



# Monitoring the AppNav-XE and ISR-WAAS Components

---

This chapter describes how to monitor the AppNav-XE and ISR-WAAS components and contains the following sections:

- [AppNav Controller Show Commands, page 4-1](#)
- [AppNav Service Node Auto Discovery Show Commands, page 4-12](#)
- [Container Show Commands, page 4-14](#)

## AppNav Controller Show Commands

You can use **show** commands to check status and display data.

- [Checking the Status of the AppNav Controller, page 4-1](#)
- [Checking the Membership of the AppNav Controller Group, page 4-2](#)
- [Displaying Detailed Information About Service Node Groups and Service Nodes, page 4-2](#)
- [Displaying Class Maps and Policy Maps, page 4-3](#)
- [Displaying Service Context Information, page 4-4](#)
- [Displaying Data Path Statistics, page 4-5](#)
- [Displaying Alarms, page 4-12](#)

## Checking the Status of the AppNav Controller

Use the following command to check on the general status of the AppNav Controller. The command also lists all the interfaces that have “service-insertion waas” configured.

```
router# show service-insertion status
```

```
Hostname: Branch-router  
Device ID:30f7.0d54.5510  
Platform Type:cisco (ISR4452/K9) 2RU  
IOS Version: 15.3(20130102:194350)  
AppNav Controller Version: 1.0.0  
AppNav Enabled Interfaces:  
GigabitEthernet0/0/1
```

## Checking the Membership of the AppNav Controller Group

Use the following command to check the membership of the AppNav Controller group. It also lists all the service nodes configured and registered with the AppNav Controller.

```

router# show service-insertion appnav-controller-group
All AppNav Controller Groups in service context
Appnav Controller Group                : acg
Member Appnav Controller Count         : 2
Members:
  IP Address
    21.0.0.36
    21.0.0.160

AppNav Controller                      : 21.0.0.36
Local AppNav Controller                 : Yes
Current status of AppNav Controller    : Alive
Time current status was reached        : Wed Sep  5 15:50:06 2012
Cluster protocol ICIMP version         : 1.1
Cluster protocol Incarnation Number    : 1
Cluster protocol Last Sent Sequence Number : 0
Cluster protocol Last Received Sequence Number : 0
Current AC View of AppNav Controller
  IP Address
    21.0.0.36
    21.0.0.160

Current SN View of AppNav Controller
  IP Address
    21.0.0.149

AppNav Controller                      : 21.0.0.160
Local AppNav Controller                 : No
Current status of AppNav Controller    : Alive
Time current status was reached        : Thu Dec  6 20:17:53 2012
Cluster protocol ICIMP version         : 1.1
Cluster protocol Incarnation Number    : 1
Cluster protocol Last Sent Sequence Number : 1355098374
Cluster protocol Last Received Sequence Number : 1355089899

Current AC View of AppNav Controller
  IP Address
    21.0.0.36
    21.0.0.160

Current SN View of AppNav Controller
  IP Address
    21.0.0.149

```

## Displaying Detailed Information About Service Node Groups and Service Nodes

Use the **show service-insertion service-node-group** [*sng\_name* | **all**] command to display detailed information about service node groups and individual service nodes. You can also use this command to check the status of individual application accelerators.

The output of this command shows the following:

- Cluster protocol information. The *last sent sequence number* and the *last received sequence number* values should be increasing continuously.

