

# Carrier Grade IPv6 over Virtualized Services Module (VSM)

This module describes how to implement the Carrier Grade IPv6 (CGv6) over Virtualized Services Module ( VSM).


- Virtualized Services Module, on page 1
- Installing CGv6 on VSM , on page 2
- Uninstalling CGv6 on VSM, on page 6
- Upgrading CGv6 OVA Package, on page 7
- VSM scale numbers, on page 9
- Implementing NAT 64 over VSM, on page 10
- NAT44 and NAT 64 over VSM, on page 13
- Implementing NAT44 or NAT64 over VSM, on page 17
- Traffic Flow Mirroring, on page 63
- Mapping of Address and Port-Encapsulation Mode, on page 74
- Configuring MAP-E, on page 74
- MAP-E on VSM Configuration: Example, on page 89
- Mapping of Address and Port-Translation Mode, on page 90
- Configuring MAP-T, on page 91
- MAP-T Configuration on VSM: Example, on page 109
- Configuration Examples for Implementing the Carrier Grade NAT, on page 109
- Configuring TCP Sequence-Check: Example, on page 116
- Configuring Address and Port-Dependent Filtering: Example, on page 116
- NAT0 Mode Configuration: Example, on page 116
- Configuration of Multiple NetFlow Servers: Example, on page 117
- Configuration of Multiple Syslog Servers: Servers, on page 117
- CGN Sequential Allocation Algorithm, on page 118


# Virtualized Services Module

Virtualized Services Module (VSM) is the next generation service card on the Cisco ASR 9000 Series Aggregation Services Router. The software infrastructure on this card provides a virtual environment and the services run as virtual machines (VM) in this environment. The VMs simulate individual physical computing

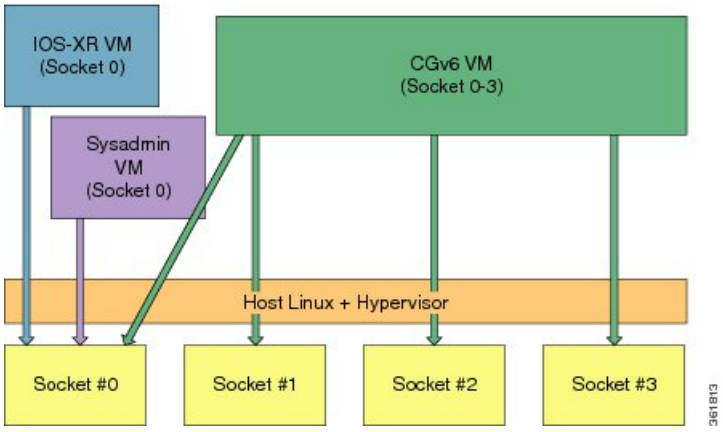
environments over a common hardware. The available hardware resources, like processor, memory, hard disk, and so on, are virtualized and allocated to individual virtual machines by the hypervisor.

**Note** A9K-VSM-500 is not supported on Cisco IOS XR 64-bit image.

### VSM Components

VSM is capable of hosting multiple VMs. It consists of the following components:

- IOS XR VM: This VM is used for managing the routing functions.
- System Admin VM: This VM is used for the system administration
- Application VM: CGv6 is the application VM running on VSM. In the current release, only one CGv6 VM can run at a given time.
- Linux Host and Hypervisor: The routing functions and the system administration functions are run on separate virtual machines (VMs) over a Linux host operating system. The CGv6 VM, along with the other VMs, runs on the top of the KVM hypervisor.




### Features and Considerations

Some of the features and considerations of VSM are:

- The CGv6 application has to run in a VM environment.
- The IOS XR Service Enablement CLIs are needed to create, delete, access, and operate on CGv6 VM.
- The VSM card can co-exist with other LCs including ISM.
- Each NP has 6 NP ports and can send traffic to 24 CGv6 Application processes.
- For each VSM card, a ServiceInfra interface needs to be configured.
- Traffic diversion may be done based on a static route or ACL-based forwarding (ABF).
- In the current release, VSM does not support multiple CGv6 VMs on the same card.

# Installing CGv6 on VSM

The process of installing CGv6 on VSM involves the following:

**Note** If you are performing an upgrade or a downgrade, the CGv6 VM needs to be deactivated first, uninstalled, installed, and then activated. See Uninstalling CGv6 on VSM for information on deactivating the CGv6 VM and uninstalling the CGv6 OVA package.

# Prerequisites

Ensure that you have installed the following images:

- asr9k-mini-px.vm (Base IOS-XR image)
- asr9k-services-infra.pie (VSM Services Infra package)
- asr9k-services-px.pie (CGv6 Services package)
- asr9k-fpd-px.pie (FPGA Image IOS XR package)
- asr9k-vsm-cgv6-<version>.ova (Linux Open Virtual Alliance or OVA package)

# Installing CGv6 OVA Package

The CGv6 Virtual Machine (VM) is provided as an OVA package. Open Virtualization Appliance (OVA) is a single file distribution of the file package. The CGv6 OVA package consists of the following files:

- OVA Profile Descriptor file
- Package version file
- Linux Image file

The process of installation of CGv6 OVA package consists of the following steps:

1. Copy the OVA file from the remote location to the RP disk.

**Note** Once the CGv6 OVA package is copied to RP's disk, you can install it on multiple VSMs on the same chasis.

2. Install CGv6 VM on a specific VSM card.

```
RP/0/RSP0/CPU0:router# virtual-service install name <service/VM name> package <OVA package
 name> node <VSM_location>
```

**Note** The service or VM name can contain only alphanumeric characters (A to Z, a to z, or 0 to 9), or an underscore (_). All other special characters are not allowed. The installation process might take about 7-8 minutes.

3. Check the progress of the installation process by using the show virtual-service list command. Once the installation is complete, the status is changed to Installed.

```
RP/0/RSP0/CPU0:router# show virtual-service list
```

```
Virtual Service List:
         Name     Status         Package Name          Node
         cgn1     Installing     asr9k-vsm-cgv6.ova    0/1/CPU0

RP/0/RSP0/CPU0:NAT#sh virtual-service list

Virtual Service List:
         Name     Status         Package Name          Node
         cgn1     Installed     asr9k-vsm-cgv6.ova    0/1/CPU0
```

# Activating CGv6 VM

The steps to activate the CGv6 VM are as follows:

1. Configure the CGv6 VM and the 12 Gigabit Ethernet (GE) interfaces in the global configuration mode.

```
RP/0/RSP0/CPU0:router(config)# virtual-service cgn123

RP/0/RSP0/CPU0:router(config-virt-service)# vnic interface tenGigE 0/2/1/0

RP/0/RSP0/CPU0:router(config-virt-service)# vnic interface tenGigE 0/2/1/1

RP/0/RSP0/CPU0:router(config-virt-service)# vnic interface tenGigE 0/2/1/2

RP/0/RSP0/CPU0:router(config-virt-service)# vnic interface tenGigE 0/2/1/3

RP/0/RSP0/CPU0:router(config-virt-service)# vnic interface tenGigE 0/2/1/4

RP/0/RSP0/CPU0:router(config-virt-service)# vnic interface tenGigE 0/2/1/5

RP/0/RSP0/CPU0:router(config-virt-service)# vnic interface tenGigE 0/2/1/6

RP/0/RSP0/CPU0:router(config-virt-service)# vnic interface tenGigE 0/2/1/7

RP/0/RSP0/CPU0:router(config-virt-service)# vnic interface tenGigE 0/2/1/8

RP/0/RSP0/CPU0:router(config-virt-service)# vnic interface tenGigE 0/2/1/9

RP/0/RSP0/CPU0:router(config-virt-service)# vnic interface tenGigE 0/2/1/10

RP/0/RSP0/CPU0:router(config-virt-service)# vnic interface tenGigE 0/2/1/11

RP/0/RSP0/CPU0:router(config-virt-service)# commit
```

2. Activate the CGv6 VM.

```
RP/0/RSP0/CPU0:router(config-virt-service)# activate
RP/0/RSP0/CPU0:router(config-virt-service)#commit
```

3. Check the progress of the activation process by using the show virtual-service list command. Once the VM is activated, the status changes to Activated.

```
RP/0/RSP0/CPU0:router# show virtual-service list

Virtual Service List:
         Name     Status         Package Name
         cgn1     Activated     asr9k-vsm-cgv6.ova
```

**Note** Once the VM is activated, it takes about 5 minutes for the CGv6 applications to come up.

4. Configure the ServiceInfra interface.

```
RP/0/RSP0/CPU0:router# configure terminal

RP/0/RSP0/CPU0:router(config)# interface ServiceInfra 1
```

```
RP/0/RSP0/CPU0:router(config-int)# ipv4 address 3.1.1.1 255.255.255.252
RP/0/RSP0/CPU0:router(config-int)# service-location 0/2/CPU0
RP/0/RSP0/CPU0:router(config-int)# commit
```

5. Before you configure NAT44, ensure that the 12 Gigabit Ethernet (GE) interfaces are up. If they are in the shutdown mode, then change their mode by using the **no shut** command.

**Note** In IOS-XR, by default, any interface that is not configured is shut down when the associated line card is reloaded. To prevent this behavior on the VSM TenGigE interface (port), add a minor configuration (such as, *description*) on the interface.

```
RP/0/RSP0/CPU0:router(config)# interface tenGigE 0/2/1/0
RP/0/RSP0/CPU0:router(config-if)# interface tenGigE 0/2/1/1
RP/0/RSP0/CPU0:router(config-if)# interface tenGigE 0/2/1/2
RP/0/RSP0/CPU0:router(config-if)# interface tenGigE 0/2/1/3
RP/0/RSP0/CPU0:router(config-if)# interface tenGigE 0/2/1/4
RP/0/RSP0/CPU0:router(config-if)# interface tenGigE 0/2/1/5
RP/0/RSP0/CPU0:router(config-if)# interface tenGigE 0/2/1/6
RP/0/RSP0/CPU0:router(config-if)# interface tenGigE 0/2/1/7
RP/0/RSP0/CPU0:router(config-if)# interface tenGigE 0/2/1/8
RP/0/RSP0/CPU0:router(config-if)# interface tenGigE 0/2/1/9
RP/0/RSP0/CPU0:router(config-if)# interface tenGigE 0/2/1/10
RP/0/RSP0/CPU0:router(config-if)# interface tenGigE 0/2/1/11
RP/0/RSP0/CPU0:router(config-if)# no shut
RP/0/RSP0/CPU0:router(config-if)# commit
```

**Note** When a virtual service is in *Activating* state and being recovered from a failure event like VSM card reload or server disconnect, and if another failure happens like the VSM card reload or server disconnect, then the virtual-service activation will not be attempted. In such conditions, virtual service is moved into *Activation Failed* state and needs to be recovered manually with the following sequence of operations:

```
RP/0/RSP0/CPU0:router# configure terminal
RP/0/RSP0/CPU0:router(config)# virtual-service cgn123
RP/0/RSP0/CPU0:router(config-virt-service)# no activate
RP/0/RSP0/CPU0:router(config-virt-service)# commit
RP/0/RSP0/CPU0:router(config-virt-service)# activate
RP/0/RSP0/CPU0:router(config-virt-service)# commit
```

**Note** After insertion of VSM card in chassis, you can install using the following CLI.
This command will install and automatically activate the CGN NAT44 services.

```
virtual-service autoActivate name <service_name> package <ova_location> location <VSM_location>
```

Alternatively, you can deactivate and uninstall the service by using the following CLI:

```
virtual-service autounistall name <service_name>
```

Ensure that you reload the VSM line card after uninstalling the VSM line card.

# Uninstalling CGv6 on VSM

The process of uninstalling CGv6 VSM involves the following processes:

- Deactivating CGv6 VM
- Uninstalling CGv6 OVA Package
- Disabling the Service Enablement Feature

### Deactivating CGv6 VM

To de-activate the CGv6 VM, perform the following in the global configuration mode:

```
RP/0/RP0/CPU0:router(config)# virtual-service cgn123
RP/0/RP0/CPU0:router(config-virt-service)# no activate
RP/0/RP0/CPU0:router(config-virt-service)# commit
```

After you have deactivated the CGv6 VM, reload the VSM card.

To remove the CGv6 instance, perform the following in the global configuration mode:

```
RP/0/RP0/CPU0:router(config)# no virtual-service cgn123
RP/0/RP0/CPU0:router(config)# commit
```

### Uninstalling CGv6 OVA Package

To uninstall the CGv6 OVA package, run the following commands in the EXEC mode:

```
RP/0/RSP0/CPU0:router# virtual-service uninstall name cgn123
```

After you have uninstalled the CGv6 OVA package, reload the VSM card before installing any other OVA package.

### Disabling the Service Enablement Feature

**Note** Perform this operation only if you are removing the CGN service completely from the router. If there are multiple VSM line cards in the chassis and you are removing or replacing an individual VSM card, do not perform this operation.

To disable the service enablement feature, run the following commands in the global configuration mode.

```
RP/0/RP0/CPU0:router(config)# no virtual-service enable
RP/0/RP0/CPU0:router(config)# commit
```

# Upgrading CGv6 OVA Package

The CGv6 Virtual Machine (VM) is provided as an OVA package. Open Virtualization Appliance (OVA) is a single file distribution of the file package. The CGv6 OVA package consists of the following files:

- OVA Profile Descriptor file
- Package version file
- Linux Image file

The process of upgrading the CGv6 OVA package consists of the following steps:

1. Saving the virtual service configuration
2. Deactivating CGv6 VM
3. Uninstalling CGv6 OVA Package
4. Upgrading Cisco IOS XR
5. Installing CGv6 on VSM
6. Restoring the virtual service configuration

**Note** Before upgrading or downgrading the CGv6 OVA package on the Active VSM card in HA (high availability) mode, perform a graceful shift of the traffic from Active VSM to Standby VSM. This will ensure that the CGN-related configuration is replicated into a standby card. To perform graceful shift of the traffic, run the `service redundancy failover service-type all preferred-active` <*active-VSM-slot*> command in EXEC mode.

### Saving the Virtual Service Configuration

Before you start upgrading the CGv6 OVA package, save the virtual service configuration and related data.

To see the virtual service configuration, perform the following in the global configuration mode:

```
RP/0/RP0/CPU0:router(config)# show run virtual-service
```

### Deactivating CGv6 VM

To de-activate the CGv6 VM, perform the following in the global configuration mode:

```
RP/0/RP0/CPU0:router(config)#virtual-service cgn123
RP/0/RP0/CPU0:router(config-virt-service)#no activate
RP/0/RP0/CPU0:router(config-virt-service)#commit
```

After you have deactivated the CGv6 VM, reload the VSM card.

To remove the CGv6 instance, perform the following in the global configuration mode:

```
RP/0/RP0/CPU0:router(config)#no virtual-service cgn123
RP/0/RP0/CPU0:router(config)#commit
```

### Uninstalling CGv6 OVA Package

To uninstall the CGv6 OVA package, run the following commands in the EXEC mode:

```
RP/0/RSP0/CPU0:router# virtual-service uninstall name cgn123
```

After you have uninstalled the CGv6 OVA package, reload the VSM card before installing any other OVA package.

