Detailed Configuration

This chapter provides detailed configuration information for AppNav-XE on the Cisco ISR 4451-X and contains the following sections:

Configuring the AppNav Controller

To configure the AppNav-XE Controller, follow these configurations tasks:

Configuring AppNav Controller Groups

The AppNav Controller group configures the AppNav Controller. To configure the AppNav Controller group, enter the IP addresses used by the AppNav Controllers.

Restrictions

- The AppNav Controller group must always contain exactly one local IP address. This is the IP address of the local AppNav Controller (the local router). Note that this local IP address must belong to an interface from which all the other AppNav Controllers in the AppNav Controller group and all the service nodes are reachable.

- The AppNav Controller group cannot have more than four AppNav Controllers. This must include exactly one local IP address and optionally up to three non-local IP addresses.
- You can use the IP address from GigE, the VLAN interface, the loopback interface etc, but the interface must not have VRF configured.
- When you have two Cisco ISR AppNav controllers but only a single WAN link, then configure AppNav redirection only on an active WAN router. If the secondary AppNav controllers WAN link is not enabled, the AppNAv controller will drop the queries from an active WAN router / AppNav controller and pass-through traffic will be dropped.
- The system only supports configuration of one AppNav Controller group.

Use the following command:

```
(config)# [no] service-insertion appnav-controller-group group-name
```

Submode command:

```
(config-service-insertion-acg)# [no] appnav-controller IP_address
```

Optional command:

```
(config-service-insertion-acg)# [no] description description_text
```

Configuring Service Node Groups

You must configure a service node under a service node group. The AppNav-XE component intelligently distributes flows to the service node within the service node group.

Beginning with the Cisco IOS XE 3.13 release, a total of 64 service nodes may be included in a cluster. (Earlier releases permitted 32.)

Restriction

You cannot use VRF with either the AppNav Controller or the service node IP address. The IP addresses must be explicitly accessible without VRF. For example, you cannot use the management interface's IP address (with vrf Mgmt-intf) as the AppNav Controller IP address. Routing through VRF leaking is not supported for service-node to Appnav controller communication..

Use the following command:

```
(config)# [no] service-insertion service-node-group sng_name
```

Submode commands:

```
(config-service-insertion-sng)# [no] description group description
(config-service-insertion-sng)# [no] service-node IP_address
```

Configuring AppNav Class Maps

The AppNav-XE component uses AppNav classes to determine the traffic to be used. Use the appnav type class-map to classify the traffic based on the following set of parameters:

- Access list
- Service node peer device ID
- Special protocols supported by the service node

Table 3-1 ACL and ACE Platform Limits

Platforms	ACLs	ACEs (IPv4)	ACEs per ACL(IPv4)	ACEs (IPv6)	ACEs per ACL(IPv6)
Cisco ISR 4451-X	4K	20K	20K	10K	10K

To create or modify a class map to be used for matching connections to a specified class, use the **class-map** command in global configuration mode. To remove an existing class map, use the **no** form of this command. The **class-map** command enters class-map configuration mode in which you can enter an optional description command and one or more of the match commands to configure the match criteria for this class.

The syntax for defining a class map is as shown below:

```
(config)# [no] class-map type appnav [match-all | match-any] appnav_class_name
```

If you do not specify a match, the default is match-all.

```
(config-cmap)# [no] description description_text
(config-cmap)# [no] match access-group {ACL_number | name ACL_name}
(config-cmap)# [no] match peer device_ID
(config-cmap)# [no] match protocol app_def
```

Match Access-Group Command

The **match access-group** command specifies a numbered access-list or named access list whose contents are used as the match criteria against packets to determine if they belong to the class. The access list number ranges from 1 to 2699.

Match Peer Command

The **match peer** command identifies a peer service node that may be performing optimization at the client side of a connection and must be specified in 01:23:45:67:89:ab format. The match peer clause is only useful if the AppNav-XE component is acting as core, that is, receiving a connection that has already been through a peer WAAS device.