- Number of service nodes and associated service contexts.
- Status of each service node, which can be either Alive or Dead
- Load state, which displays the health of the application accelerators. The load state can be one of the following:
  - green—application accelerator is functional and accepting new flows
  - yellow—application accelerator is functional but not accepting new flows
  - red—application accelerator is not functional
- Overall availability of the service node group for each application accelerator

```

router# show service-insertion service-node-group
Service Node Group name :sng1
  Service Context :          waas/1
  Member Service Node count : 1

Service Node (SN) :          21.0.0.149
Auto discovered :           No
SN belongs to SNG :         sng1
Current status of SN :      Alive
Time current status was reached : Thu Dec 6 20:17:11 2012

Cluster protocol DMP version :          1.1
Cluster protocol incarnation number :    2
Cluster protocol last sent sequence number : 1355101043
Cluster protocol last received sequence number: 1348909100

Health Markers:
AO      Load State      Since
tcp     GREEN              0d 5h 39m 38s
epm     GREEN              0d 5h 39m 38s
cifs    GREEN              0d 5h 39m 38s
mapi    GREEN              0d 5h 39m 38s
http    GREEN              0d 5h 39m 38s
video   GREEN              0d 5h 39m 38s
nfs     GREEN              0d 5h 39m 38s
ssl     YELLOW             0d 5h 39m 38s
ica     RED                0d 0h 0m 0s

SNG Availability per Accelerator
AO      Available      Since
tcp     Yes             0d 5h 39m 38s
epm     Yes             0d 5h 39m 38s
cifs    Yes             0d 5h 39m 38s
mapi    Yes             0d 5h 39m 38s
http    Yes             0d 5h 39m 38s
video   Yes             0d 5h 39m 38s
nfs     Yes             0d 5h 39m 38s
ssl     No              0d 0h 0m 0s
ica     No              0d 0h 0m 0s

```

## Displaying Class Maps and Policy Maps

The following commands reflect the running configuration and are useful for checking classifications without having to scan through an entire running configuration.

To display all type AppNav class maps and their matching criteria, or a specific AppNav class map and its matching criteria, use the following command:

```

router# show class-map type appnav [AppNav_class_name]

```

To display all type AppNav policy maps and their class and action mappings, or a specified policy map and its class or action mappings, use the following command:

```
router# show policy-map type appnav [AppNav_policy_name]
```

The **show policy-map target service-context** [*service\_context\_name*] command displays policy map information for service contexts. Use this command to view the flow level stats of all the class maps and policy maps that are configured under a service context. If you do not specify a service context name, the command displays all the configured class maps and policy maps.

Here are two examples:

```
router# show policy-map target service-context waas/1
Service-policy appnav input: p1
```

```
Class-map: c1 (match-all)
  Match: access-group 101
  distribute service-node-group sng1
    Distributed: 0 packets, 0 bytes
    Passed through: 0 packets, 0 bytes
  Aggregate: 0 packets, 0 bytes
  monitor-load http
Class-map: class-default (match-any)
  Match: any
```

```
router# show policy-map target service-context
```

```
Service-policy appnav input: p1
Class-map: c1 (match-all)
  Match: access-group 101
  distribute service-node-group sng1
    Distributed: 0 packets, 0 bytes
    Passed through: 0 packets, 0 bytes
  Aggregate: 0 packets, 0 bytes
  monitor-load http

Class-map: class-default (match-any)
  Match: any
```

```
Service-policy appnav input: p3
```

```
Class-map: c3 (match-all)
  Match: access-group 101
  distribute service-node-group sng3
    Distributed: 0 packets, 0 bytes
    Passed through: 0 packets, 0 bytes
  Aggregate: 0 packets, 0 bytes

Class-map: class-default (match-any)
  Match: any
```

## Displaying Service Context Information

To display information about service contexts, use the **show service-insertion service-context** [*service\_context\_name*] command. The output of this command displays the status of the specified service context, including the following:

- Current and last states of the Cluster Membership Manager (CMM) and FSM
- State of the cluster
- Views of the stable and current AppNav Controller and service nodes

Here is an example:

```
router# show service-insertion service-context waas/1

Service Context                               : waas/1
Cluster protocol ICIMP version                : 1.1
Cluster protocol DMP version                 : 1.1
Time service context was enabled              : Thu Sep  8 08:38:41 2011
Current FSM state                             : Operational
Time FSM entered current state                : Thu Sep  8 08:48:26 2011
Last FSM state                                : Converging
Time FSM entered last state                  : Thu Sep  8 08:48:16 2011
Cluster operational state                     : Operational

Stable AppNav Controller View:
    2.58.2.40
Stable SN View:
    2.43.139.170    2.58.2.40
Current AppNav Controller View:
    2.58.2.40
Current SN View:
    2.43.139.170    2.58.2.40
```

## Displaying Data Path Statistics

- [Displaying AppNav Controller Group Statistics, page 4-5](#)
- [Displaying Per Service Node and Service Node Group Statistics, page 4-6](#)
- [Displaying Service Context Statistics, page 4-7](#)
- [Displaying Flow Statistics, page 4-8](#)
- [Displaying Application and Session Statistics, page 4-9](#)
- [Displaying Classification Statistics, page 4-10](#)
- [Displaying Pass Through Reason Statistics, page 4-10](#)