### Upgrading Cisco IOS XR

Upgrade Cisco IOS XR. For details refer the *Upgrading and Managing Cisco IOS XR software* chapter in

### Installing the CGv6 OVA Package

The process of installing the CGv6 OVA package consists of the following steps:

1. Copy the OVA file from the remote location to the RP disk.

**Note** Once the CGv6 OVA package is copied to RP's disk, you can install it on multiple VSMs on the same chasis.

2. Install CGv6 VM on a specific VSM card.

```
RP/0/RSP0/CPU0:router#virtual-service install name <service/VM name> package <OVA package
 name> node <VSM_location>
```

**Note** The installation process might take about 7-8 minutes.

3. Check the progress of the installation process by using the show virtual-service list command. Once the installation is complete, the status is changed to Installed.

```
RP/0/RSP0/CPU0:router#sh virtual-service list

Virtual Service List:
         Name     Status        Package Name          Node
         cgn1     Installing    asr9k-vsm-cgv6.ova    0/1/CPU0

RP/0/RSP0/CPU0:NAT#sh virtual-service list

Virtual Service List:
         Name     Status        Package Name          Node
         cgn1     Installed    asr9k-vsm-cgv6.ova    0/1/CPU0
```

### Restoring the Virtual Service Configuration

Restore the virtual service configuration.

CGv6 VM is activated automatically after the virtual service configuration is restored.

Check the progress of the activation process by using the show virtual-service list command. Once the VM is activated, the status changes to Activated.

```
RP/0/RSP0/CPU0:router# show virtual-service list

Virtual Service List:
        Name     Status        Package Name
        cgn1     Activated    asr9k-vsm-cgv6.ova
```

**Note** Once the VM is activated, it takes about 5 minutes for the CGv6 applications to come up.

# VSM scale numbers

ASR9K supports the following VSM scale numbers:

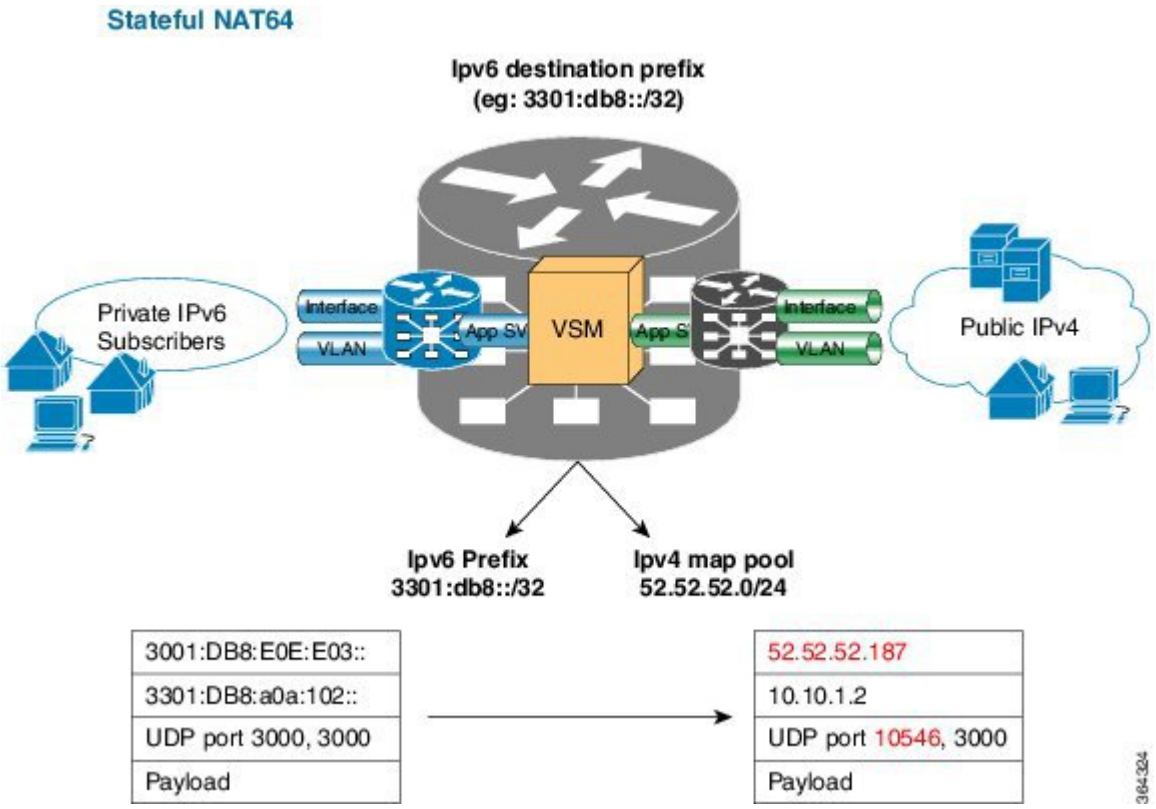
| Parameter Name | Value per VSM | Value per ASR9K Chassis with VSM |
|---|---|---|
| Number of CGN or CGv6 Instances | 1 | 8 |
| Number of Service Infra Interfaces | 1 | 8 |
| Number of Service App interfaces | 512 | |
| Number of NAT44 instances | 1 | 8 |
| Number of Stateful Translation | 80 Millions | |
| Number of NAT session | 80 Million | |
| Number of NAT users | 4 Million | |
| Number of Static Port Forwarding Entries | 6000 | |
| Number of Public IPv4 addresses | 65536 or 16 | |
| Number of VRF per NAT44 instance | 128 (inside) + 128 (outside) | |
| BNG | 32k per np | |
| VRF | 8000 | |
| DS-Lite Sessions | 80 Million | |
| NAT64 Sessions | 80 Million | |

| Parameter Name | Value per VSM | Value per ASR9K Chassis with VSM |
|---|---|---|
| 6RD<br>(ASR 9000 Enhanced Ethernet Line Card is inline with 6RD with an expectation rate of 90 percent.) | | |

**Note** Number of VSM cards per chassis can be adjusted based on the type of chassis and traffic assessment.

# Implementing NAT 64 over VSM

This section explains how NAT64 is implemented over VSM. The figure illustrates the implementation of NAT64 over VSM.



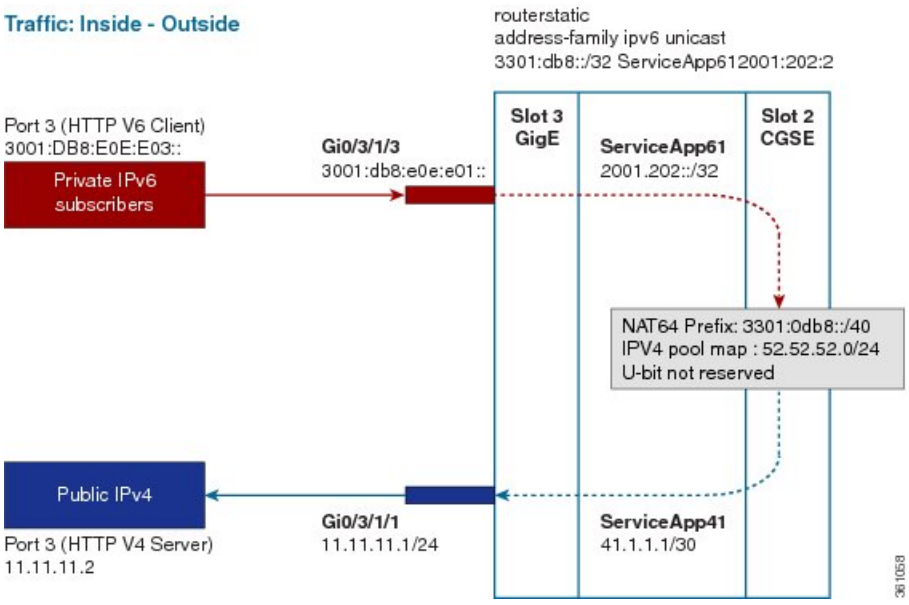

The components of this implementation are as follows:

- Private IP6 subscribers – It denotes a private network.
- Interface/VLAN- It denotes a designated interface or VLAN which is associated with the VRF.

- Inside VRF – It denotes the VRF that handles packets coming from the subscriber network. It is known as inside VRF as it forwards packets from the private network.
- App SVI- It denotes an application interface that forwards the data packet to and from the VSM. The data packet may be sent from another line card through a backplane. Because the VSM card does not have a physical interface, the APP SVI acts as a logical entry into it.

  The inside VRF is bound to an App SVI. There are 2 App SVIs required; one for the inside VRF and the other one for the outside VRF. Each App SVI pair will be associated with a unique "inside VRF" and a unique public IP address pool. The VRF consists of a static route for forwarding packets to App SVI1.
- Outside VRF- It denotes the VRF that handles packets going out to the public network. It is known as outside VRF as it forwards packets from the public network.
- Public IPV4- It denotes a public network.

The following figure illustrates the path of the data packet from a private network to a public network in a NAT64 implementation.

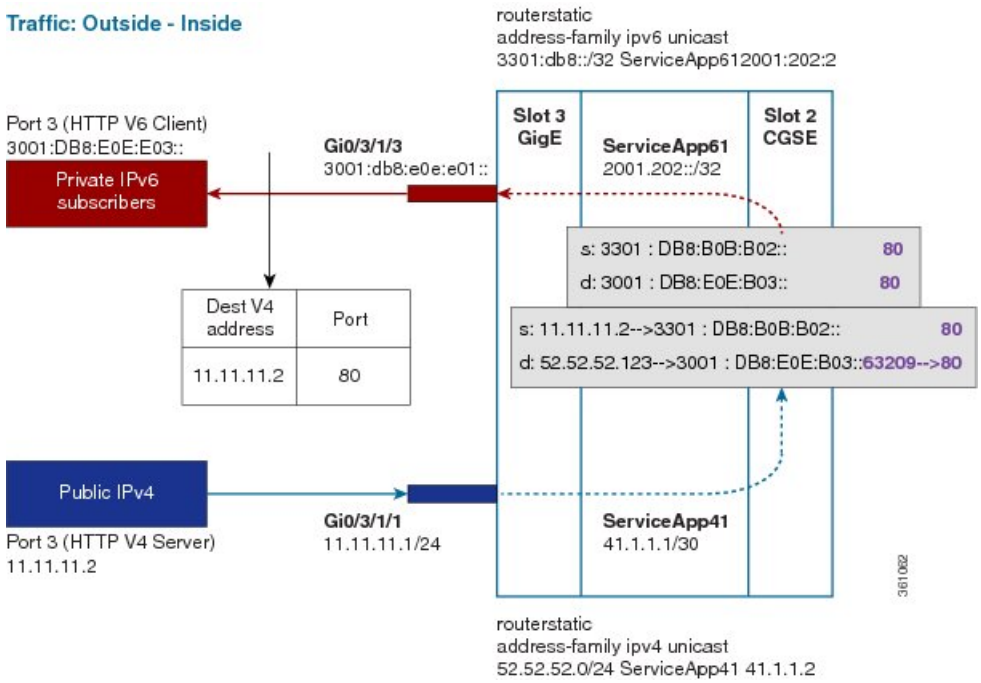



The packet goes through the following steps when it travels from the private network to the public network:

1. In the network shown in this figure, the packet travels from the host A (having the IP address 3001:DB8:E0E:E03::/40) in the private network to host B (having the IP address 11.11.11.2) in the public network. The private address has to be mapped to the public address by NAT64 that is implemented in VSM.
2. The packet enters through the ingress port on the Gigabit Ethernet (GigE) interface at Slot 3.
3. Once the packet reaches the designated interface or VLAN on ASR9K, it is forwarded to the inside VRF either through static routing or ACL-based forwarding (ABL). Based on this routing decision, the packet that needs address translation is determined and is forwarded to the App SVI that is bound to the VRF.
4. The packet is forwarded by AppSVI1 through a default static route. The destination address and the port get translated because of the CGN configuration applied on VSM.

5. The VSM applies NAT64 to the packet and a translation entry is created. The CGN determines the destination address from the FIB Look Up. It pushes the packet to the egress port.
6. The packet is then forwarded to the egress port on the interface through App SVI2. The packet is forwarded by App SVI2 through the default static route. Then the packet is sent to the public network.
7. The packets that do not need the address translation can bypass the App SVI and can be forwarded to the destination through a different static route and a different egress port.

The following figure illustrates the path of the packet coming from the public network to the private network.




The packet goes through the following steps when it travels from the public network to the private network:

1. In the network shown in this figure, the packet travels from the host A (having the IP address 11.11.11.2) in the public network to host B (having the IP address 3001:DB8:E0E:E03::) in the private network. The public address has to be mapped to the private address by NAT64 that is implemented in VSM.
2. The packet enters through the ingress port on the Gigabit Ethernet (GigE) interface at Slot 3.
3. Once the packet reaches the designated interface or VLAN on ASR9K, it is forwarded to the outside VRF either through static routing or ACL-based forwarding (ABL). Based on this routing decision, the packet is forwarded to the App SVI that is bound to the VRF.
4. The packet is forwarded by App SVI2 through a default static route. The destination address and the port are mapped to the translated address.
5. The VSM applies NAT64 to the packet. The CGN determines the destination address from the FIB Look Up. It pushes the packet to the egress port.
6. The packet is then forwarded to the egress port on the interface through App SVI2. Then the packet is sent to the private network through the inside VRF.

7. The packets that do not need the address translation can bypass the App SVI and can be forwarded to the destination through a different static route and a different egress port.

# NAT44 and NAT 64 over VSM

The following are the features that are supported by NAT44 over VSM:

- TCP Sequence Check
- Address and Port-Dependent Filtering
- NAT0 or NAT Bypass Mode
- Static Destination Address Translation
- Multiple NetFlow and Syslog Servers for CGN Logging

The following are the features that are supported by NAT64 over VSM:

- Address and Port-Dependent Filtering
- Multiple NetFlow Servers for CGN Logging

### VSM scale numbers supported in NAT 44

NAT 44 supports the following VSM scale numbers:

| Parameter Name | Value per VSM | Value per ASR9K Chassis with VSM |
|---|---|---|
| Number of CGN or CGv6 Instances | 1 | 8 |
| Number of Service Infra Interfaces | 1 | 8 |
| Number of Service App interfaces | 512 | |
| Number of NAT44 instances | 1 | 8 |
| Number of Stateful Translation | 80 Millions | 80 x Number of Installed VSM cards based on type of chassis. Max 8 number of VSM cards supported per chassis. |
| Number of NAT session | 80 Million | 80 x Number of Installed VSM cards based on type of chassis. Max 8 number of VSM cards supported per chassis. |

| Parameter Name | Value per VSM | Value per ASR9K Chassis with VSM |
|---|---|---|
| Number of NAT users | 4 Million | 4 x Number of Installed VSM cards based on type of chassis. Max 8 number of VSM cards supported per chassis. |
| Number of Static Port Forwarding Entries | 6000 | 6000 x Number of Installed VSM cards based on type of chassis. Max 8 number of VSM cards supported per chassis. |
| Number of Public IPv4 addresses | 65536 or 16 | |
| Number of VRF per NAT44 instance | 128 (inside) + 128 (outside) | |

### VSM Scale numbers supported in NAT 64

NAT 64 supports the following VSM scale number:

| Parameter Name | Value per VSM | Value per ASR9K Chassis with VSM |
|---|---|---|
| NAT64 Sessions | 80 Millions | |

# TCP Sequence Check

In order to overcome security threats to less secure networks, Cisco Virtualized Services Module (VSM) performs TCP sequence check.