Match Protocol Command

The **match protocol** command gets one of the following protocols:

- CITRIX
- MAPI
- MS-AD-REP
- MS-EXCH-NSPI
- MS-FRS
- MS-FRSAPI
- MS-RFR
- MS-SQL
- MSN-MESSENGER
- NETLOGON

The protocol is only used along with additional information provided by the service node to associate the packet with specific applications. The match protocol filter should not be confused with the **monitor-load** keyword in AppNav policy described below.

AppNav NBAR Application

Network Based Application Recognition (NBAR) feature provides intelligent network classification for network infrastructure. NBAR is a classification engine that can recognize a wide variety of applications, including Web-based applications and client/server applications that dynamically assign port numbers. After the application is recognized, the network uses specific services for that particular application.

Using NBAR FIF (First In Flow) capability, AppNav-XE identifies the application information on the SYN packet itself and decides whether traffic needs to be redirected to WAAS. In order to allow APPNAV class maps to define rules based on APP-ID, an nbar-protocol filter is added in AppNav class-map.

The syntax for defining a class map is as shown below:

```
(config)# [no] class-map type appnav [match-all | match-any] appnav_class_name
```

If you do not specify a match, the default is match-all.

Submode commands:

```
(config-cmap)# [no] description description_text
(config-cmap)# [no] match access-group {ACL_number | name ACL_name}
(config-cmap)# [no] match nbar-protocol NBAR Protocols
(config-cmap)# [no] match peer device_ID
(config-cmap)# [no] match protocol app_def
```

APPNAV app-id based class maps

To minimize the size of the **app-id based class group**, use app-id based class maps and then add similar types of app-id class maps with existing AppNav class map types. For every APPNAV class map type, a matching application-id based class map is created. The APPNAV genesis class maps are as follows:

- RTSP
- MAPI
- HTTP
- HTTPs
- CIFS
- Citrix-ICA
- Citrix-CGP
- NFS
- Epmapm
- Default

For each of the above class map, a class map of application ids are created according to customer application policy. The following example shows the HTTP_Optimized_App class map:

```
(config)# class-map type appnav match-any HTTP_Optimized_App
(config-cmap)# match nbar-protocol youtube
(config-cmap)# match nbar-protocol abc-news
```

APPNAV Nested class maps

APPNAV feature supports nested class maps in order to combine two class maps in single APPNAV class map. You can maintain the original class maps and continue the functionality of load reporting and traffic distribution under existing APPNAV protocol class.

The following is the example for HTTP applications:

```
(config)# class-map type appnav match-all HTTP
(config-cmap)# match access-group name APPNAV-ACL-HTTP
(config-cmap)# match class HTTP_Optimized_App
(config-cmap)# APPNAV-ACL-HTTP dstPort {80, 8080, 8088, 8000, 3128}
HTTP_Optimized_App AppIds  {youtube, LoiusViton, NFL}
```

Configuring AppNav Policy Maps

After you configure the AppNav class maps, you can assign actions to them by using an AppNav policy map.

Limits for AppNav Policy Maps, Class maps, and Match Filters Per Class

Table 3-2 lists the limits for AppNav policy maps, class maps, and match filters per class.

Table 3-2 Limits for AppNav Policy Maps, Class maps, and Match Filters Per Class

Policy/Class/Filter Capacity	Cisco ISR 4451-X
Unique policy maps	4096 (16000 from Cisco IOS-XE Release 3.10 for RP2, ESP40, ESP100, ESP200 models only)
Unique class maps	4096
Number of classes per policy map	256
Number of filters per class map	32

To create or modify a policy map that defines the service policy for the candidate optimization traffic, use the **policy-map** command in global configuration mode.

```
(config)# [no] policy-map type appnav appnav_policy_name
```

Submode commands:

```
(config-pmap)# [no] description description_text
(config-pmap)# [no] class appnav_class_name
```

The **class** command above enters the policy-map-class configuration submode:

```
(config-pmap-c)# [no] distribute service-node-group SNG_name
(config-pmap-c)# [no] monitor-load application_accelerator_name
(config-pmap-c)# [no] pass-through
```

Distribute Command

The **distribute** command is the most common action in this class. The system sends the traffic that matches the class map to the service node group identified by the specified *sng_name* parameter. If no service node group is available, or if no distribute is specified, the default action is to pass-through the traffic.