## Displaying AppNav Controller Group Statistics

To see the number of “keepalives” sent to the other AppNav Controllers and received from the other AppNav Controllers and other statistics related to the AppNav Controller group, use the following command:

```
router# show service-insertion statistics appnav-controller-group
Appnav Controller Group           : acg
Number of AppNav Controllers      : 2
Members:
  IP Address
  21.0.0.36
  21.0.0.160

Aggregate Appnav Controller statistics
-----
Time since statistics were last reset/cleared : 0d 5h 47m 14s

Aggregate number of keepalives sent to ACs      : 168484
Aggregate number of keepalives received from ACs : 166372
Aggregate number of invalid keepalives received :
  Total                                         : 0
  Incompatible ICIMP version                   : 0
```

```

Authentication Failed           : 0
Stale keepalive                 : 0
Malformed keepalive            : 0
Unknown keepalive              : 0
Inactive keepalive             : 0
Aggregate number of times liveliness lost with ACs : 1
Aggregate number of times liveliness gained with ACs: 2

```

## Displaying Per Service Node and Service Node Group Statistics

To show the connections, packets, and bytes sent to each service node, use the following command:

```
router# show service-insertion statistics service-node [IP_address]
```

To show the aggregated connections, packets, and bytes sent to each service node group, use this command:

```
router# show service-insertion statistics service-node-group [NAME]
```

Here is an example:

```

router# show service-insertion statistics service-node

Statistics for Service Node 21.0.0.149
-----
Time since statistics were last reset/cleared: 0d 18h 7m 54s
Number of probe requests sent to SN: 326024
Number of probe responses received from SN: 326014
Number of invalid probe responses received:
  Total      : 0
  Incompatible DMP version: 0
  Authentication failed: 0
  Stale response: 0
  Malformed response: 0
  Unknown response: 0
Number of times liveliness lost with SN: 0
Number of times liveliness regained with SN:1

Cluster IPC statistics
-----
Time since statistics were last reset/cleared: 0d 18h 8m 24s
Number of load updates received from CMM: 4
Number of erroneous load updates: 0
Time since last load update was received: 0d 14h 32m 43s

Load stats for Service Node 21.0.0.149
-----

Accelerator state transition statistics
-----
Time since Accl load stats were last cleared: 0d 18h 8m 24s
Accl   Current   Previous   Red     Yellow   Green
tcp    GREEN        RED        0       0        1
epm    GREEN        RED        0       0        1
cifs   GREEN        RED        0       0        1
mapi   GREEN        RED        0       0        1
http   GREEN        RED        0       0        1
video  GREEN        RED        0       0        1
nfs    GREEN        RED        0       0        1
ssl    YELLOW       RED        0       1        0
ica    RED          RED        0       0        0

Traffic distribution statistics for service node 21.0.0.149

```

```

-----
Time since distribution stats were last cleared: 0d 18h 8m 24s

Packet and byte counts
-----
Redirected Bytes: 2338
Redirected Packets: 50
Received Bytes: 3350
Received Packets: 50

Occurences
-----
Initial Redirects: 2
Initial Redirects Accepted: 2
Initial Redirect -> Passthrough: 0
Redirect -> Passthrough: 0

```

The important statistics are as follows:

- Probe Requests: The number of heartbeats sent to the service node.
- Probe Responses: The number of heartbeats received from the service node.
- Redirected Bytes: The number of bytes redirected to the service node.
- Redirected Packets: The number of data packets redirected to the service node.
- Received Bytes: The number of bytes received from the service node.
- Received Packets: The number of data packets received from the service node.
- Initial Redirects: The number of times that the SYN packet (the first packet for requesting connection in a TCP flow) was redirected to the service node.
- Initial Redirects Accepted: The number of times that the service node decided to optimize on SYN packet.
- Initial Redirects -> Passthrough: The number of times that the service node decided to pass-through on SYN packet.
- Redirect -> Passthrough: The number of times that the service node decided to pass-through a flow after it was initially accepted for optimize (e.g. due to lack of peer).