A sequence number is a 32-bit number that is included in a packet in a TCP session. The sequence numbers of the incoming packets are stored in the translation or session entry. If a packet's sequence number does not match the expected sequence number, then the packet is dropped . In this way, the networks can be secured from spoofed packets.

You can perform these TCP sequence checks by using the sequence-check command. An optional keyword, diff-window, has been provided for a user to define and configure the accepted expected range of sequence numbers. But it is recommended that the user does not specify this range and instead allows the router to compute the range for each TCP session based on the client-server negotiation.

Two counters are configured for the TCP sequence checks:

- Out-to-In packets counter: This counter keeps a count of the packets whose sequence numbers did not match the expected range. But yet these packets are translated and forwarded because TCP sequence check has not been configured.
- Dropped packets counter: This counter keeps a count of the packets that were dropped because of the TCP sequence check.

The counters are displayed by using the show cgn nat44 counters command.

# Address and Port-Dependent Filtering

Currently, CGN on VSM implements the following by default:

- Endpoint-Independent Mapping: This mapping process reuses the port mapping for subsequent packets that are sent from the same internal IP address and port to any external IP address and port.
- Endpoint-Independent Filtering: This filtering process filters out only packets that are not destined to the internal address and port regardless of the external IP address and port source.

In such a configuration, by knowing the translated IP address and the port of a private host, any malicious host in a public network can initiate packet floods to that private host. In order to prevent such attacks, the address and port-dependent filtering feature has to be enabled by using the **filter-policy** command. The user can disable the filtering based on port by using the **ignore-port** keyword with this command.

Two counters are configured for the address and port-dependent filtering:

- Total number of sessions created due to Out2In packets: This counter keeps a count of the sessions that were created by the packets coming from outside.
- Number of Out2In drops due to end point filtering: This counter keeps a count of the packets that were dropped because of the endpoint filtering.

The counters are displayed by using the **show cgn nat44 counters** command.

# NAT0 or NAT Bypass Mode

For some subscribers, a service provider may want to provide public addresses directly. Hence the Network Address Translation (NAT) is not required for these IP addresses. But at the same time, services like endpoint dependent filtering and TCP sequence check are required so that the subscribers do not receive any unwanted traffic from the Internet. In such cases, NAT0 or NAT Bypass mode is supported per inside-VRF.

The NAT0 mode along with the TCP sequence check and endpoint dependent filtering offers protection to the subscribers from Outside-to-Inside DoS attacks.

In this mode, when a subscriber sends a packet, an entry is made in the database. When the packet comes from the Outside-to-Inside direction, the entry is checked. But no translation occurs. The packet is allowed only if the entry exists.

NAT0 mode can be enabled by using the **nat-mode no-nat** command.

### Considerations

Some of the considerations regarding the NAT0 mode are as follows:

- Static port forwarding is applicable in this mode.
- Bulk port allocation is not applicable in this mode.
- There is no need for the public address pool to be configured as the translation does not happen.
- If you have not initiated any traffic, then you will not receive any packet. Hence bandwidth can be saved and controlled by the public IP user.

# Static Destination Address Translation

In static destination address translation, the VSM translates the destination IP address along with the source IP address. When a packet goes through NAT44, the source IP address translation happens. If the static destination address translation is enabled, then the destination IP address translation also takes place. For the source address translation, the IP address is taken from the public address pool. For the destination address translation, the user needs to provide a 1:1 mapping of the addresses in a .csv file.

To configure static destination address translation, use the **static-mapping -file direction** command. To delete the existing configuration use the **no static-mapping-file direction** command. If you want to modify the existing configuration, you can edit the existing .csv file and upload it back to the disk.

### About the Static Mapping Configuration File

As mentioned before, the static mapping configuration file is in .csv format. The 1:1 mapping of the private addresses and the public addresses for the static destination address translation is provided in the .csv file. If this file becomes corrupt, then the destination address will not get translated. This file must be stored on the local disk.

The 1:1 mapping of the addresses must be in the following format in the .csv file:

```
13.1.1.0, 12.1.1.0
13.1.1.1, 12.1.1.1
13.1.1.2, 12.1.1.2
13.1.1.3, 12.1.1.3
13.1.1.4, 12.1.1.4
13.1.1.5, 12.1.1.5
```



**Note** The IP address, 13.1.1.0, is known as the premap IP address and the IP address, 12.1.1.0, as the postmap IP address

### Considerations

Some of the considerations regarding the static destination address translation are as follows:

- The static destination address translation feature currently supports static destination NAT in the Inside-to-Outside direction and static source NAT in the Outside-to-Inside direction.
- Currently, the static destination address translation supports NAT44 only.
- The maximum value of entries for static mapping of addresses per VSM is 50K.

### High Availability Support for the Route Switch Processor (RSP) Switchover

To ensure high availability during the RSP switchover, the user should manually copy the static mapping configuration file on both active RSP as well as the standby RSP.

### VSM High Availability Support

The high availability of VSM is supported by default.

# Multiple NetFlow and Syslog Servers for CGN Logging

Cisco IOS XR supports High Availability for NAT44 through multiple NetFlow or Syslog servers. Starting from Release 5.2.0, you can configure up to 8 additional NetFlow or Syslog Servers for a single inside VRF instance.

**Note** In an inside VRF with multiple servers configured, the NetFlow logging policy will be enabled on all servers if it is enabled for one or more servers inside the VRF.

**Note** In an inside VRF with multiple servers configured, the Syslog logging policy will be enabled on all servers if it is enabled for one or more servers inside the VRF.

# Implementing NAT44 or NAT64 over VSM

This section explains the implementation of NAT44 or NAT64 on VSM.

# Configuring NAT44 over VSM

Perform the following tasks to configure NAT44 on VSM:

1. Configuring a NAT44 Instance, on page 17
2. Configuring the Application Service Virtual Interface (NAT44), on page 19
3. #unique_43

## Configuring a NAT44 Instance

Perform this task to configure a NAT44 instance.

**Note** The system does not support deleting VRF on live traffic in the following scenarios:

- If you are in the global configuration mode.
- If you are within the CGN instance.
- If you are in the static route table.

### SUMMARY STEPS

1. **configure**
2. **service cgn nat44***instance-name*
3. **service-location preferred-active** *VSM location*

4. **service-type nat44 nat1**
5. **end** or **commit**

## DETAILED STEPS

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **service cgn nat44***instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# service cgn cgn1`<br>`RP/0/RP0/CPU0:router(config-cgn)#` | Configures the instance named cgn1 for the CGv6 NAT44 application and enters CGv6 configuration mode. |
| **Step 3** | **service-location preferred-active** *VSM location*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# service-location preferred-active 0/3/CPU0` | Configures the NAT preferred active VSM location. |
| **Step 4** | **service-type nat44 nat1**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# service-type nat44 nat1` | Configures the service type keyword definition for CGv6 NAT44 application. |
| **Step 5** | **end** or **commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# end`<br>`or`<br>`RP/0/RP0/CPU0:router(config-cgn)# commit` | Saves configuration changes.<br>• When you issue the end command, the system prompts you to commit changes:<br>`Uncommitted changes found, commit them before exiting (yes/no/cancel)?`<br>`[cancel]:`<br>  • Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>  • Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>  • Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

# Configuring the Application Service Virtual Interface (NAT44)

Perform this task to configure the application service virtual interface (SVI) to forward data traffic.

## SUMMARY STEPS

1. **configure**
2. **interface ServiceApp** *value*
3. **service cgn** *instance-name* **service-type** *nat44*
4. **vrf** *vrf-name*
5. **end** or **commit**

## DETAILED STEPS

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **interface ServiceApp** *value*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# interface ServiceApp 1`<br>`RP/0/RP0/CPU0:router(config-if)#` | Configures the application SVI as 1 and enters interface configuration mode. |
| **Step 3** | **service cgn** *instance-name* **service-type** *nat44*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-if)# service cgn cgn1` | Configures the instance named cgn1 for the CGN application and enters CGN configuration mode. |
| **Step 4** | **vrf** *vrf-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-if)# vrf insidevrf1` | Configures the VPN routing and forwarding (VRF) for the Service Application interface |
| **Step 5** | **end** or **commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-if)# end`<br>`or`<br>`RP/0/RP0/CPU0:router(config-if)# commit` | Saves configuration changes.<br>• When you issue the **end** command, the system prompts you to commit changes:<br>*Uncommitted changes found, commit them before exiting (yes/no/cancel)?*<br>*[cancel]:*<br>• Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. |

| | Command or Action | Purpose |
|---|---|---|
| | | • Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>• Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

## Configuring an Inside and Outside Address Pool Map (NAT44)

Perform this task to configure an inside and outside address pool map with the following scenarios:



**Note** Do not configure multiple outside address-pools to be mapped to a single inside-vrf. If you have multiple outside address-pools to be mapped, then create multiple inside-vrfs and map each outside address-pool to a single inside-vrf inside the NAT44 configuration.

- The designated address pool is used for CNAT.
- One inside VRF is mapped to only one outside VRF.
- Multiple non-overlapping address pools can be used in a specified outside VRF mapped to different inside VRF.
- Max Outside public pool per CGSE/CGN instance is 64 K or 65536 addresses. That is, if a /16 address pool is mapped, then we cannot map any other pool to that particular CGSE.
- Multiple inside vrf cannot be mapped to same outside address pool.
- While Mapping Outside Pool minimum value for prefix is 16 and maximum value is 26.

### SUMMARY STEPS

1. **configure**
2. **service cgn** *instance-name*
3. **service-type nat44 nat1**
4. **inside-vrf** *vrf-name*
5. **map** [**outside-vrf** *outside-vrf-name*] **address-pool** *address/prefix*
6. **end** or **commit**

## DETAILED STEPS

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`Router# configure` | Enters global configuration mode. |
| **Step 2** | **service cgn** *instance-name*<br>**Example:**<br>`Router(config)# service cgn cgn1`<br>`RP/0/RP0/CPU0:router(config-cgn)#` | Configures the instance named cgn1 for the CGN application and enters CGN configuration mode. |
| **Step 3** | **service-type nat44 nat1**<br>**Example:**<br>`Router(config-cgn)# service-type nat44 nat1` | Configures the service type keyword definition for CGN NAT44 application. |
| **Step 4** | **inside-vrf** *vrf-name*<br>**Example:**<br>`Router(config-cgn-nat44)# inside-vrf insidevrf1`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)#` | Configures an inside VRF named insidevrf1 and enters CGN inside VRF configuration mode. |
| **Step 5** | **map** [**outside-vrf** *outside-vrf-name*] **address-pool** *address/prefix*<br>**Example:**<br>`Router(config-cgn-invrf)# map outside-vrf outside vrf1 address-pool 10.10.0.0/16`<br>`or`<br>`Router(config-cgn-invrf)# map address-pool 100.1.0.0/16` | Configures an inside VRF to an outside VRF and address pool mapping. |
| **Step 6** | **end** or **commit**<br>**Example:**<br>`Router(config-cgn-invrf-afi)# end`<br>`or`<br>`Router(config-cgn-invrf-afi)# commit` | Saves configuration changes.<br>• When you issue the **end** command, the system prompts you to commit changes:<br>*Uncommitted changes found, commit them before exiting (yes/no/cancel)?*<br>*[cancel]:*<br>• Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>• Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. |

| | Command or Action | Purpose |
|---|---|---|
| | | • Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

# Predefined NAT

In classic NAT, the process of mapping a private IP to a public IP or a private port to an outside port is random. Therefore, it becomes difficult to track the subscribers using an IP and a port at a given time. Predefined NAT avoids this random process by mapping a private IP address to a range of ports associated with the corresponding public IP address. This is done through an algorithm that helps the user to recognize a private IP address without having to refer to the massive CGN logs. The address and port translation is done in accordance with the algorithm.

In a predefined NAT configuration, if you want to trace a subscriber's private IP address from a public IP address and the associated port, perform the following steps:

- Whenever NAT is configured on a router or when there is a change in the existing configuration, use the following command to get the complete mapping information of private to public users:

  **show cgn nat44** *instance-name* **mapping** {**inside-address | outside-address**} **inside-vrf***vrf-instance* **start-addr** *start address* [ **end-addr***end address*]

  In the above command, specify the lowest address of the configured public IP pool as start address and the highest address of the pool as end address. This command dumps all the mapping for each private IP, the translated public IP, and port range. It is recommended that you divert this output in to a file and save it for future reference. Save this output to separate files each time you change the NAT44 configuration parameters and note down the time at which the changes were made and the corresponding file name.

- Whenever there is a request to trace back the subscriber's private IP address, access the right file based on the timestamp provided. The file will have the public IP and port range to which the specified port belongs. The private IP address in that row will help identify the subscriber.

# Considerations and Limitations of Predefined NAT

The considerations and the limitations of the predefined mode for NAT 44 are as follows:

- You can configure the predefined mode for each of the inside VRF instance.
- A new parameter, private address range, has been added to the NAT 44 configuration for the predefined mode. You can specify a minimum of one private address range to a maximum of eight private address ranges. Ensure that you specify atleast one private address range because the available public addresses and the associated ports are mapped to the private addreses specified in this range. If the incoming packet has an address that is outside the private address range, then the packet is discarded. Ensure that the sum of all addresses should not exceed one million across all predefined mode-enabled VRFs.

- The Bulk Port Allocation configuration is not available in the predefined mode. If you try to configure Bulk Port Allocation on an inside VRF that has the predefined mode enabled, the configuration is rejected during verification.
- The port-preservation option is not available in the predefined mode.
- The global port limit parameter is not available for the predefined mode. Even though you will be allowed to configure the global port limit, the inside VRF, which has predefined mode enabled, ignores that port limit and uses the port limit configured by the algorithm.
- If you turn the predefined mode on or off for an inside VRF during the active translations, all the translations on that VRF are deleted.
- If a request for configuring static port on a private address that is not in the address range is made, the request is rejected.
- Ensure that you configure NetFlow or syslog only if it is very much required.
- Any configuration change that results in changes in mapping deletes the existing translations. Therefore, ensure that you record such configuration changes. You might need this information to trace the port usage by a subscriber.
- Ensure uniform port allocation uniform for all subscribers.

# Configuring the Predefined Mode for NAT44

Perform these tasks to configure the predefined mode for NAT44.