To configure primary and backup service node groups, use two **distribute** command statements:

```
(config-pmap-c)# distribute service-node-group primary_SNG_name
(config-pmap-c)# distribute service-node-group backup_SNG_name
```

If the service nodes in the primary service node group are not available, the system will use the backup service node group.

Monitor-Load Command

The **monitor-load** command determines which load values should be monitored. When you monitor an application accelerator, the AppNav Controller checks for overload on that application accelerator and does not send new flows to a service node that is overloaded. Flows are sent to a different service node in the service node group.

This command is optional; if you use it, the system monitors the application accelerator indicated by the *application_accelerator_name* parameter. If you do not use this command, the system monitors the TFO accelerator status. If you specify an application accelerator, it replaces the existing monitor-load if one exists.

The supported application accelerators are:

- MS-port-mapper (monitor Microsoft Endpoint Port Mapper load)
- cifs (monitor SMB or CIFS accelerator load)
- http (monitor HTTP accelerator load)
- ica (monitor ICA accelerator load)
- mapi (monitor MAPI accelerator load)
- nfs (monitor NFS accelerator load)
- ssl (monitor SSL accelerator load)
- video (monitor video accelerator load)

Pass-Through Command

Use the **pass-through** command to explicitly indicate that no redirection is to take place. You cannot use the **pass-through** command with the **distribute** or **monitor-load** commands. If you use the **pass-through** command, the system blocks any **distribute** or the **monitor-load** command actions and displays an error message. If you use either the **distribute** or the **monitor-load** command, then the system blocks any **pass-through** command actions.

Configuring Service Contexts

A service context is used to tie the AppNav Controller group, service node group, and AppNav policy map together.

Note If AppNav-XE is managed by WCM, the authentication key in the service-context configuration cannot be modified using the command line interface (CLI).

Use the following command to create a service context:

```
(config)# service-insertion service-context virtual_instance_name/interface_ID
```

interface_ID is a number that is unique across all service contexts. It determines the naming of the automatically-created virtual interfaces called AppNav-Compress*interface_ID* and AppNav-UnCompress*interface_ID*.

Submode commands:

```
(config-service-insertion-context)# [no] appnav-controller-group acg_name
(config-service-insertion-context)# [no] authentication sha1 key authentication_key
(config-service-insertion-context)# [no] service-node-group sng_name
(config-service-insertion-context)# [no] service-policy appnav_policy_name
(config-service-insertion-context)# [no] vrf { name VRF_name | default | global }
(config-service-insertion-context)# [no] enable
```

AppNav Controller Group Command

acg_name is the name of the AppNav Controller group to which this service context belongs. You can only configure one AppNav Controller group for each service context.

Authentication SHA1 Key Command

authentication_key is the shared authentication key used during AppNav Controller to service node registration. You must configure the key identically on service nodes in the same service context. Currently, the AppNav Controller group only supports one authentication key. All service contexts must use authentication or no service contexts can use authentication.

Service Node Group Command

sng_name is the name of one or more service node groups that are part of the service context. The list is used to cross check the ones used in the AppNav policy. Note that the same service node group cannot be shared between two service contexts.

Service Policy Command

appnav_policy_name is the name of the AppNav policy for the service context.