## Displaying Service Context Statistics

To display statistics about the service context, use the **show service-insertion statistics service-context** [*name*] command. The output of this command displays the time spent in each FSM state by the CMM and the amount of time that each service context has been in each FSM state.

Here is an example:

```

Router# show service-insertion statistics service-context
Time spent in various FSM states

Converging      :      0d 0h 0m 31s
Initializing    :      0d 0h 0m 0s
Operational     :      1d 19h 27m 53s
Degraded        :      0d 0h 0m 0s
Internal Error  :      0d 0h 0m 0s
Admin Disabled  :      0d 0h 0m 0s

Number of entries into Converging State:      3
Number of entries into Initializing State:     1
Number of entries into Operational State:      3
Number of entries into Degraded State:         0

```

```
Number of entries into Internal Error State: 0
Number of entries into Admin Disabled State: 0
```

## Displaying Flow Statistics

To query the flows in the flow table and to optionally filter the output by using specific criteria, use the following command:

```
router# show service-insertion statistics connection [[summary] | [vrf-name name]
[client-ip IP_address] [client-port port_number] [server-ip IP_address] [server-port
port_number] [detail]]
```

As part of the flow query, the following information for every flow is available:

- Client IP address, client TCP port and server IP address, server TCP port number
- Service node IP address, passthrough
- VRF name

Here is an example:

```
router# show service-insertion statistics connection
Collecting Records. Please wait...
Client          Server          SN-IP          VRF-Name
51.0.222.4:64234 11.0.0.3:80    21.0.0.104    br_vrf
51.0.222.4:22415 11.0.0.3:80    21.0.0.104
51.0.222.4:15264 11.0.0.3:80    21.0.0.104
51.0.222.4:37759 11.0.0.3:80    21.0.0.104
51.0.222.4:55408 11.0.11.2:23   Passthrou
```

If you include the *detail* keyword, the report also displays the following on a per flow basis:

- Presence of session (3T) or App (2T) association
- Application ID
- Peer ID

The following is an example:

```
router# show service-insertion statistics connection detail
Collecting Records. Please wait...

Client: 192.168.80.4:60973
Server: 192.168.180.4:135
Service Node IP: 172.16.0.2
Flow association: 2T:No,3T:No
VRF-Name:
Application ID: 0
Peer-ID: 00:21:5e:76:65:08

Client: 192.168.80.4:60959
Server: 192.168.180.4:1092
Service Node IP: 172.16.0.2
Flow association: 2T:Yes,3T:Yes
VRF-Name:
Application ID: 78
Peer-ID: 00:21:5e:76:65:08
```

If you include the *summary* keyword, the report displays only the number of 2T and 3T entries, the number of optimized flows, the number of passthrough flows, and the number of flow synchronization failures due to VRF config mismatch on the AppNav Controllers.

The following is an example:



```
router# show service-insertion statistics connection summary
Number of 2T optimized flows    = 0
Number of 3T optimized flows    = 0
Number of optimized flows       = 3
Number of pass-through flows    = 1
Flow sync failures due to vrf mismatch = 0
```

You can also use the **show platform software** command. It works exactly the same as the **show service-insertion statistics** command, but it can also be used to query the flows on the standby FP.

```
router# show platform software appnav-controller <f0 | f1 | fp active | fp standby>
connections ...
```

## Displaying Application and Session Statistics

To query the application and session entries and to optionally filter the output by using specific criteria, use the following command:

```
router# show service-insertion statistics sessions [[vrf-name name] [client-ip
IP_address][server-ip IP_address] [server-port port_number] [detail]]
```

Application entries do not have client or service node IP addresses.

Here is an example:

```
router# show service-insertion statistics sessions

Collecting Records. Please wait...
Client          Server          SN-IP          VRF-Name
N/A             192.168.180.4:1092  N/A
192.168.80.4:0  192.168.180.4:1092  172.16.0.2
```

If you include the *detail* keyword, the report also displays the application ID and the time since the last activity.