### SUMMARY STEPS

1. **configure**
2. **service cgn** *instance-name*
3. **service-type nat44** *nat1*
4. **inside-vrf** *vrf-name*
5. **map address-pool** *address/prefix*
6. **nat-mode**
7. **predefined** *ipaddress/prefix*
8. **end** or **commit**

### DETAILED STEPS

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **service cgn** *instance-name*<br>**Example:** | Configures the instance named cgn1 for the CGN application and enters CGN configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| | `RP/0/RP0/CPU0:router(config)# service cgn cgn1`<br>`RP/0/RP0/CPU0:router(config-cgn)#` | |
| **Step 3** | **service-type nat44** *nat1*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# service-type nat44 nat1` | Configures the service type keyword definition for CGN NAT44 application. |
| **Step 4** | **inside-vrf** *vrf-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-nat44)# inside-vrf insidevrf1`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)#` | Configures an inside VRF named *insidevrf1* and enters CGv6 inside VRF configuration mode. |
| **Step 5** | **map address-pool** *address/prefix*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)# map address-pool 10.10.0.0/16` | Maps an inside VRF to an outside VRF and address pool mapping. |
| **Step 6** | **nat-mode**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)# nat-mode` | Specifies the predefined mode for NAT44. |
| **Step 7** | **predefined** *ipaddress/prefix*<br>**Example:**<br>`RP/0/RP1/CPU0:router(config-cgn-invrf-natmode)# predefined private-pool 192.1.106.0/24`<br>`RP/0/RP1/CPU0:router(config-cgn-invrf-natmode)# predefined private-pool 192.1.107.0/26`<br>`RP/0/RP1/CPU0:router(config-cgn-invrf-natmode)# predefined private-pool 192.1.107.128/26` | Specifies the private address range for the predefined mode. You can specify a minimum of one address range to eight addess ranges. |
| **Step 8** | **end** or **commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-natmode)# end`<br>`or`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-natmode)# commit` | Saves configuration changes.<br>• When you issue the **end** command, the system prompts you to commit changes:<br>*Uncommitted changes found, commit them before exiting (yes/no/cancel)?*<br>*[cancel]:*<br>• Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>• Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. |

| | Command or Action | Purpose |
|---|---|---|
| | | • Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

# Configuring NAT64 over VSM

Perform the following tasks to configure NAT64 on VSM:

1. Configuring a NAT64 Instance, on page 25
2. Configuring the Application Service Virtual Interface (NAT64), on page 26
3. Configuring an Inside and Outside Address Pool Map (NAT64), on page 28

## Configuring a NAT64 Instance

Perform this task to configure a NAT64 instance.

### SUMMARY STEPS

1. **configure**
2. **service cgn** *instance-name*
3. **service-location preferred-active** *VSM location*
4. **service-type nat64 stateful** *instance-name*
5. **end** or **commit**

### DETAILED STEPS

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **service cgn** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# service cgn cgn1`<br>`RP/0/RP0/CPU0:router(config-cgn)#` | Configures the instance named cgn1 for the CGv6 NAT64 application and enters CGv6 configuration mode. |
| **Step 3** | **service-location preferred-active** *VSM location*<br>**Example:** | Configures the NAT preferred active VSM location. |

| | Command or Action | Purpose |
|---|---|---|
| | `RP/0/RP0/CPU0:router(config-cgn)# service-location preferred-active 0/3/CPU0` | |
| **Step 4** | **service-type nat64 stateful** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# service-type nat64 stateful nat64-inst` | Configures the service type keyword definition for CGv6 Stateful NAT64 application. |
| **Step 5** | **end** or **commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-nat64-stateful)# end`<br>or<br>`RP/0/RP0/CPU0:router(config-cgn-nat64-stateful)# commit` | Saves configuration changes.<br>• When you issue the end command, the system prompts you to commit changes:<br>`Uncommitted changes found, commit them before exiting (yes/no/cancel)? [cancel]:`<br>  • Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>  • Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>  • Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

## Configuring the Application Service Virtual Interface (NAT64)

Perform this task to configure the application service virtual interface (SVI) to forward data traffic.

### SUMMARY STEPS

1. **configure**
2. **interface ServiceApp** *value*
3. **vrf** *vrf-name*
4. **ipv6 address***address*
5. **service cgn** *instance-name* **service-type nat64 stateful**
6. **commit**
7. **interface ServiceApp** *value*
8. **vrf** *vrf-name*
9. **ipv4 address** *address*
10. **service cgn** *instance-name* **service-type nat64 stateful**
11. **end** or **commit**

## DETAILED STEPS

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **interface ServiceApp** *value*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# interface ServiceApp 1`<br>`RP/0/RP0/CPU0:router(config-if)#` | Configures the application SVI as 1 and enters interface configuration mode. |
| **Step 3** | **vrf** *vrf-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-if)# vrf insidevrf1` | Configures the VPN routing and forwarding (VRF) for the Service Application interface |
| **Step 4** | **ipv6 address***address*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-if)# ipv6 address 5001::5555/96` | Configures the IPv6 address. |
| **Step 5** | **service cgn** *instance-name* **service-type nat64 stateful**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# service cgn cgn1 service-type nat64 stateful`<br>`RP/0/RP0/CPU0:router(config-cgn-nat64-stateful)#` | Configures the instance named cgn1 and the service type keyword definition for CGv6 Stateful NAT64 application. |
| **Step 6** | **commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-nat64-stateful)# commit` | Saves configuration changes. |
| **Step 7** | **interface ServiceApp** *value*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# interface ServiceApp 2`<br>`RP/0/RP0/CPU0:router(config-if)#` | Configures the application SVI as 2 and enters interface configuration mode. |
| **Step 8** | **vrf** *vrf-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-if)# vrf outsidevrf1` | Configures the VPN routing and forwarding (VRF) for the Service Application interface |
| **Step 9** | **ipv4 address** *address*<br>**Example:** | Configures the IPv4 address. |

| | Command or Action | Purpose |
|---|---|---|
| | `RP/0/RP0/CPU0:router(config-if)# ipv4 address 211.1.1.1 255.255.255.0` | |
| **Step 10** | **service cgn** *instance-name* **service-type nat64 stateful**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# service cgn cgn1 service-type nat64 stateful`<br>`RP/0/RP0/CPU0:router(config-cgn-nat64-stateful)#` | Configures the instance named cgn1 and the service type keyword definition for CGv6 Stateful NAT64 application. |
| **Step 11** | **end** or **commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-nat64-stateful)# end`<br>or<br>`RP/0/RP0/CPU0:router(config-cgn-nat64-stateful)# commit` | Saves configuration changes.<br>• When you issue the **end** command, the system prompts you to commit changes:<br>*Uncommitted changes found, commit them before exiting (yes/no/cancel)?*<br>*[cancel]:*<br>• Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>• Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>• Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

## Configuring an Inside and Outside Address Pool Map (NAT64)

Perform this task to configure an inside and outside address pool map with the following scenarios:

- The designated address pool is used for CNAT.
- One inside VRF is mapped to only one outside VRF.
- Multiple non-overlapping address pools (up to a maximum of eight) can be used in a specified outside VRF mapped to different inside VRF.
- Max Outside public pool per CGSE/CGN instance is 64 K or 65536 addresses. That is, if a /16 address pool is mapped, then we cannot map any other pool to that particular CGSE.
- Multiple inside vrf cannot be mapped to same outside address pool.
- While Mapping Outside Pool Minimum value for prefix is 16 and maximum value is 26.

## SUMMARY STEPS

1. **configure**
2. **service cgn** *instance-name*
3. **service-type nat64 stateful** *instance-name*
4. **ipv6-prefix** *address/prefix*
5. **ipv4-address-pool** *address/prefix*
6. **dynamic-port-range start** *port-number*
7. **address-family ipv4**
8. **interface ServiceApp** *value*
9. **address-family ipv6**
10. **interface ServiceApp** *value*
11. **end** or **commit**

## DETAILED STEPS

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **service cgn** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)#service cgn cgn1` | Configures the instance named cgn1 for the CGv6 application and enters the CGv6 configuration mode. |
| **Step 3** | **service-type nat64 stateful** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# service-type nat64 stateful nat64_1` | Configures the service type keyword definition for CGv6 Stateful NAT64 application. |
| **Step 4** | **ipv6-prefix** *address/prefix*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-nat64-stateful)# ipv6-prefix 3301:db8:1::/96` | Configures the IPv6 prefix that is used to convert destination IPv6 address to an external destination IPv4 address. |
| **Step 5** | **ipv4-address-pool** *address/prefix*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-nat64-stateful)# ipv4 address-pool 62.1.0.0/18` | Defines the IPv4 address pool. |
| **Step 6** | **dynamic-port-range start** *port-number*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-nat64-stateful)#dynamic-port-range start 1` | Configures the port range from 1 to 65535. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 7** | **address-family ipv4**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-nat64-stateful)# address-family ipv4` | Enters the address family IPv4 configuration mode. |
| **Step 8** | **interface ServiceApp** *value*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-nat64-stful-afi)# interface ServiceApp2` | Specifies the ServiceApp on which IPv4 traffic enters and leaves. |
| **Step 9** | **address-family ipv6**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-nat64-stful-afi)# address-family ipv6` | Enters the address family IPv6 configuration mode. |
| **Step 10** | **interface ServiceApp** *value*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-nat64-stful-afi)# interface ServiceApp1` | Specifies the ServiceApp on which IPv6 traffic enters and leaves. |
| **Step 11** | **end** or **commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-nat64-stful-afi)# commit` | Saves configuration changes.<br>• When you issue the **end** command, the system prompts you to commit changes:<br>*Uncommitted changes found, commit them before exiting (yes/no/cancel)?*<br>*[cancel]:*<br>• Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>• Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>• Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

# Policy Functions

## Configuring Port Limit per Subscriber

Perform this task to restrict the number of ports used by an IPv6 address.

### SUMMARY STEPS

1. **configure**
2. **service cgn** *instance-name*
3. **service-type nat64 stateful** *instance-name*
4. **portlimit** *value*
5. **end** or **commit**

### DETAILED STEPS

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **service cgn** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# service cgn cgn1`<br>`RP/0/RP0/CPU0:router(config-cgn)#` | Configures the instance named cgn1 for the CGv6 application and enters CGv6 configuration mode. |
| **Step 3** | **service-type nat64 stateful** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# service-type nat64 stateful nat64-inst`<br>`RP/0/RP0/CPU0:router(config-cgn-nat64-stateful)#` | Configures the service type keyword definition for CGv6 Stateful NAT64 application. |
| **Step 4** | **portlimit** *value*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-nat64-stateful)#portlimit 66`<br>`RP/0/RP0/CPU0:router(config-cgn-nat64-stateful)` | Configures a value to restrict the number of ports used by an IPv6 address. |
| **Step 5** | **end** or **commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-nat64-stateful)# end`<br>`or`<br>`RP/0/RP0/CPU0:router(config-cgn-nat64-stateful)# commit` | Saves configuration changes.<br>• When you issue the **end** command, the system prompts you to commit changes:<br>*Uncommitted changes found, commit them before exiting (yes/no/cancel)?*<br>*[cancel]:* |

| | Command or Action | Purpose |
|---|---|---|
| | | • Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>• Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>• Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

## Configuring the Timeout Value for ICMP, TCP and UDP Sessions

Perform this task to configure the timeout value for ICMP, TCP or UDP sessions for a Dual Stack Lite (DS Lite) instance:

### SUMMARY STEPS

1. **configure**
2. **service cgn** *instance-name*
3. **service-type ds-lite** *instance-name*
4. **protocol tcp session {active | initial} timeout** *value* or **protocol {icmp | udp} timeout** ***value***
5. **end** or **commit**

### DETAILED STEPS

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **service cgn** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# service cgn cgn1`<br>`RP/0/RP0/CPU0:router(config-cgn)#` | Configures the instance named cgn1 for the CGv6 application and enters CGv6 configuration mode. |
| **Step 3** | **service-type ds-lite** *instance-name*<br>**Example:** | Configures the service type keyword definition for CGv6 DS-Lite application. |

| | Command or Action | Purpose |
|---|---|---|
| | `RP/0/RP0/CPU0:router(config-cgn)# service-type ds-lite ds-lite-inst`<br>`RP/0/RP0/CPU0:router(config-cgn-ds-lite)#` | |
| **Step 4** | **protocol tcp session {active \| initial} timeout** *value* or **protocol {icmp \| udp} timeout** ***value*** <br> **Example:** <br> `RP/0/RP0/CPU0:router(config-cgn-ds-lite)#protocol tcp session active timeout 90`<br>`or`<br>`protocol icmp timeout 90`<br>`RP/0/RP0/CPU0:router(config-cgn-ds-lite)` | Configures the initial and active session timeout values for TCP.<br>Configures the timeout value in seconds for ICMP and UDP. |
| **Step 5** | **end** or **commit** <br> **Example:** <br> `RP/0/RP0/CPU0:router(config-cgn-ds-lite)# end`<br>`or`<br>`RP/0/RP0/CPU0:router(config-cgn-ds-lite)# commit` | Saves configuration changes.<br>• When you issue the **end** command, the system prompts you to commit changes:<br>*Uncommitted changes found, commit them before exiting (yes/no/cancel)?*<br>*[cancel]:*<br>• Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>• Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>• Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

## FTP-ALG

CGN supports both passive and active FTP. FTP clients are supported with inside (private) address and servers with outside (public) addresses. Passive FTP is provided by the basic NAT function. Active FTP is used with the ALG.

## RTSP-ALG

CGN supports the Real Time Streaming Protocol (RTSP), an application-level protocol for control over the delivery of data with real-time properties. RTSP provides an extensible framework to enable controlled, on-demand delivery of real-time data, such as audio and video. Sources of data can include both live data feeds and stored clips.

## PPTP-ALG

PPTP is a network protocol that enables secure transfer of data from a remote client to a private enterprise server by creating a Virtual Private Network (VPN). It is used to provide IP security at the network layer. PPTP uses a control channel over TCP and a GRE tunnel operating to encapsulate PPP packets.

PPTP-ALG is a CGN solution that allows traffic from all clients through a single PPTP tunnel.

A PPTP tunnel is instantiated on the TCP port. This TCP connection is then used to initiate and manage a second GRE tunnel to the same peer.

PPTP uses an access controller and network server to establish a connection.

PPTP Access Controller (PAC)

A device attached to one or more PSTN or ISDN lines capable of PPP operation and handling the PPTP protocol. It terminates the PPTP tunnel and provides VPN connectivity to a remote client.

PPTP Network Server (PNS)

A device which provides the interface between the Point-to-Point Protocol (encapsulated in the PPTP protocol) and a LAN or WAN. The PNS uses the PPTP protocol to support tunneling between a PPTP PAC and the PNS. It requests to establish a VPN connectivity using PPTP tunnel.

Control Connection

A control connection is established between a PAC and a PNS for TCP.

Tunnel

A tunnel carries GRE encapsulated PPP datagrams between a PAC and a PNS



**Note** Active FTP, PPTP ALG, and RTSP ALG are supported on NAT44 applications. Active FTP and RTSP ALG are supported on DS-Lite applications.

## TCP Maximum Segment Size Adjustment

When a host initiates a TCP session with a server, the host negotiates the IP segment size by using the maximum segment size (MSS) option. The value of the MSS option is determined by the maximum transmission unit (MTU) that is configured on the host.

## Static Port Forwarding

Static port forwarding helps in associating a private IP address and port with a statically allocated public IP and port. After you have configured static port forwarding, this association remains intact and does not get removed due to timeouts until the CGSE is rebooted. In case of redundant CGSE cards, it remains intact until both of the CGSEs are reloaded together or the router is reloaded. There are remote chances that after a reboot, this association might change. This feature helps in cases where server applications running on the private network needs access from public internet.



**Note** NAT64 on VSM does not support static port forwarding.

# Configuring Dynamic Port Range

Perform this task to configure a dynamic port range.