VRF Name Command

VRF_name is the name of the VRF on the LAN interface for the traffic seen by AppNav. You can enter more than one VRF name. You can define up to 64 VRF names, but there is no limit to the number of VRFs supported. VRF global is the same as the other VRF definitions except that it identifies traffic with no VRF. The VRF names are listed one after another such as the following:

```
vrf name v1
vrf name v2
vrf name v3
vrf global
```

If you do not configure a VRF in the service context, the system automatically applies the default configuration of **vrf default**. The purpose of **vrf default** is to match traffic that does not match a configured VRF name or **vrf global**.

The following logic is used to pick the right service context for a packet: The system compares the VRF on the LAN interface traversed by the packet against the VRF names (or **vrf global) that is** configured in the service contexts. If there is a match, the system picks the corresponding service context. If there is no match, the system picks a service context with **vrf default,** if available. If there is no such service context, then the system passes through the packet.

Enabling AppNav Interception

Currently, the only service supported by the AppNav-XE component is WAAS. To enable the AppNav-XE component, identify your WAN interface and then use the **service-insertion** command.

```
(config)# interface interface_name
(config-if)# [no] service-insertion virtual_instance_name
```

Note Both the incoming and outgoing TCP traffic of the interface are subject to AppNav processing according to their VRF and the service policy associated with the service context identified by the VRF.

Configuring the AppNav Service Node Auto Discovery Feature

This section contains the following sub-sections:

Enabling the AppNav Service Node Auto Discovery Feature

Note Configuring the AppNav service node auto discovery feature is only applicable if you do not use the EZConfig program. If you do use the EZConfig program, you do not need to configure the AppNav service node auto discovery feature.

To configure the AppNav service node auto discovery feature, perform the following steps:

Procedure

Step 1 In Cisco IOS-XE, enter the following command.
For the *sng_name* parameter, enter the name of the service node group for which you want to enable the AppNav service node auto discovery feature. Ensure that the WAAS device is in the same subnet as the AppNav-XE component.

```
router(config)# service-insertion service-node-group sng_name
```

Step 2 Enable the feature by entering the following:

```
router(config-service-insertion-sng)# node-discovery enable
```

Step 3 On the WAAS device, enter the following command:

```
WAAS(config)# service-insertion service-node
```

Step 4 Select the interface to use and make sure it is in the same subnet as the AppNav service requestor:

```
WAAS(config)# node-discovery enable GigabitEthernet 0/1
```

Note If interface is not specified, the default is GigabitEthernet0/0

Step 5 Configure and enable the AppNav service node auto discovery feature by entering the following:

```
WAAS(config)# enable
```

Disabling the AppNav Service Node Auto Discovery Feature

You can disable the AppNav service node auto discovery feature by doing either of the following:

- Go to Cisco IOS-XE and disable the entire service node auto discovery feature for the entire system by entering the following:

```
router(config)# service-insertion service-node-group sng
router(config-service-insertion-sng)# no node-discovery enable
```

- Disable the service response feature on the service node, as follows:

```
router(config)# service-insertion service-node
router(config)# no enable
```

Configuring the Container

Note Configuring the container is only applicable if you do not use the EZConfig program. If you use the EZConfig program, you do not need to configure the container.

Copying the ISR-WAAS OVA Package to the Cisco ISR 4451-X

Before you can install ISR-WAAS, you must copy the OVA package to the Cisco ISR 4451-X.

Restriction

- You can only deploy one instance of the ISR-WAAS application on a Cisco ISR 4451-X.

Use the following CLI command to copy the ISR-WAAS OVA package to the Cisco ISR 4451-X:

```
router# copy tftp: harddisk:
```

```
Address or name of remote host []? 1.1.220.10
Source filename []? ISR4451-X-WAAS-eft.ova
Destination filename [ISR4451-X-WAAS-eft.ova]?
Accessing tftp://1.1.220.10/ISR4451-X-WAAS-eft.ova...
Loading ISR4451-X-WAAS-eft.ova from 1.1.220.10 (via GigabitEthernet0):
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
[OK - 1331268265/4096 bytes]

1331268265 bytes copied in 165.024 secs (8067119 bytes/sec)
```

Installing the ISR-WAAS OVA Package

Use the **virtual-service install name** *name* **package** *package* command to install the ISR-WAAS OVA package onto the Cisco ISR 4451-X.