Here is an example:

```
Router# show service-insertion statistics sessions detail
Collecting Records. Please wait...
Client: 192.168.80.4:0
Server: 192.168.180.4:1098
Service Node IP: 172.16.0.2
VRF-Name:
Application ID: 78
Time since last activity : 0hr 36min 30sec

Client: N/A
Server: 192.168.180.4:1098
Service Node IP: N/A
VRF-Name:
Application ID: 78
Time since last activity : 0hr 36min 30sec
```

You can also use the **show platform software** command. It works exactly the same as the **show service-insertion statistics** command, but it can also be used to query the application and session entries on the standby FP.

```
router# show platform software appnav-controller <f0 | f1 | fp active | fp standby>
sessions ...
```

## Displaying Classification Statistics

Use the **show policy-map target service-context** [*service\_context\_name*] command to view the flow level statistics of all the class maps and policy maps that are configured under a service context. If you do not enter a service context name, the system displays all the configured class maps and policy map output.

The following are examples:

```
router# show policy-map target service-context waas/1
```

```
Service-policy appnav input: p1
Class-map: c1 (match-all)
  Match: access-group 101
  distribute service-node-group sng
    Distributed: 313450 packets, 135820480 bytes
    Passed through: 0 packets, 0 bytes
  Aggregate: 313450 packets, 135820480 bytes
```

```
monitor-load http
```

```
Class-map: c2 (match-all)
  Match: access-group 102
  Pass-through
    Distributed: 0 packets, 0 bytes
    Passed through: 40 packets, 30000 bytes
  Aggregate: 40 packets, 30000 bytes
```

```
Class-map: class-default (match-any)
  Match: any
```

```
router# show policy-map target service-context
```

```
Service-policy appnav input: p1
Class-map: c1 (match-all)
  Match: access-group 101
  distribute service-node-group sng1
    Distributed: 0 packets, 0 bytes
    Passed through: 0 packets, 0 bytes
  Aggregate: 0 packets, 0 bytes
  monitor-load http
```

```
Class-map: class-default (match-any)
  Match: any
```

```
Service-policy appnav input: p3
Class-map: c3 (match-all)
  Match: access-group 101
  distribute service-node-group sng3
    Distributed: 0 packets, 0 bytes
    Passed through: 0 packets, 0 bytes
  Aggregate: 0 packets, 0 bytes
```

```
Class-map: class-default (match-any)
  Match: any
```

## Displaying Pass Through Reason Statistics

To view the passthrough reason statistics aggregated for all the classes of a policy associated with the specified service context, use the following command:

```
router# show policy-map target service-context context_name passthru-reason
```

To view the passthrough reason statistics for a particular class of a policy associated with the specified service context, use the following command:

```
router# show policy-map target service-context context_name class class_name
passthru-reason
```

Here is an example:

```
router# show policy-map target service-context waas/1 class c4 passthru-reason
```

```
Service-policy appnav input: p4
```

```
Class-map: c4 (match-all)
  Match: access-group 101
  distribute service-node-group sng4
    Distributed: 11 packets, 222 bytes
    Passed through: 100 packets, 22000 bytes
  Aggregate: 111 packets, 22222 bytes
Collected by SC:
```

Passthrough Reasons	Packets	Bytes
PT Flow Learn Failure	0	0
PT SNG Overload	0	0
PT Appnav Policy	0	0
PT Cluster Degrade	0	0
PT ZBFW	0	0
PT NAT ALG	0	0
PT Unknown	0	0

Indicated by SN:

Passthrough Reasons	Packet	Bytes
PT No Peer	100	22000
PT Rjct Capabilities	0	0
PT Rjct Resources	0	0
PT Rjct No License	0	0
PT App Config	0	0
PT Global Config	0	0
PT Asymmetric	0	0
PT In Progress	0	0
PT Intermediate	0	0
PT Overload	0	0
PT Internal Error	0	0
PT App Override	0	0
PT Server Black List	0	0
PT AD Version Mismatch	0	0
PT AD AO Incompatible	0	0
PT AD AOIM Progress	0	0
PT DM Version Mismatch	0	0
PT Peer Override	0	0
PT Bad AD Options	0	0
PT Non-optimizing Peer	0	0
PT SN Interception ACL	0	0
PT IP Fragment Unsupported	0	0
PT Overall	100	22000

## Displaying Alarms

Use the following command to display the alarms seen on the AppNav Controller. The **detail** option gives a brief explanation of each alarm and the **support** option gives a longer explanation along with a recommended action.

```
router# show service-insertion alarms [critical | major | minor] [detail [support]]
```

The following is an example:

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

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.