### SUMMARY STEPS

1. **configure**
2. **service cgn** *instance-name*
3. **service-type nat64 stateful** *instance-name*
4. **dynamic-port-range start** *port-number*
5. **end**or **commit**

### DETAILED STEPS

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **service cgn** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# service cgn cgn1`<br>`RP/0/RP0/CPU0:router(config-cgn)#` | Configures the instance for the CGv6 application and enters CGv6 configuration mode. |
| **Step 3** | **service-type nat64 stateful** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# service-type nat64 stateful nat64-inst`<br>`RP/0/RP0/CPU0:router(config-cgn-nat64-stateful)` | Configures the service type keyword definition for CGv6 Stateful NAT64 application. |
| **Step 4** | **dynamic-port-range start** *port-number*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-nat64-stateful)#dynamic-port-range start 66`<br>`RP/0/RP0/CPU0:router(config-cgn-nat64-stateful)` | Configures the port range from 1 to 65535. |
| **Step 5** | **end**or **commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-nat64-stateful)# end`<br>`or`<br>`RP/0/RP0/CPU0:router(config-cgn-nat64-stateful)# commit` | Saves configuration changes.<br>• When you issue the end command, the system prompts you to commit changes:<br>`Uncommitted changes found, commit them before exiting (yes/no/cancel)?`<br>`[cancel]:`<br>• Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. |

| | Command or Action | Purpose |
|---|---|---|
| | | • Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>• Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

# (Only NAT44) Configuring One-to-One Mapping

Perform this task to configure one-to-one mapping for private addresses.

### SUMMARY STEPS

1. **configure**
2. **service cgn** *instance-name*
3. **service-type nat44 nat1**
4. **inside-vrf** *vrf-name*
5. **map ip one-to-one**
6. **end** or **commit**

### DETAILED STEPS

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **service cgn** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# service cgn cgn1`<br>`RP/0/RP0/CPU0:router(config-cgn)#` | Configures the instance named cgn1 for the CGv6 application and enters CGv6 configuration mode. |
| **Step 3** | **service-type nat44 nat1**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# service-type nat44 nat1` | Configures the service type keyword definition for CGv6 NAT44 application. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | **inside-vrf** *vrf-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-nat44)# inside-vrf insidevrf1`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)#` | Configures the inside VRF for the CGv6 instance named cgn1 and enters CGv6 inside VRF configuration mode. |
| **Step 5** | **map ip one-to-one**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)# map ip one-to-one` | Configures one-to-one mapping for a CGv6 NAT44 instance. |
| **Step 6** | **end** or **commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-afi)# end`<br>`or`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-afi)# commit` | Saves configuration changes.<br>• When you issue the **end** command, the system prompts you to commit changes:<br>*Uncommitted changes found, commit them before exiting (yes/no/cancel)?*<br>*[cancel]:*<br>• Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>• Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>• Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

# (Only NAT44) Configuring TCP Sequence Check

Perform the following steps for checking the sequence numbers of the packets in a TCP session:

### SUMMARY STEPS

1. **configure**
2. **service cgn** *instance-name*
3. **service-type nat44 nat1**
4. **inside-vrf** *vrf-name*
5. **firewall protocol tcp**
6. **sequence-check**
7. **end** or **commit**

## DETAILED STEPS

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **service cgn** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# service cgn cgn1`<br>`RP/0/RP0/CPU0:router(config-cgn)#` | Configures the instance named cgn1 for the CGN application and enters CGN configuration mode. |
| **Step 3** | **service-type nat44 nat1**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# service-type nat44 nat1` | Configures the service type keyword definition for NAT44 application. |
| **Step 4** | **inside-vrf** *vrf-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-nat44)# inside-vrf insidevrf1`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)#` | Configures the inside VRF for the CGN instance named cgn1 and enters CGN inside VRF configuration mode. |
| **Step 5** | **firewall protocol tcp**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)# protocol tcp`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-proto)#` | Enters the firewall mode and the protocol tcp submode. |
| **Step 6** | **sequence-check**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)# sequence-check 10` | Enables checking of the sequence numbers. The optional **diff-window** keyword allows user to configure a value equal to the difference between the expected and received sequence numbers. The range for this value is 0 to 1,073,725,440. |
| **Step 7** | **end** or **commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-ivrf-sport-inside)# end`<br>`or`<br>`RP/0/RP0/CPU0:router(config-cgn-ivrf-sport-inside)# commit` | Saves configuration changes.<br>• When you issue the **end** command, the system prompts you to commit changes:<br>*Uncommitted changes found, commit them before exiting (yes/no/cancel)?*<br>*[cancel]:*<br>• Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. |

| | Command or Action | Purpose |
|---|---|---|
| | | • Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>• Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

# (Only NAT44) Enabling Address and Port-Dependent Filtering

Perform the following steps to enable address and port-dependent filtering in a NAT44 configuration.

### SUMMARY STEPS

1. **configure**
2. **service cgn** *instance-name*
3. **service-type nat44** *instance-name*
4. **inside-vrf** *instance-name*
5. **filter-policy ignore-port**
6. **end** or **commit**

### DETAILED STEPS

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **service cgn** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# service cgn cgn1`<br>`RP/0/RP0/CPU0:router(config-cgn)#` | Configures the instance named cgn1 for the CGv6 application and enters CGv6 configuration mode. |
| **Step 3** | **service-type nat44** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# service-type nat44 nat1` | Configures the service type keyword definition for CGv6 NAT44 application. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | **inside-vrf** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-nat44)# inside-vrf insidevrf1`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)#` | Configures an inside VRF named insidevrf1 and enters CGv6 inside VRF configuration mode. |
| **Step 5** | **filter-policy ignore-port**<br>**Example:**<br>To enable address and port-dependent filtering<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)# filter-policy`<br>To enable address and port-dependent filtering when the port is not checked:<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)# filter-policy ignore-port` | Enables the address and port-dependent filtering. The optional **ignore-port** keyword is used to disable the port-dependent filtering. |
| **Step 6** | **end** or **commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-if)# end`<br>`or`<br>`RP/0/RP0/CPU0:router(config-if)# commit` | Saves configuration changes.<br>• When you issue the **end** command, the system prompts you to commit changes:<br>*Uncommitted changes found, commit them before exiting (yes/no/cancel)?*<br>*[cancel]:*<br>• Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>• Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>• Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

# (Only NAT44) Configuring NAT0 Mode

Perform these tasks to configure the NAT0 mode for NAT44.

## SUMMARY STEPS

1. **configure**

2. **service cgn** *instance-name*
3. **service-type nat44** *nat1*
4. **inside-vrf** *vrf-name*
5. **map** [**outside-vrf** *outside-vrf-name*] **address-pool** *address/prefix*
6. **nat-mode** *no-nat*
7. **end** or **commit**

## DETAILED STEPS

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **service cgn** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# service cgn cgn1`<br>`RP/0/RP0/CPU0:router(config-cgn)#` | Configures the instance named cgn1 for the CGN application and enters CGN configuration mode. |
| **Step 3** | **service-type nat44** *nat1*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# service-type nat44 nat1` | Configures the service type keyword definition for CGN NAT44 application. |
| **Step 4** | **inside-vrf** *vrf-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-nat44)# inside-vrf insidevrf1`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)#` | Configures an inside VRF named *insidevrf1* and enters CGv6 inside VRF configuration mode. |
| **Step 5** | **map** [**outside-vrf** *outside-vrf-name*] **address-pool** *address/prefix*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)# map outside-vrf ovrf outsideServiceApp ServiceApp2 address-pool 0.0.0.0/0` | Configures an inside VRF to an outside VRF and address pool mapping. |
| **Step 6** | **nat-mode** *no-nat*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)# nat-mode no-nat` | Specifies the NAT0 or NAT bypass mode for NAT44. |
| **Step 7** | **end** or **commit**<br>**Example:** | Saves configuration changes.<br>• When you issue the **end** command, the system prompts you to commit changes: |

| | Command or Action | Purpose |
|---|---|---|
| | `RP/0/RP0/CPU0:router(config-cgn-invrf-natmode)# end`<br>or<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-natmode)# commit` | *Uncommitted changes found, commit them before exiting (yes/no/cancel)?*<br>*[cancel]:*<br>• Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>• Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>• Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

# (Only NAT44) Configuring the Static Destination Address Translation

Perform the following steps to configure static destination address translation:

### SUMMARY STEPS

1. **configure**
2. **service cgn** *instance-name*
3. **service-type nat44 nat1**
4. **inside-vrf** *vrf-name*
5. **map** [**outside-vrf** *outside-vrf-name*] **address-pool** *address/prefix*
6. **static-mapping-file direction i2o-dst** *path of the .csv file*
7. **end** or **commit**

### DETAILED STEPS

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **service cgn** *instance-name*<br>**Example:** | Configures the instance named cgn1 for the CGN application and enters CGN configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| | `RP/0/RP0/CPU0:router(config)# service cgn cgn1`<br>`RP/0/RP0/CPU0:router(config-cgn)#` | |
| **Step 3** | **service-type nat44 nat1**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# service-type nat44 nat1` | Configures the service type keyword definition for CGN NAT44 application. |
| **Step 4** | **inside-vrf** *vrf-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-nat44)# inside-vrf insidevrf1`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)#` | Configures an inside VRF named insidevrf1 and enters CGN inside VRF configuration mode. |
| **Step 5** | **map** [**outside-vrf** *outside-vrf-name*] **address-pool** *address/prefix*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)# map outside-vrf outside vrf1 address-pool 10.10.0.0/16`<br>`or`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)# map address-pool 100.1.0.0/16` | Configures an inside VRF to an outside VRF and address pool mapping. |
| **Step 6** | **static-mapping-file direction i2o-dst** *path of the .csv file*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-afi)# static-mapping-file direction i2o-dst /disk0:/30K.CSV` | Configures static destination address translation using the static mapping configuration file (.csv). In this example, the file is stored in disk0: location. |
| **Step 7** | **end** or **commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-afi)# end`<br>`or`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-afi)# commit` | Saves configuration changes.<br>• When you issue the **end** command, the system prompts you to commit changes:<br>*Uncommitted changes found, commit them before exiting (yes/no/cancel)?*<br>*[cancel]:*<br>• Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>• Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>• Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. |

| | Command or Action | Purpose |
|---|---|---|
| | | • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

# (Only NAT44) Configuring Multiple NetFlow Servers

Perform this task to configure multiple NetFlow servers. Repeat the same task to configure multiple servers.

### SUMMARY STEPS

1. **configure**
2. **service cgn** *instance-name*
3. **service-type nat44 nat1**
4. **inside-vrf** *vrf-name*
5. **external-logging netflow**
6. **server**
7. **address** *address* **port** *number*
8. **ath-mtu** *value*
9. **refresh-rate** *value*
10. **timeout** *value*
11. **session-logging**
12. **end** or **commit**
13.

### DETAILED STEPS

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **service cgn** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# service cgn cgn1`<br>`RP/0/RP0/CPU0:router(config-cgn)#` | Configures the instance named cgn1 for the CGv6 application and enters CGv6 configuration mode. |
| **Step 3** | **service-type nat44 nat1**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# service-type nat44 nat1` | Configures the service type keyword definition for CGv6 NAT44 application. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | **inside-vrf** *vrf-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-nat44)# inside-vrf insidevrf1`<br>`RP/0/RP0/CPU0:router(config-cgn-nat44-invrf)#` | Configures the inside VRF for the CGv6 instance named cgn1 and enters CGv6 inside VRF configuration mode. |
| **Step 5** | **external-logging netflow**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)# external-logging netflow version 9`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog)#` | Configures the external-logging facility for the NAT44 instance. |
| **Step 6** | **server**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog)# server`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog-server)#` | Configures the logging server information for the IPv4 address and port for the server that is used for the netflow-v9 based external-logging facility. |
| **Step 7** | **address** *address* **port** *number*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog-server)# address 2.3.4.5 port 45` | Configures the IPv4 address and port number 45 to log Netflow entries for the NAT table. |
| **Step 8** | **ath-mtu** *value*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog-server)# path-mtu 2900` | Configures the path MTU with the value of 2900 for the netflowv9-based external-logging facility. |
| **Step 9** | **refresh-rate** *value*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog-server)# refresh-rate 50` | Configures the refresh rate value of 50 to log Netflow-based external logging information for an inside VRF. |
| **Step 10** | **timeout** *value*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog-server)# timeout 50` | Configures the timeout value of 50 for Netflow logging of NAT table entries for an inside VRF. |
| **Step 11** | **session-logging**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog-server)# session-logging` | Configures the session logging for a NAT44 instance. |
| **Step 12** | **end** or **commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog-server)# end` | Saves configuration changes.<br>• When you issue the **end** command, the system prompts you to commit changes: |

| | Command or Action | Purpose |
|---|---|---|
| | or<br>RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog-server)# commit | *Uncommitted changes found, commit them before exiting (yes/no/cancel)?*<br>*[cancel]:*<br>• Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>• Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>• Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |
| **Step 13** | | To configure a second server, repeat the steps from 7 to 11. |

# (Only NAT64) Configuring External Logging for NetFlow Servers

Perform this task to configure external logging for NetFlow servers.

## SUMMARY STEPS

1. **configure**
2. **service cgn** *instance-name*
3. **service-location preferred-active** *node-id*
4. **service-type nat64 stateful** *name*
5. **external-logging netflow**
6. **server**
7. **address** *address* **port** *number*
8. **ath-mtu** *value*
9. **refresh-rate** *value*
10. **timeout** *value*
11. **session-logging**
12. **end** or **commit**

## DETAILED STEPS

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **service cgn** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# service cgn cgn1`<br>`RP/0/RP0/CPU0:router(config-cgn)#` | Configures the instance named cgn1 for the CGv6 application and enters CGv6 configuration mode. |
| **Step 3** | **service-location preferred-active** *node-id*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# service-location preferred-active 0/1/CPU01` | Specifies the location in which the application starts. |
| **Step 4** | **service-type nat64 stateful** *name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# service-type nat64 stateful nat64_1` | Configures the service type keyword and the name for CGv6 NAT64 application. |
| **Step 5** | **external-logging netflow**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)# external-logging netflow version 9`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog)#` | Configures the external-logging facility for the NAT64 instance. |
| **Step 6** | **server**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog)# server`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog-server)#` | Configures the logging server information for the IPv6 address and port for the server that is used for the netflow-v9 based external-logging facility. |
| **Step 7** | **address** *address* **port** *number*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog-server)# address 2.3.4.5 port 614` | Configures the IPv6 address and port number 614 to log Netflow entries for the NAT table. |
| **Step 8** | **ath-mtu** *value*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog-server)# path-mtu 2900` | Configures the path MTU with the value of 2900 for the netflowv9-based external-logging facility. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 9** | **refresh-rate** *value*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog-server)# refresh-rate 50` | Configures the refresh rate value of 50 to log Netflow-based external logging information for an inside VRF. |
| **Step 10** | **timeout** *value*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog-server)# timeout 50` | Configures the timeout value of 50 for Netflow logging of NAT table entries for an inside VRF. |
| **Step 11** | **session-logging**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog-server)# session-logging` | Configures the session logging for a NAT64 instance. |
| **Step 12** | **end** or **commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog-server)# end`<br>`or`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog-server)# commit` | Saves configuration changes.<br>• When you issue the **end** command, the system prompts you to commit changes:<br>*Uncommitted changes found, commit them before exiting (yes/no/cancel)?*<br>*[cancel]:*<br>• Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>• Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>• Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

# (Only NAT44) Configuring Multiple Syslog Servers

Perform this task to configure an additional Syslog server. Repeat the same task to configure multiple servers.

