In many cases, it helps to use telnet to simulate a connection to the server. For example, enter **telnet** *HTTP_server* **80**.

If the connection hangs during the TCP 3-way handshake, verify that both the connection and the route to the service node are properly set up.

If the connection hangs after the connection was established, verify the connection along the path. Make sure that the MTU along the path is correct.

Use the **show service-insertion statistics connection** command on the AppNav Controller and the **show statistics connection** command on the service node to cross check the connections between the AppNav Controller and the service node.

Use the **show platform hardware qfp active statistics drop** command to check for packet drops.

Connection Resets

You can usually see the reason for the connection reset by issuing the **show statistics connection closed** command and the **show statistics connection closed conn-id** *connection_ID* command on the service node. Capturing packets is also useful in analyzing the reason for the connection reset.

If you use ISR-WAAS as the service node, verify the DRE peer ID by using the **show dre** command on the service node and making sure that both of the ISR-WAAS DRE peer IDs are not the same. **vm init** is required for ISR-WAAS during configuration.

Use the **show platform hardware qfp active statistics drop** command to check for dropped packet.

Application Accelerator Status Shows as Red with No Load

Some older service nodes may not support all application accelerators.

Individual application accelerators, such as the video application accelerator, require a separate license.

The AppNav-XE Component Fails to Initialize

If the system displays an ERROR_NOTIFY syslog message when you enable the **service-insertion waas** command on the interface, it could be that the AppNav-XE component failed to initialize due to low memory. Check the amount of memory by using the following command:

```
router# show platform hardware qfp active infrastructure exmem statistics
QFP exmem statistics

Type: Name: DRAM, QFP: 0
  Total: 268435456
  InUse: 102283264
  Free: 166152192
  Lowest free water mark: 166152192
Type: Name: IRAM, QFP: 0
  Total: 134217728
  InUse: 8186880
  Free: 126030848
  Lowest free water mark: 126030848
Type: Name: SRAM, QFP: 0
  Total: 32768
  InUse: 15088
  Free: 17680
  Lowest free water mark: 17680
```

If the available memory is less than 10 percent of the total memory, the AppNav-XE component may not be able to initialize, which results in no flows being redirected.

If the output of the **show policy-map target service-context waas/1** command is blank, instead of listing the AppNav policy being used, it may indicate that the system was unable to initialize.

Flow Limit Reached

Both the AppNav Controller and the service nodes have a limit on the number of flows that they can support. On the AppNav Controller, the limit is 2 million flows. Beyond that, all flows are passed through. If you exceed the limit, the system displays the following error message:

```
03/10 00:53:51.720 [errmsg]: (warn): %CFT_CLIENT-4-MAX_FCS_TOUCH_WARN: CFT number of
flow-context threshold is reached, can't allocate more memory for flow-context.
```

The flow limit may be reached in advance due to available memory. In this case, the system displays the following syslog message:

```
*Aug 24 00:29:17.205: %CFT_CLIENT-4-CFT_MEMORY_BOUNDARY_TOUCH_WARN: F0: cpp_cp:  CFT
reached maximum configured memory utilization. Can't allocate more memory for
flow-context.
```

In both cases, when the existing flows are completed and the number of flows dips below the threshold, flows are optimized again.

Application Installation Errors

The following errors can happen during the application installation:

- Code signing failure.
- Not able to extract the libvirt data.
- Resource not available on box.
- Profile selection failed due to the profile database not being populated correctly or not being synced to Cisco IOS-XE.
- The OVA package is not compatible with Cisco IOS-XE or the AppNav-XE component.
- ISR-WAAS is not able to get the boot strap configuration.
- The Cisco IOS-XE image does not support KVM, ISR-WAAS, or the AppNav-XE component.

These errors might occur due to an incompatibility issue between the ISR-WAAS OVA and the Cisco IOS-XE image. Ensure that you have the Cisco IOS-XE Release 3.9 image and the latest compatible OVA package. When selecting an OVA profile to use during activation, make sure that the Cisco ISR 4451-X has enough memory and storage resources for the profile selected.

Degraded Performance

If your performance is degraded, first check the CPU utilization on the application using the following command:

```
router# show virtual-service utilization name ISR4451-X-WAAS_application_name
```

If the application CPU is at 100 percent, check the bandwidth and compression ratio in the ISR4451-XWAAS application. If the bandwidth is within the profile limits, enable ISR-WAAS debugging. If the application CPU is not at 100 percent, check the router CPU using the following command:

```
router# show processes cpu
```

If the platform CPU is at 100 percent, follow the standard platform debug process. If the application is bound by bandwidth and there is CPU headroom on the Cisco ISR 4451-X, consider upgrading the application profile by reinstalling the ISR-WAAS application and selecting a higher profile.

Another reason for degraded performance is application disk failure. Check the ISR-WAAS installation log and ISR-WAAS alarms on WCM to see if any disk failure has occurred.

End Users Cannot Reach the Server

If ISR-WAAS crashes, traffic should not be impacted because the AppNav-XE component stops redirecting traffic.

Note Crashes in ISR-WAAS application accelerators do not impact connectivity because traffic goes into bypass mode.

Other AppNav-XE Known Issues

If the AppNav Controller does not respond to a WAAS TCP trace, the system forwards the TCP trace to the service node and the service node generates a response along with a list of service nodes along the path.

Accessing the ISR-WAAS Application

You can access the ISR-WAAS application by using the following CLI command:

```
router# virtual-service connect name AUTOWAAS console
Connected to appliance. Exit using ^c^c^c
Cisco Wide Area Application Engine Console

Username: admin
Password: xyz
System is initializing. Please wait...
Please use 'show disks details' to monitor system status.
NO-HOSTNAME#
```

For more information, see the *WAAS Configuration Guide*.

Example of ISR-WAAS Running Configuration

A sample ISR-WAAS configuration is displayed below:

Note This sample does not include the policy configuration.

```
router# show run no-policy
! waas-universal-k9 version 5.1.0 (build b15 Aug 17 2012)
!
device mode application-accelerator
!
interception-method appnav-controller
!
!
hostname router-ISR4451-X-WAAS
!
!
primary-interface Virtual 1/0
!
interface Virtual 1/0
ip address 2.79.27.2 255.255.255.0
exit
!
ip default-gateway 2.79.27.1
!
!
```

```
no auto-register enable
!
! ip path-mtu-discovery is disabled in WAAS by default
!
username admin password 1 ****
username admin privilege 15
!
!
authentication login local enable primary
authentication configuration local enable primary
!
tfo tcp optimized-send-buffer 2048
tfo tcp optimized-receive-buffer 2048
!
!
no dre auto-bypass enable
!
no service-insertion pass-through offload enable
!
!
kernel kdb
central-manager address 2.79.5.12
cms enable
!
!
!
stats-collector logging enable
stats-collector logging rate 30
!
service-insertion service-node
enable
exit
!
! End of WAAS configuration
```