Notes

- It can take two to three minutes to get a package installed successfully.
- Ensure that the output of the **show virtual-service list** command shows the status as “installed” before proceeding further.

```
router# virtual-service install name ISR4451-X-WAAS package
harddisk:ISR4451-X-WAAS-eft.ova
Package "harddisk:/ISR4451-X-WAAS-eft.ova" is currently being installed for virtual
service "ISR-WAAS". Once the install is finished, please activate the VM to run the VM.

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

*Sep 16 00:55:07.588: %IOSXE_VMAN-3-RSPMSGHDLR: Failed to deliver response message:
License Register fails

router# show virtual-service list
Virtual Service List:

Name                    Status             Package Name
------------------------------------------------------------------------
ISR-WAAS                   Installed          ISR4451-X-WAAS-eft.ova
```

Here is the output of the **show virtual-service detail** command at this stage after installation.

```
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
```

Configuring the Virtual Port Group Interface

Configure a virtual port group interface. Use the IP address of the host end of the bridge between the host and virtual service when activated. Here is an example of a part of the running config output:

```
interface VirtualPortGroup4
 ip address 33.1.1.1 255.255.255.0
```

Listing and Selecting a Profile

The output of the **show virtual-service profile name ISR-WAAS** command displays the various profiles that are available in the installed ISR-WAAS OVA package. Using the **detail** keyword with this command displays the resource requirements associated with the profile description.

```
router# show virtual-service profile name ISR-WAAS
```

```
Virtual Service ISR-WAAS profiles:

Name                    Description                                          Allowed
-------------------------------------------------------------------------------------
ISR-WAAS-750               ISR-WAAS profile for 750 TCP connections             Yes
ISR-WAAS-200               ISR-WAAS profile for 200 TCP connections             Yes

router# show virtual-service profile name ISR-WAAS detail
Virtual Service ISR-WAAS Profile Details:

Profile name : ISR-WAAS-750
Description : ISR-WAAS profile for 750 TCP connections

License name : V-WAAS
License version : 1.0
Resource admission : Yes
Resource requirements :
   Disk space    : Not specified
   Memory        : 4096MB
   CPU           : Not specified
   VCPUs         : 2

Profile name : ISR-WAAS-200
Description : ISR-WAAS profile for 200 TCP connections

License name : V-WAAS
License version : 1.0
Resource admission : Yes
Resource requirements :
   Disk space    : Not specified
   Memory        : 2048MB
   CPU           : Not specified
   VCPUs         : 1
```

Creating the ISR-WAAS Application Container

To create the ISR-WAAS application container, associate the virtual service with a profile name and a virtual port group. Configure the IP address of the virtual service end of the bridge that is created between the host and the virtual service when activated. Here is an example:

```
router(config)# virtual-service ISR-WAAS
router(config-virt-serv)# profile ISR-WAAS-750
router(config-virt-serv)# interface VirtualPortGroup31
router(config-virt-serv-intf)# ip address 33.1.1.2
```

Activating the ISR-WAAS Application Container

The following is an example of the commands used to activate the ISR-WAAS application container:

Note Ensure that the output of the **show virtual-service list** command displays the status of the virtual service as “Activated” to confirm that the service has started successfully.

```
router# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.

router(config)# virtual-service ISR-WAAS
```

```
router(config-virt-serv)# activate

router(config-virt-serv)# end

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

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

router#
*Sep 16 01:04:06.196: %VIRT_SERVICE-5-ACTIVATION_STATE: Successfully activated virtual
service ISR-WAAS

router#
*Sep 16 01:04:08.196: %LINK-3-UPDOWN: Interface VirtualPortGroup4, changed state to up
*Sep 16 01:04:09.197: %LINEPROTO-5-UPDOWN: Line protocol on Interface VirtualPortGroup4,
changed state to up

router# show virtual-service list
Virtual Service List:

Name                     Status              Package Name
------------------------------------------------------------------------
ISR-WAAS                    Activated     ISR4451-X-WAAS-eft.ova
```