### SUMMARY STEPS

1. **configure**
2. **service cgn** *instance-name*
3. **service-type nat44 nat1**

4. **inside-vrf** *vrf-name*
5. **external-logging syslog**
6. **server**
7. **address***address***port***number*
8. **path-mtu***value*
9. **refresh-rate** *value*
10. **timeout** *value*
11. **session-logging**
12. **end** or **commit**

## DETAILED STEPS

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **service cgn** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# service cgn cgn1`<br>`RP/0/RP0/CPU0:router(config-cgn)#` | Configures the instance named cgn1 for the CGv6 application and enters CGv6 configuration mode. |
| **Step 3** | **service-type nat44 nat1**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# service-type nat44 nat1` | Configures the service type keyword definition for CGv6 NAT44 application. |
| **Step 4** | **inside-vrf** *vrf-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-nat44)# inside-vrf insidevrf1`<br>`RP/0/RP0/CPU0:router(config-cgn-nat44-invrf)#` | Configures the inside VRF for the CGv6 instance named cgn1 and enters CGv6 inside VRF configuration mode. |
| **Step 5** | **external-logging syslog**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)# external-logging syslog`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog)#` | Configures the external-logging facility for the NAT44 instance. |
| **Step 6** | **server**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog)# server`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog-server)#` | Configures the logging server information for the IPv4 address and port for the server that is used for the netflow-v9 based external-logging facility. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 7** | **address***address***port***number*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog-server)# address 2.3.4.5 port 45` | Configures the IPv4 address and port number 45 to log Netflow entries for the NAT table. |
| **Step 8** | **path-mtu***value*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog-server)# path-mtu 200` | Configures the path MTU with the value of 200 for the syslog-based external-logging facility. |
| **Step 9** | **refresh-rate** *value*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog-server)# refresh-rate 50` | Configures the refresh rate value of 50 to log Netflow-based external logging information for an inside VRF. |
| **Step 10** | **timeout** *value*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog-server)# timeout 50` | Configures the timeout value of 50 for Netflow logging of NAT table entries for an inside VRF. |
| **Step 11** | **session-logging**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog-server)# session-logging` | Configures the session logging for a NAT44 instance. |
| **Step 12** | **end** or **commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog-server)# end`<br>`or`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog-server)# commit` | Saves configuration changes.<br>• When you issue the **end** command, the system prompts you to commit changes:<br>*Uncommitted changes found, commit them before exiting (yes/no/cancel)?*<br>*[cancel]:*<br>• Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>• Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>• Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

**Example**

# (Only NAT44) Configuring External Logging for the NAT Table Entries

Perform the following to configure external logging for NAT table entries.

## Configuring the Server Address and Port for Netflow Logging

Perform this task to configure the server address and port to log network address translation (NAT) table entries for Netflow logging.

### SUMMARY STEPS

1. **configure**
2. **service cgn** *instance-name*
3. **service-type nat44 nat1**
4. **inside-vrf** *vrf-name*
5. **external-logging netflowv9**
6. **server**
7. **address** *address* **port** *number*
8. **end** or **commit**

### DETAILED STEPS

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **service cgn** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# service cgn cgn1`<br>`RP/0/RP0/CPU0:router(config-cgn)#` | Configures the instance named cgn1 for the CGN application and enters CGN configuration mode. |
| **Step 3** | **service-type nat44 nat1**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# service-type nat44 nat1` | Configures the service type keyword definition for NAT44 application. |
| **Step 4** | **inside-vrf** *vrf-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# inside-vrf insidevrf1`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)#` | Configures the inside VRF for the CGN instance named cgn1 and enters CGN inside VRF configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 5** | **external-logging netflowv9**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)# external-logging netflowv9`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog)#` | Configures the external-logging facility for the CGN instance named cgn1 and enters CGN inside VRF address family external logging configuration mode. |
| **Step 6** | **server**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog)# server`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog-server)#` | Configures the logging server information for the IPv4 address and port for the server that is used for the netflowv9-based external-logging facility and enters CGN inside VRF address family external logging server configuration mode. |
| **Step 7** | **address** *address* **port** *number*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog-server)# address 2.3.4.5 port 45` | Configures the IPv4 address and port number 45 to log Netflow entries for the NAT table. |
| **Step 8** | **end** or **commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog-server)# end`<br>`or`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog-server)# commit` | Saves configuration changes.<br>• When you issue the **end** command, the system prompts you to commit changes:<br>*Uncommitted changes found, commit them before exiting (yes/no/cancel)?*<br>*[cancel]:*<br>• Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>• Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>• Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

## Configuring the Path Maximum Transmission Unit for Netflow Logging

Perform this task to configure the path maximum transmission unit (MTU) for the netflowv9-based external-logging facility for the inside VRF.

## SUMMARY STEPS

1. **configure**
2. **service cgn** *instance-name*
3. **service-type nat44 nat1**
4. **inside-vrf** *vrf-name*
5. **external-logging netflowv9**
6. **server**
7. **path-mtu** *value*
8. **end** or **commit**

## DETAILED STEPS

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **service cgn** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# service cgn cgn1`<br>`RP/0/RP0/CPU0:router(config-cgn)#` | Configures the instance named cgn1 for the CGN application and enters CGN configuration mode. |
| **Step 3** | **service-type nat44 nat1**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# service-type nat44 nat1` | Configures the service type keyword definition for NAT44 application. |
| **Step 4** | **inside-vrf** *vrf-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# inside-vrf insidevrf1`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)#` | Configures the inside VRF for the CGN instance named cgn1 and enters CGN inside VRF configuration mode. |
| **Step 5** | **external-logging netflowv9**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)# external-logging netflowv9`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog)#` | Configures the external-logging facility for the CGN instance named cgn1 and enters CGN inside VRF address family external logging configuration mode. |
| **Step 6** | **server**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog)# server`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog-server)#` | Configures the logging server information for the IPv4 address and port for the server that is used for the netflowv9-based external-logging facility and enters CGN inside VRF address family external logging server configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 7** | **path-mtu** *value*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog-server)# path-mtu 2900` | Configures the path MTU with the value of 2900 for the netflowv9-based external-logging facility. |
| **Step 8** | **end** or **commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog-server)# end`<br>or<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog-server)# commit` | Saves configuration changes.<br>• When you issue the **end** command, the system prompts you to commit changes:<br>*Uncommitted changes found, commit them before exiting (yes/no/cancel)?*<br>*[cancel]:*<br>• Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>• Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>• Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

## Configuring the Refresh Rate for Netflow Logging

Perform this task to configure the refresh rate at which the Netflow-v9 logging templates are refreshed or resent to the Netflow-v9 logging server.

### SUMMARY STEPS

1. **configure**
2. **service cgn** *instance-name*
3. **service-type nat44 nat1**
4. **inside-vrf** *vrf-name*
5. **external-logging netflowv9**
6. **server**
7. **refresh-rate** *value*
8. **end** or **commit**

## DETAILED STEPS

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **service cgn** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# service cgn cgn1`<br>`RP/0/RP0/CPU0:router(config-cgn)#` | Configures the instance named cgn1 for the CGN application and enters CGN configuration mode. |
| **Step 3** | **service-type nat44 nat1**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# service-type nat44 nat1` | Configures the service type keyword definition for NAT44 application. |
| **Step 4** | **inside-vrf** *vrf-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# inside-vrf insidevrf1`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)#` | Configures the inside VRF for the CGN instance named cgn1 and enters CGN inside VRF configuration mode. |
| **Step 5** | **external-logging netflowv9**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)# external-logging netflowv9`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog)#` | Configures the external-logging facility for the CGN instance named cgn1 and enters CGN inside VRF address family external logging configuration mode. |
| **Step 6** | **server**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog)# server`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog-server)#` | Configures the logging server information for the IPv4 address and port for the server that is used for the netflowv9-based external-logging facility and enters CGN inside VRF address family external logging server configuration mode. |
| **Step 7** | **refresh-rate** *value*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog-server)# refresh-rate 50` | Configures the refresh rate value of 50 to log Netflow-based external logging information for an inside VRF. |
| **Step 8** | **end** or **commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog-server)# end`<br>`or` | Saves configuration changes.<br>• When you issue the **end** command, the system prompts you to commit changes:<br>*Uncommitted changes found, commit them before exiting (yes/no/cancel)?* |

| | Command or Action | Purpose |
|---|---|---|
| | `RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog-server)# commit` | *[cancel]:*<br>• Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>• Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>• Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

## Configuring the Timeout for Netflow Logging

Perform this task to configure the frequency in minutes at which the Netflow-V9 logging templates are to be sent to the Netflow-v9 logging server.

### SUMMARY STEPS

1. **configure**
2. **service cgn** *instance-name*
3. **service-type nat44 nat1**
4. **inside-vrf** *vrf-name*
5. **external-logging netflowv9**
6. **server**
7. **timeout***value*
8. **end** or **commit**

### DETAILED STEPS

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **service cgn** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# service cgn cgn1`<br>`RP/0/RP0/CPU0:router(config-cgn)#` | Configures the instance named cgn1 for the CGN application and enters CGN configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 3** | **service-type nat44 nat1**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# service-type nat44 nat1` | Configures the service type keyword definition for NAT44 application. |
| **Step 4** | **inside-vrf** *vrf-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# inside-vrf insidevrf1`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)#` | Configures the inside VRF for the CGN instance named cgn1 and enters CGN inside VRF configuration mode. |
| **Step 5** | **external-logging netflowv9**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)# external-logging netflowv9`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog)#` | Configures the external-logging facility for the CGN instance named cgn1 and enters CGN inside VRF address family external logging configuration mode. |
| **Step 6** | **server**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog)# server`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog-server)#` | Configures the logging server information for the IPv4 address and port for the server that is used for the netflowv9-based external-logging facility and enters CGN inside VRF address family external logging server configuration mode. |
| **Step 7** | **timeout***value*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog-server)# timeout 50` | Configures the timeout value of 50 for Netflow logging of NAT table entries for an inside VRF. |
| **Step 8** | **end** or **commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog-server)# end`<br>`or`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog-server)# commit` | Saves configuration changes.<br>• When you issue the **end** command, the system prompts you to commit changes:<br>*Uncommitted changes found, commit them before exiting (yes/no/cancel)?*<br>*[cancel]:*<br>• Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>• Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>• Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. |

| | Command or Action | Purpose |
|---|---|---|
| | | • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

# (Only NAT44) Syslog Logging

Perform the following tasks to configure Syslog Logging for NAT table entries.

## Configuring the Server Address and Port for Syslog Logging

Perform this task to configure the server address and port to log DS-Lite entries for Syslog logging.

### SUMMARY STEPS

1. **configure**
2. **service cgn** *instance-name*
3. **service-type ds-lite** *instance_name*
4. **external-logging syslog**
5. **server**
6. **address***address***port***number*
7. **end** or **commit**

### DETAILED STEPS

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **service cgn** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# service cgn cgn1`<br>`RP/0/RP0/CPU0:router(config-cgn)#` | Configures the instance named cgn1 for the CGv6 application and enters CGv6 configuration mode. |
| **Step 3** | **service-type ds-lite** *instance_name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# service-type ds-lite ds-lite1` | Configures the service type keyword definition for the DS-Lite application. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | **external-logging syslog**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-ds-lite)#external-logging syslog`<br>`RP/0/RP0/CPU0:router(config-cgn-ds-lite-extlog)#` | Configures the external-logging facility for the CGv6 instance named cgn1 and enters CGv6 external logging configuration mode. |
| **Step 5** | **server**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-ds-lite-extlog)# server`<br>`RP/0/RP0/CPU0:router(config-cgn-ds-lite-extlogserver)#` | Configures the logging server information for the IPv4 address and port for the server that is used for the syslog-based external-logging facility and enters CGv6 external logging server configuration mode. |
| **Step 6** | **address***address***port***number*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-ds-lite-extlogserver)#address 2.3.4.5 port 45` | Configures the IPv4 address and port number 45 to log Netflow entries. |
| **Step 7** | **end** or **commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-ds-lite-extlogserver)#end`<br>`or`<br>`RP/0/RP0/CPU0:router(config-cgn-ds-lite-extlogserver)#commit` | Saves configuration changes.<br>• When you issue the **end** command, the system prompts you to commit changes:<br>*Uncommitted changes found, commit them before exiting (yes/no/cancel)?*<br>*[cancel]:*<br>• Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>• Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>• Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

## Configuring the Host-Name for Syslog Logging

Perform this task to configure the host name to be filled in the Netflow header for the syslog logging.

### SUMMARY STEPS

1. **configure**
2. **service cgn** *instance-name*

3. **service-type nat44 nat1**
4. **inside-vrf** *vrf-name*
5. **external-logging syslog**
6. **server**
7. **host-name***name*
8. **end** or **commit**

## DETAILED STEPS

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **service cgn** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# service cgn cgn1`<br>`RP/0/RP0/CPU0:router(config-cgn)#` | Configures the instance named cgn1 for the CGv6 application and enters CGv6 configuration mode. |
| **Step 3** | **service-type nat44 nat1**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# service-type nat44 nat1` | Configures the service type keyword definition for CGv6 NAT44 application. |
| **Step 4** | **inside-vrf** *vrf-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-nat44)# inside-vrf insidevrf1`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)#` | Configures the inside VRF for the CGv6 instance named cgn1 and enters CGv6 inside VRF configuration mode. |
| **Step 5** | **external-logging syslog**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)# external-logging syslog`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog)#` | Configures the external-logging facility for the CGv6 instance named cgn1 and enters CGv6 inside VRF address family external logging configuration mode. |
| **Step 6** | **server**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog)# server`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog-server)#` | Configures the logging server information for the IPv4 address and port for the server that is used for the syslog-based external-logging facility and enters CGv6 inside VRF address family external logging server configuration mode. |
| **Step 7** | **host-name***name*<br>**Example:** | Configures the host name for the syslog-based external-logging facility. |

| | Command or Action | Purpose |
|---|---|---|
| | `RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog-server)# host-name host1` | |
| **Step 8** | **end** or **commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog-server)# end`<br>`or`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog-server)# commit` | Saves configuration changes.<br>• When you issue the **end** command, the system prompts you to commit changes:<br>*Uncommitted changes found, commit them before exiting (yes/no/cancel)?*<br>*[cancel]:*<br>• Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>• Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>• Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

## Configuring the Path Maximum Transmission Unit for Syslog Logging

### SUMMARY STEPS

1. **configure**
2. **service cgn** *instance-name*
3. **service-type nat44 nat1**
4. **inside-vrf** *vrf-name*
5. **external-logging syslog**
6. **server**
7. **path-mtu***value*
8. **end** or **commit**