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").

2 sn_unreachable	29007	cmm	192.168.2.31		N/A	N/A
------------------	-------	-----	--------------	--	-----	-----

Cluster protocol on device cannot communicate with peer SN ("192.168.2.31").

3 sng_unavailable	30001	fdm	N/A		N/A	sng1
-------------------	-------	-----	-----	--	-----	------

Service Node Group ("sng1") has become unavailable.

4 sng_ao_unavailable	30000	fdm	N/A		sslsng	
----------------------	-------	-----	-----	--	--------	--

Service Node Group ("sng") has become unavailable for accelerator - ("ssl").

Minor Alarms:

-----

None

## AppNav Service Node Auto Discovery Show Commands

Use the following commands to show information about the AppNav service node auto discovery feature.

**show service-insertion service-node-group *sng\_name***

```
router# show service-insertion service-node-group sng
```

```
Service Node Group name : sng
Service Context : waas/1
Member Service Node count : 2
```

```
Service Node (SN) : 20.20.20.20
Auto discovered : Yes
SN belongs to SNG : sng
Current status of SN : Alive
```

Time current status was reached : Thu Dec 6 00:51:48 2012

Cluster protocol DMP version : 1.1  
 Cluster protocol incarnation number : 13  
 Cluster protocol last sent sequence number : 1355026900  
 Cluster protocol last received sequence number: 131214317

Health Markers:

AO	Load State	Since
tcp	YELLOW	0d 14h 3m 53s
epm	RED	0d 0h 0m 0s
cifs	RED	0d 0h 0m 0s
mapi	RED	0d 0h 0m 0s
http	RED	0d 0h 0m 0s
video	RED	0d 0h 0m 0s
nfs	RED	0d 0h 0m 0s
ssl	RED	0d 0h 0m 0s
ica	RED	0d 0h 0m 0s

Service Node (SN) : 1.2.3.4  
 Auto discovered : No  
 SN belongs to SNG : sng  
 Current status of SN : Dead  
 Time current status was reached : Thu Dec 6 14:19:52 2012

Cluster protocol DMP version : 0.0  
 Cluster protocol incarnation number : 0  
 Cluster protocol last sent sequence number : 1355026901  
 Cluster protocol last received sequence number: 0

Health Markers:

AO	Load State	Since
tcp	RED	0d 0h 0m 0s
epm	RED	0d 0h 0m 0s
cifs	RED	0d 0h 0m 0s
mapi	RED	0d 0h 0m 0s
http	RED	0d 0h 0m 0s
video	RED	0d 0h 0m 0s
nfs	RED	0d 0h 0m 0s
ssl	RED	0d 0h 0m 0s
ica	RED	0d 0h 0m 0s

SNG Availability per Accelerator

AO	Available	Since
tcp	No	0d 0h 0m 0s
epm	No	0d 0h 0m 0s
cifs	No	0d 0h 0m 0s
mapi	No	0d 0h 0m 0s
http	No	0d 0h 0m 0s
video	No	0d 0h 0m 0s
nfs	No	0d 0h 0m 0s
ssl	No	0d 0h 0m 0s
ica	No	0d 0h 0m 0s

**show service-insertion service-node-group *sng\_name* auto-discovered**

router# **show service-insertion service-node-group sng auto-discovered**

MAC Address	Resp Elapsed Minutes	IP Address
50:57:a8:e1:af:1	0	20.20.20.20

**show mdns req**

router# **show mdns request**

```
MDNS Outstanding Requests
=====
Request name : _appnav_waas_node._udp.local
Request type : PTR
Request class : IN
```

**show mdns stat**

```
router# show mdns stat
```

```
mDNS Statistics
mDNS packets sent : 852
mDNS packets received : 510
mDNS packets dropped : 0
```

## Container Show Commands

- [Displaying Virtual Service Information, page 4-14](#)
- [Displaying Details for a Virtual Service, page 4-14](#)
- [Displaying a List of Virtual Services, page 4-15](#)
- [Displaying Storage Volume Information for a Virtual Service, page 4-16](#)
- [Displaying Statistics for a Virtual Service, page 4-16](#)