Verifying the ISR-WAAS Application Container Activation

Issue the following command to verify that the ISR-WAAS application container activated correctly:

```
router# show virtual-service detail name ISR-WAAS
Virtual Service ISR-WAAS 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 :
  Request failed
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_1     ieobc
NIC               dp_1_4      net2

Network interfaces:
MAC address             Attached to interface
------------------------------------------------------
54:0E:00:0B:0C:02       ieobc_1
30:F7:0D:53:C6:1F       VirtualPortGroup4

Guest interface:
Interface: eth0
  ip address: 33.1.1.1 /24

Guest routes:
Address/Mask                          Next Hop                           Intf.
-------------------------------------------------------------------------------
0.0.0.0/0                             33.1.1.2                            eth0


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

Stopping the ISR-WAAS Application

To stop the ISR-WAAS application, enter the following commands and then issue the **virtual-service list** command to ensure that the application is deactivated:

```
router# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.

router(config)# virtual-service ISR-WAAS
router(config-virt-serv)# no activate
router(config-virt-serv)# end

router# show virtual-service list
Virtual Service List:

Name                    Status             Package Name
------------------------------------------------------------------
ISR-WAAS                   Deactivated        ISR4451-X-WAAS-eft.ova
```

Uninstalling the ISR-WAAS Application

When you uninstall the ISR-WAAS application, the system releases all the disk storage that was reserved for this virtual service and any associated saved data is lost.

The following commands demonstrate how to uninstall the ISR-WAAS application. Use the **no activate** command to stop the virtual service before you uninstall it.

Note Stopping the ISR-WAAS application can take some time. The **show virtual-service list** command will show the status as "Deactivating" when the application is de-activating. Ensure that the application is deactivated before you uninstall the application

```
router(config)# virtual-service ISR-WAAS
router(config-virt-serv)# no activate

router# virtual-service uninstall name ISR-WAAS
router#
*Sep 16 01:25:44.996: %VIRT_SERVICE-5-INSTALL_STATE: Successfully uninstalled virtual
service ISR-WAAS
router#

router# show virtual-service list
Virtual Service List:
```

Removing the AppNav-XE Configuration

To remove the AppNav-XE configuration, follow these steps:

Procedure

Step 1 From configuration mode, remove the interception from the WAN interface. Use these CLI commands:

```
router(config)# interface GigabitEthernet0/0/1
router(config-if)# no service-insertion waas
router(config-if)# exit
```

Step 2 Disable the AppNav service context. Use these CLI commands:

```
router(config)# service-insertion service-context  waas/1
router(config-service-insertion-context)# no enable
router(config-service-insertion-context)# exit
```

Step 3 Remove the AppNav service context, service node group, and AppNav controller group. Use these CLI commands:

```
router(config)# no service-insertion service-context  waas/1
router(config)# no service-insertion service-node-group ISR-WAAS-SNG
router(config)# no service-insertion appnav-controller-group ISR-WAAS-SCG
```

Step 4 Remove the AppNav policy map, class map, and access list. Use these CLI commands:

```
router(config)# no policy-map type appnav ISR-WAAS
router(config)# no class-map type appnav match-any ISR-WAAS
router(config)# no ip access-list extended ISR-WAAS
router(config)# end
```

Configuring Port Channel Support for AppNav-XE

You can configure port channel support for AppNav-XE by indicating to the dataplane to swap IP addresses in the packets so that they can be distributed between different port channels.

To do this, use the following command:

```
(config)# service-insertion swap src-ip
(config)# [no] service-insertion swap src-ip
```

This command also enables AppNav-XE to handle packets from the Service Node whose ip addresses are swapped.