### DETAILED STEPS

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:** | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| | `RP/0/RP0/CPU0:router# configure` | |
| **Step 2** | **service cgn** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# service cgn cgn1`<br>`RP/0/RP0/CPU0:router(config-cgn)#` | Configures the instance named cgn1 for the CGv6 application and enters CGv6 configuration mode. |
| **Step 3** | **service-type nat44 nat1**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# service-type nat44 nat1` | Configures the service type keyword definition for CGv6 NAT44 application. |
| **Step 4** | **inside-vrf** *vrf-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-nat44)# inside-vrf insidevrf1`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)#` | Configures the inside VRF for the CGv6 instance named cgn1 and enters CGv6 inside VRF configuration mode. |
| **Step 5** | **external-logging syslog**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)# external-logging syslog`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog)#` | Configures the external-logging facility for the CGv6 instance named cgn1 and enters CGv6 inside VRF address family external logging configuration mode. |
| **Step 6** | **server**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog)# server`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog-server)#` | Configures the logging server information for the IPv4 address and port for the server that is used for the syslog-based external-logging facility and enters CGv6 inside VRF address family external logging server configuration mode. |
| **Step 7** | **path-mtu***value*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog-server)# path-mtu 200` | Configures the path MTU with the value of 200 for the syslog-based external-logging facility. |
| **Step 8** | **end** or **commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog-server)# end`<br>or<br>`RP/0/RP0/CPU0:router(config-cgn-invrf-af-extlog-server)# commit` | Saves configuration changes.<br>• When you issue the **end** command, the system prompts you to commit changes:<br>*Uncommitted changes found, commit them before exiting (yes/no/cancel)?*<br>*[cancel]:*<br>• Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. |

| | Command or Action | Purpose |
|---|---|---|
| | | • Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>• Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

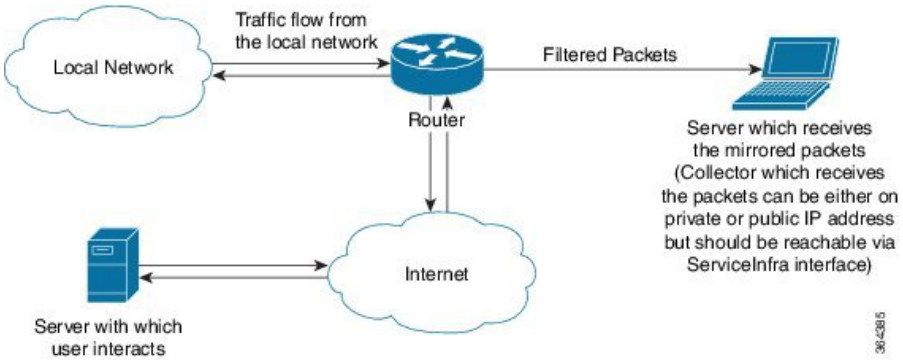
# Traffic Flow Mirroring

Traffic flow mirroring is a solution which enables you to monitor the incoming and outgoing traffic on the VSM module (of ASR9K) running a CGN instance. This solution helps you to debug and analyze packets for issues pertaining to NAT-ing (NAT44). The traffic is filtered based on a set of particular parameters, which can be set by the user. The packets, collected, are encapsulated in a GRE envelope and sent to a pre-configured collector like a UNIX system, laptop, etc. This envelope contains a field, which provides information about the type of packet whether the packets are In2Out packet, Out2In packet, pre-NAT, post-NAT, or dropped, analyzing this field information, the issues pertaining to NAT can be debugged.

**Salient Features:**

- Any packets dropped will be mirrored.
- The packets are filtered based on destination address; and refined further based on port number, protocol, and IP addresses of the subscriber devices that are mirrored.
- Mirroring of up to 16 VRFs is supported when the destination address filter is configured. There is no limit on the number of VRFs supported when the mirroring is enabled for only the dropped packets.

***Figure 1: Traffic Flow Mirroring Topology***



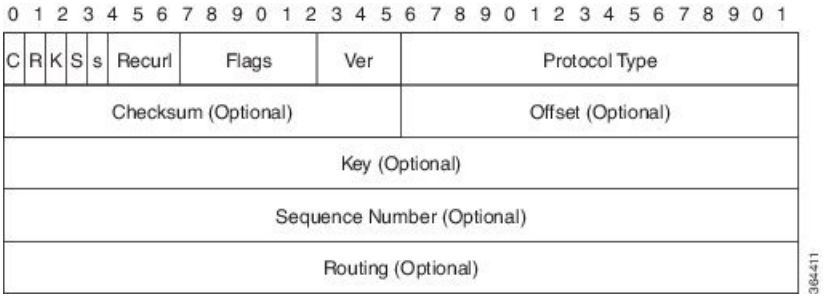

If the packets are filtered based on the destination IP address, then destination IP address is a mandatory field for the solution whereas a few of the fields like protocol used, destination port, private source prefix, etc. are optional.

**Note** Mirroring occurs only for packets that are intercepted after the feature is turned on.

### Mirrored Packet Data Interpretation

The packets received at the collector have the original packet as the payload encapsulated in a GRE header. A typical GRE header is as shown in the following figure.

***Figure 2: GRE Header***




The KEY field in the GRE header contains the value. The following table lists the values and the description associated with those values.

***Table 1: List of KEY field values and their descriptions:***

| Value | Description |
|---|---|
| 1 | In to Out direction, pre-nat packet |
| 2 | Out to In direction, pre-nat packet |
| 3 | In to Out direction, post-nat packet |
| 4 | Out to In direction, post-nat packet |
| 5 | Dropped In to Out TCP, PPTP control message packet. |
| 6 | Dropped In to Out TCP Fragment packet. (Received non-first fragment.) |
| 7 | Dropped In to Out TCP packet. (Failed to create new NAT entry.) |
| 8 | Dropped In to Out TCP packet due to no session entry. |
| 9 | Dropped In to Out TCP packet. (Source port is zero) |
| 10 | Dropped In to Out TCP packet. (None sync drop) |
| 11 | Dropped In to Out TCP packet (Session creation fail) |
| 12 | Dropped In to Out TCP packet with TTL <= 1. (No ICMP generated due to throttling) |
| 13 | Dropped packet as ICMP is sent for first fragment only. |

| | |
|---|---|
| 14 | Dropped packet due to Invalid ICMP error code. |
| 15 | Dropped In to Out due ICMP error packet with TTL <= 1. |
| 16 | Dropped In to Out ICMP packet due to no NAT entry. |
| 17 | Dropped Out to In ICMP packet (ipv4 packet too large for the tunnel) |
| 18 | Dropped Out to In ICMP packet due to no NAT entry. |
| 19 | Dropped In to Out ICMP packet due to no session. |
| 20 | Dropped In to Out ICMP packet with TTL <= 1. (No ICMP generated due to throttling) |
| 21 | Dropped In to Out ICMP query packet due to no NAT entry. |
| 22 | Dropped Out to In ICMP query. (No NAT entry) |
| 23 | Dropped Out to In ICMP query packet due to end point filtering. (EDF is enabled). |
| 24 | Dropped Out to In ICMP query packet, could not generate ICMP packet due to throttling. |
| 25 | Dropped Out to In ICMP packet due to no session. |
| 26 | Dropped Out to In ICMP packet due to no NAT entry. |
| 27 | Dropped port control protocol (PCP) packet, as it couldn't be handled. |
| 28 | Dropped In to Out PPTP packet (PPTP not configured) |
| 29 | Dropped In to Out PPTP packet with TTL <= 1 (No ICMP generated due to throttling) |
| 30 | Dropped Out to In PPTP packet (PPTP not configured) |
| 31 | Dropped Out to In PPTP fragment packet (No NAT entry) |
| 32 | Dropped Out to In PPTP packet (No NAT entry) |
| 33 | Dropped Out to In PPTP packet with TTL <= 1. (No ICMP generated due to throttling.) |
| 34 | Dropped In to Out UDP packet (Has no available ports) |
| 35 | Dropped In to Out UDP packet (UDP port value of 0). |
| 36 | Dropped In to Out UDP packet (No configuration available). |
| 37 | Dropped In to Out UDP packet (No ICMP message generated). |
| 38 | Dropped In to Out UDP packet (Create session failed). |
| 39 | Dropped In to Out UDP packet (VRF not in run state) |

| | |
|---|---|
| 40 | Dropped In to Out UDP packet (Port limit exceeded) |
| 41 | Dropped In to Out UDP packet with TTL <= 1. (No ICMP generated due to throttling.) |
| 42 | Dropped In to Out UDP packet (No direct port available). |
| 43 | Dropped Out to In UDP packet (No NAT entry). |
| 44 | Dropped Out to In UDP packet due to end point filtering. (EDF is enabled) |
| 45 | Dropped Out to In UDP packet (No NAT entry). |
| 46 | Dropped Out to In UDP packet (Create session DB failed or Session limit exceeded.) |
| 47 | Dropped Out to In UDP packet as it is too large for tunneling.<br>**Note**<br>ICMP not generated due to throttling. |
| 48 | Dropped Out to In UDP packet (Create session failed.) |
| 49 | Dropped Out to In UDP fragment packet (No NAT entry). |
| 50 | Not used |
| 51 | Dropped Out to In Error fragment packet. |
| 52 | Dropped Out to In unsupported protocol Fragment packet. |
| 53 | Dropped Out to In TCP packet (PPTP control message dropped.) |
| 54 | Dropped Out to In TCP packet (No NAT entry) |
| 55 | Dropped Out to In TCP packet (First fragment packet drop) |
| 56 | Dropped Out to In TCP due to end point filtering. (EDF is enabled.) |
| 57 | Dropped Out to In UDP packet as it is too large for tunneling.<br>**Note**<br>ICMP not generated due to throttling. |
| 58 | Dropped Out to In TCP packet. (Create session failed.) |
| 59 | Dropped Out to in TCP fragment packet (No NAT entry) |
| 60 | Dropped Out to in TCP packet (SYN or RST flags not set for TCP session to be established.) |
| 61 | Dropped Out to in TCP packet (Sequence mismatch) |
| 62 | Dropped Out to In TCP packet with TTL <= 1. (No ICMP generated due to throttling.) |

# Limitations and Assumptions

The following are a few of the assumptions and limitations of the traffic flow mirroring solution:

- At any given point in time, only one traffic flow mirroring per inside-vrf is allowed.
- If the collector IP address is not configured, the traffic packet mirroring is blocked. In case the collector IP address is not reachable, the mirrored packets are dropped.
- If the protocol is not provided, both TCP and UDP packets are mirrored.
- If the port number is not mentioned, the traffic flowing through all the destination ports are mirrored.
- If a private source IP address is not configured, the mirroring is performed for all subscribers of the VRF, that is listed. This can reduce the performance of VSM and also lead to choking the collector. It is advisable to configure as many parameters as possible to filter and mirror only the required packets.
- Performance figures of VSM are not guaranteed when traffic mirroring is on.
- Traffic flow mirroring solution assumes that the collector is reachable to the router in the default VRF. The router does not attempts to ping or get acknowledgments to ascertain if the collector is receiving the packets.

# Configuring Mirroring Using Destination Address Filter and Collector IP Address

Perform this task to configure mirroring the traffic packets using a destination address filter and collector IP address.

### SUMMARY STEPS

1. **configure**
2. **service cgn** *instance-name*
3. **service-location preferred-active** *node-id*
4. **service-type nat44 nat1** *instance*
5. **inside-vrf** *vrf-name*
6. **mirror-packets**
7. **end** or **commit**

### DETAILED STEPS

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 2** | **service cgn** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# service cgn cgn1`<br>`RP/0/RP0/CPU0:router(config-cgn)#` | Configures the instance named cgn1 for the CGN application and enters CGN configuration mode. |
| **Step 3** | **service-location preferred-active** *node-id*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# service-location preferred-active 0/1/CPU0` | Specifies the global command applied per CGN instance. It initiates the particular instance of the CGN application on the active and standby locations. |
| **Step 4** | **service-type nat44 nat1** *instance*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# service-type nat44 nat1` | Configures the service type keyword definition for the NAT44 NAT1 application. |
| **Step 5** | **inside-vrf** *vrf-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-nat44)# inside-vrf BLR_BTM`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)#` | Configures an inside VRF named BLR_BTM and enters CGN inside VRF configuration mode. |
| **Step 6** | **mirror-packets**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)# mirror-packets`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)# destination-ipv4-address 201.22.3.45`<br>`!`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)# collector-ipv4-address 187.2.3.55`<br>`!`<br>`!` | Filters the traffic such that the packets are mirrored onto the provided destination collector IP address. |
| **Step 7** | **end** or **commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)# end`<br>`or`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)# commit` | Saves configuration changes.<br>• When you issue the **end** command, the system prompts you to commit changes:<br>*Uncommitted changes found, commit them before exiting (yes/no/cancel)?*<br>*[cancel]:*<br>• Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>• Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. |

| | Command or Action | Purpose |
|---|---|---|
| | | • Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

### Example

The following example shows how to filter and configure data packets to be mirrored onto a collector with the destination IP address and the collector IP address provided.

```
service cgn cgn1
 service-location preferred-active 0/1/CPU0
 service-type nat44 nat1
  inside-vrf BLR_BTM
   mirror-packets
    destination-ipv4-address 201.22.3.45
    !
    collector-ipv4-address 187.2.3.55
   !
  !
 !
!
```

# Configuring Mirroring Using Destination Address, Port Number, Protocol Type, Source-Prefix Filters, and Collector IP Address

Perform this task to configure mirroring the traffic packets using a destination address, port number, protocol type, source-prefix filter and collector IP address.

### SUMMARY STEPS

1. **configure**
2. **service cgn** *instance-name*
3. **service-location preferred-active** *node-id*
4. **service-type nat44 nat1** *instance*
5. **inside-vrf** *vrf-name*
6. **mirror-packets**
7. **end** or **commit**

## DETAILED STEPS

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **service cgn** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# service cgn cgn1`<br>`RP/0/RP0/CPU0:router(config-cgn)#` | Configures the instance named cgn1 for the CGN application and enters CGN configuration mode. |
| **Step 3** | **service-location preferred-active** *node-id*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# service-location preferred-active 0/1/CPU0` | Specifies the global command applied per CGN instance. It initiates the particular instance of the CGN application on the active and standby locations. |
| **Step 4** | **service-type nat44 nat1** *instance*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# service-type nat44 nat1` | Configures the service type keyword definition for the NAT44 NAT1 application. |
| **Step 5** | **inside-vrf** *vrf-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-nat44)# inside-vrf BLR_BTM3`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)#` | Configures an inside VRF named BLR_BTM3 and enters CGN inside VRF configuration mode. |
| **Step 6** | **mirror-packets**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)#`<br>`mirror-packets`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)#`<br>`destination-ipv4-address 201.22.3.45`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)#`<br>`protocol-type tcp udp`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)# port 4002`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)#`<br>`source-prefix 100.1.1.252/30`<br>`!`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)#`<br>`collector-ipv4-address 187.2.3.5`<br>`!`<br>`!` | Configures the traffic packets to be mirrored onto the provided destination collector IP address. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 7** | **end** or **commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)# end`<br>`or`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)# commit` | Saves configuration changes.<br>• When you issue the **end** command, the system prompts you to commit changes:<br>*Uncommitted changes found, commit them before exiting (yes/no/cancel)?*<br>*[cancel]:*<br>• Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>• Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>• Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

### Example

The following example shows how to filter and configure packets to be mirrored onto a collector with the destination details like the IP address, protocol type, port number, source-prefix filter, and the collector IP address.

```
service cgn cgn1
 service-location preferred-active 0/1/CPU0
 service-type nat44 nat1
  inside-vrf BLR_BTM3
   mirror-packets
    destination-ipv4-address 201.22.3.45
     protocol-type tcp udp
     port 4002
     source-prefix 100.1.1.252/30
    !
    collector-ipv4-address 187.2.4.5
   !
  !
 !
!
```

# Configuring Mirroring for Dropped Packets Using Collector IP Address

Perform this task to configure mirroring the dropped traffic packets using collector IP address.