## Displaying Virtual Service Information

The **show virtual-service** CLI command provides details about the running application, the profiles supported, storage used by the application, and CPU utilization. See the following example:

```
router# show virtual-service ?
  detail      Detail information about appliance
  list        List the appliance
  profile     information about appliance profile
  storage     Storage information about appliance
  utilization Utilization information about appliance
  version     Version information about appliance
  |          Output modifiers
```

## Displaying Details for a Virtual Service

Use the following CLI command to display details for a virtual service:

```
router# show virtual-service detail
Virtual Service AUTOWAAS Detail:

Package metadata:
Package name       : ISR4451-X-WAAS-eft.ova
Application name   : ISR-WAAS
Application version : 1.0
Application description : WAAS
Certificate type   : N/A
Signing method     : SHA512
Licensing name    : V-WAAS
Licensing version  : 1.0
```

```

OVA path           : /vol/harddisk//ISR4451-X-WAAS-eft.ova
State              : Activated
Detailed guest status :

Activated profile name: ISR-WAAS-750
Disk reservation   : 270784 MB
Memory reservation  : 4096 MB
CPU reservation     : 0% system CPU
VCPUs              : 2

Attached devices:
Type              Name      Alias
-----
HDD               vdc
HDD               vdb
HDD               vda
Serial/Trace      serial3
Serial/Syslog     serial2
Serial/aux        serial1
Serial/shell      serial0
NIC               ieobc_2  ieobc
NIC               dp_2_31  net2

Network interfaces:
MAC address       Attached to interface
-----
54:0E:00:0B:0C:03  ieobc_2
30:F7:0D:53:C6:1F  VirtualPortGroup31

Guest interface:
Interface: eth0
  ip address: 33.1.1.2/24

Guest routes:
Address/Mask      Next Hop      Intf.
-----
0.0.0.0/0        33.1.1.1     eth0

Resource admission (without profile) : passed
Disk space       :
Memory           : 3072MB
CPU              : Not specified
VCPUs           : 1

```

## Displaying a List of Virtual Services

The container infrastructure provides commands to view the status and details of installed applications. To view the list of existing applications, use the following command:

```

router# show virtual-service list
System busy installing virtual-service 'ISR-WAAS'. The request may take several minutes...
Virtual Service List:

```

Name	Status	Package Name
ISR-WAAS	Installing	ISR4451-X-WAAS-eft.ova

## Displaying Storage Volume Information for a Virtual Service

Use the following CLI command to display storage volume information for a virtual service:

```
router# show virtual-service storage volume list
Virtual-Service storage volume list
```

Name	Capacity	In Use	Virtual-Service
vda.AUTOWAAS	4097 MB	Yes	AUTOWAAS
vdb.AUTOWAAS	163841 MB	Yes	AUTOWAAS

## Displaying Statistics for a Virtual Service

Use the following CLI command to display statistics for a virtual service:

```
router# show virtual-service utilization statistics CPU
```

```
-----
/cgroup
-----
directory share system % User % Num CPUs CPU core %
.          1024 70.83   29.17  8          9.57 14.59 7.87 23.65 7.39 17.06 6.51 13.35
./libvirt  8192 75.75   24.25  8          18.05 28.26 11.68 10.70 10.38 9.51 6.06 5.37
./iosbinos 1024 69.21   30.79  8          5.61 10.12 6.53 25.90 6.37 21.43 7.35 16.68

. ./libvirt ./iosbinos
61.33 18.22 20.45
-----
/cgroup/libvirt
-----
directory share system % User % Num CPUs CPU core %
.          8192 75.75   24.25  8          18.05 28.26 11.68 10.70 10.38 9.51 6.06 5.37
./AUTOWAAS 1024 75.75   24.25  8          18.05 28.26 11.68 10.70 10.38 9.51 6.06 5.37

. ./AUTOWAAS
-0.00 100.00
-----
/cgroup/libvirt/AUTOWAAS
-----
directory share system % User % Num CPUs CPU core %
.          1024 75.75   24.25  8          18.05 28.26 11.68 10.70 10.38 9.51 6.06 5.37

.
100.00
-----
/cgroup/iosbinos
-----
directory share system % User % Num CPUs CPU core %
.          1024 69.21   30.79  8          5.61 10.12 6.53 25.90 6.37 21.43 7.35 16.68
100.00
```