## SUMMARY STEPS

1. **configure**
2. **service cgn** *instance-name*
3. **service-location preferred-active** *node-id*
4. **service-type nat44 nat1** *instance*
5. **inside-vrf** *vrf-name*
6. **mirror-packets**
7. **end** or **commit**

## DETAILED STEPS

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **service cgn** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# service cgn cgn1`<br>`RP/0/RP0/CPU0:router(config-cgn)#` | Configures the instance named cgn1 for the CGN application and enters CGN configuration mode. |
| **Step 3** | **service-location preferred-active** *node-id*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# service-location preferred-active 0/1/CPU0` | Specifies the global command applied per CGN instance. It initiates the particular instance of the CGN application on the active and standby locations. |
| **Step 4** | **service-type nat44 nat1** *instance*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# service-type nat44 nat1` | Configures the service type keyword definition for the NAT44 NAT1 application. |
| **Step 5** | **inside-vrf** *vrf-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# service-type nat44 nat1`<br>`RP/0/RP0/CPU0:router(config-cgn-nat44)# inside-vrf BLR_BTM3`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)#` | Configures an inside VRF named BLR_BTM3 and enters CGN inside VRF configuration mode. |
| **Step 6** | **mirror-packets**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)# mirror-packets` | Configures the dropped traffic packets to be mirrored onto the provided destination collector IP address. |

| | Command or Action | Purpose |
|---|---|---|
| | `RP/0/RP0/CPU0:router(config-cgn-invrf)# all-drops`<br>`!`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)#`<br>`collector-ipv4-address 187.2.3.56`<br>`!`<br>`!` | |
| **Step 7** | **end** or **commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)# end`<br>`or`<br>`RP/0/RP0/CPU0:router(config-cgn-invrf)# commit` | Saves configuration changes.<br>• When you issue the **end** command, the system prompts you to commit changes:<br>*Uncommitted changes found, commit them before exiting (yes/no/cancel)?*<br>*[cancel]:*<br>• Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>• Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>• Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

### Example

The following example shows how to filter and configure dropped traffic packets to be mirrored onto a collector with the IP address provided.

```
service cgn cgn1
 service-location preferred-active 0/1/CPU0
 service-type nat44 nat1
  inside-vrf BLR_BTM2
   mirror-packets
    all-drops
    collector-ipv4-address 187.2.3.56
   !
  !
 !
!
```

# Mapping of Address and Port-Encapsulation Mode

Mapping of Address and Port-Encapsulation Mode (MAP-E) is a CGN solution that allows a service provider to enable IPv4 services at IPv6 (customer) sites to which it provides Customer Premise Equipment (CPE). This approach utilizes stateless IPv4-in-IPv6 encapsulation to transit IPv6-enabled network infrastructure. The encapsulation must be supported by the CPE and MAP-E Gateway/Border Relay, which removes the IPv6 encapsulation from IPv4 packets while forwarding them to the Internet. The provider access network can now be on IPv6, while customers see IPv6 and IPv4 service simultaneously.

MAP-E also helps manage IPv4 address exhaustion by keeping the stateful NAT44 on CPE. MAP-E is not supported on any of the VRF interfaces, that is, either IPv4 or IPv6, whereas Map-T is supported with VRF interfaces along with an SMU.

**Note**

- If the cumulative sum of EA bit value and cpe-domain ipv6 prefix value is more than 64 for an interface, then the traffic is dropped for that interface. EA bit value is calculated as (32- ipv4 prefix) + X), where 32 is a constant and X is the power value of 2 when the sharing ratio is expressed as $2^X$.

  Ensure that the EA bit value is less than or equal to 32 and sharing ratio is less than or equal to 256 to avoid any traffic drops.

- You should not modify or delete the existing MAP-E and MAP-T configuration. If you modify the existing configuration, the changes are not reflected in the PBR policy.

  To update the configuration, delete the existing MAP-E and MAP-T instance and add new instance with the required changes.

- Do not configure MAP-E and MAP-T on the same CGv6 instance simultaneously.

# Configuring MAP-E

Perform these tasks to configure MAP-E.

## Configuring the Application Service Virtual Interface

This section lists the guidelines for selecting service application interfaces for MAP-E.

- Pair ServiceApp<n> with ServiceApp<n+1>, where <n> is an odd integer. This is to ensure that the ServiceApp pairs works with a maximum throughput. For example, ServiceApp1 with ServiceApp2 or ServiceApp3 with ServiceApp4.
- Pair ServiceApp<n> with ServiceApp<n+5> or ServiceApp<n+9>, and so on, where <n> is an odd integer. For example, ServiceApp1 with ServiceApp6, ServiceApp1 with ServiceApp10, ServiceApp3 with ServiceApp8, or ServiceApp3 with ServiceApp12.
- Pair ServiceApp<n> with ServiceApp<n+4>, where <n> is an integer (odd or even integer). For example, ServiceApp1 with ServiceApp5, or ServiceApp2 with ServiceApp6.

**Warning** Although ServiceApp pairs work, the aggregate throughput for Inside-to-Outside and Outside-to-Inside traffic for the ServiceApp pair is halved.

**Caution** Do not pair ServiceApp<n> with ServiceApp<n+1>, where <n> is an even integer. When used, Outside-to-Inside traffic is dropped because traffic flows in the incorrect dispatcher and core.

Perform this task to configure the application service virtual interface (SVI) to forward data traffic.

## SUMMARY STEPS

1. **configure**
2. **interface ServiceApp** *value*
3. **service cgn** *instance -name* **service-type map-e**
4. **end**or **commit**

## DETAILED STEPS

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **interface ServiceApp** *value*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# interface ServiceApp1`<br>`ipv4 address 40.40.40.1 255.255.255.0`<br>`service cgn cgn1 service-type map-e`<br>`!`<br>`RP/0/RP0/CPU0:router(config)# interface ServiceApp2`<br>`ipv6 address 1001::101/32`<br>`service cgn cgn1 service-type map-e`<br>`!` | Configures the application SVI to 1, and enters interface configuration mode. |
| **Step 3** | **service cgn** *instance -name* **service-type map-e**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-if)# service cgn cgn1 service-type map-e map1` | Configures the application SVI to 1, and enters interface configuration mode. |
| **Step 4** | **end**or **commit**<br>**Example:** | Saves configuration changes. |

| | Command or Action | Purpose |
|---|---|---|
| | `RP/0/RP0/CPU0:router(config-cgn-map_e)# end`<br>or<br>`RP/0/RP0/CPU0:router(config-cgn-map_e)# commit` | • When you issue the end command, the system prompts you to commit changes:<br>`Uncommitted changes found, commit them before exiting (yes/no/cancel)?`<br>`[cancel]:`<br>• Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>• Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>• Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

# Configuring a MAP-E Instance

Perform this task to configure a MAP-E instance.

### SUMMARY STEPS

1. **configure**
2. **service cgn** *instance-name*
3. **service-type map-e** *instance-name*
4. **end**or **commit**

### DETAILED STEPS

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **service cgn** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# service cgn cgn1`<br>`RP/0/RP0/CPU0:router(config-cgn)#` | Configures the instance for the CGv6 application and enters CGv6 configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 3** | **service-type map-e** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# service-type map-e m1`<br>`RP/0/RP0/CPU0:router(config-cgn-map_e)#` | Configures the service type keyword definition for CGv6 MAP-E application. |
| **Step 4** | **end**or **commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-map_e)# end`<br>`or`<br>`RP/0/RP0/CPU0:router(config-cgn-map_e)# commit` | Saves configuration changes.<br>• When you issue the end command, the system prompts you to commit changes:<br>`Uncommitted changes found, commit them before exiting (yes/no/cancel)?`<br>`[cancel]:`<br>• Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>• Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>• Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

# Configuring the Policy Functions

Perform these tasks to configure the policy functions.

# Configuring Address Family

Perform these tasks to configure address family.

## Configuring IPv4 Address Family

Perform these tasks configure IPv4 address family for a MAP-E instance.

### Configuring IPv4 Interface

Perform this task to configure an IPv4 interface for a MAP-E instance.

#### SUMMARY STEPS

1. **configure**
2. **service cgn** *instance-name*

3. **service-type map-e** *instance-name*
4. **address-family ipv4 interface ServiceApp** *number*
5. **end**or **commit**

## DETAILED STEPS

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **service cgn** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-if)# service cgn cgn1` | Configures the instance named cgn1 for the CGv6 application, and enters CGv6 configuration mode. |
| **Step 3** | **service-type map-e** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# service-type map-e m1`<br>`RP/0/RP0/CPU0:router(config-cgn-map_e)` | Defines the service type keyword definition for the CGv6 MAP-E application. |
| **Step 4** | **address-family ipv4 interface ServiceApp** *number*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-map_e)# address-family ipv4 interface serviceApp 66`<br>`RP/0/RP0/CPU0:router(config-cgn-map_e-afi)#` | Configures the IPv4 interface to divert IPv4 map-e traffic. |
| **Step 5** | **end**or **commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-map_e-afi)# end`<br>`or`<br>`RP/0/RP0/CPU0:router(config-cgn-map_e-afi)#commit` | Saves configuration changes.<br>• When you issue the end command, the system prompts you to commit changes:<br>`Uncommitted changes found, commit them before exiting (yes/no/cancel)?`<br>`[cancel]:`<br>• Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>• Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>• Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. |

| | Command or Action | Purpose |
|---|---|---|
| | | • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

### Configuring TCP Maximum Segment Size

Perform this task to configure the Maximum Segment Size (MSS) for TCP.

### SUMMARY STEPS

1. **configure**
2. **service cgn** *instance-name*
3. **service-type map-e** *instance-name*
4. **address-family ipv4 tcp mss** *value*
5. **end**or **commit**

### DETAILED STEPS

#### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **service cgn** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# service cgn cgn1`<br>`RP/0/RP0/CPU0:router(config-cgn)#` | Configures the instance for the CGv6 application and enters CGv6 configuration mode. |
| **Step 3** | **service-type map-e** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# service-type map-e m1`<br>`RP/0/RP0/CPU0:router(config-cgn-map_e)#` | Defines the service type keyword definition for the CGv6 MAP-E application. |
| **Step 4** | **address-family ipv4 tcp mss** *value*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-map_e)# address-family ipv4 tcp mss 300`<br>`RP/0/RP0/CPU0:router(config-cgn-map_e-afi)#` | Configures the MSS to be used, in bytes. The range is from 28 to 1500. |
| **Step 5** | **end**or **commit**<br>**Example:** | Saves configuration changes.<br>• When you issue the end command, the system prompts you to commit changes: |

| | Command or Action | Purpose |
|---|---|---|
| | `RP/0/RP0/CPU0:router(config-cgn-map_e-afi)# end`<br>or<br>`RP/0/RP0/CPU0:router(config-cgn-map_e-afi)# commit` | `Uncommitted changes found, commit them before exiting (yes/no/cancel)? [cancel]:`<br>• Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>• Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>• Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

## Configuring IPv6 Address Family

Perform these tasks configure an IPv6 address family.

### Configuring IPv6 Interface

Perform this task to configure an IPv6 interface.

#### SUMMARY STEPS

1. **configure**
2. **service cgn** *instance-name*
3. **service-type map-e** *instance-name*
4. **address-family ipv6 interface ServiceApp** *number*
5. **end**or **commit**

#### DETAILED STEPS

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **service cgn** *instance-name*<br>**Example:** | Configures the instance for the CGv6 application and enters CGv6 configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| | `RP/0/RP0/CPU0:router(config)# service cgn cgn1`<br>`RP/0/RP0/CPU0:router(config-cgn)#` | |
| **Step 3** | **service-type map-e** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# service-type map-e m1`<br>`RP/0/RP0/CPU0:router(config-cgn-map_e)#` | Defines the service type keyword definition for the CGv6 MAP-E application. |
| **Step 4** | **address-family ipv6 interface ServiceApp** *number*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-map_e)# address-family ipv6 interface serviceApp 66`<br>`RP/0/RP0/CPU0:router(config-cgn-map_e-afi)#` | Configures the IPv6 interface to divert IPv6 map-e traffic. |
| **Step 5** | **end**or **commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-map_e-afi)# end`<br>`or`<br>`RP/0/RP0/CPU0:router(config-cgn-map_e-afi)# commit` | Saves configuration changes.<br>• When you issue the end command, the system prompts you to commit changes:<br>`Uncommitted changes found, commit them before exiting (yes/no/cancel)?`<br>`[cancel]:`<br>  • Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>  • Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>  • Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

### Configuring TCP Maximum Segment Size

Perform this task to configure the Maximum Segment Size (MSS) to be used for TCP.

### SUMMARY STEPS

1. **configure**
2. **service cgn** *instance-name*
3. **service-type map-e** *instance-name*
4. **address-family ipv6 tcp mss** *number*
5. **end**or **commit**

## DETAILED STEPS

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **service cgn** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# service cgn cgn1`<br>`RP/0/RP0/CPU0:router(config-cgn)#` | Configures the instance for the CGv6 application and enters CGv6 configuration mode. |
| **Step 3** | **service-type map-e** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# service-type map-e m1`<br>`RP/0/RP0/CPU0:router(config-cgn-map_e)#` | Defines the service type keyword definition for the CGv6 MAP-E application. |
| **Step 4** | **address-family ipv6 tcp mss** *number*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-map_e)# address-family ipv6 tcp mss 300`<br>`RP/0/RP0/CPU0:router(config-cgn-map_e-afi)#` | Configures the MSS to be used, in bytes. The range is from 28 to 1500. |
| **Step 5** | **end**or **commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-map_e-afi)# end`<br>`or`<br>`RP/0/RP0/CPU0:router(config-cgn-map_e-afi)# commit` | Saves configuration changes.<br>• When you issue the end command, the system prompts you to commit changes:<br>`Uncommitted changes found, commit them before exiting (yes/no/cancel)?`<br>`[cancel]:`<br>  • Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>  • Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>  • Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

# Configuring AFTR Endpoint Address

Perform this task to configure the Address Family Transition Router (AFTR) endpoint address.

## SUMMARY STEPS

1. **configure**
2. **service cgn** *instance-name*
3. **service-type map-e** *instance-name*
4. **aftr-endpoint-address** *IPv6 address*
5. **end**or **commit**

## DETAILED STEPS

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **service cgn** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# service cgn cgn1`<br>`RP/0/RP0/CPU0:router(config-cgn)#` | Configures the instance for the CGv6 application and enters CGv6 configuration mode. |
| **Step 3** | **service-type map-e** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# service-type map-e m1`<br>`RP/0/RP0/CPU0:router(config-cgn-map_e)#` | Defines the service type keyword definition for the CGv6 MAP-E application. |
| **Step 4** | **aftr-endpoint-address** *IPv6 address*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-map_e)# aftr-endpoint-address 2001:db8::32`<br>`RP/0/RP0/CPU0:router(config-cgn-map_e)#` | Configures the AFTR endpoint address. |
| **Step 5** | **end**or **commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-map_e)# end`<br>`or`<br>`RP/0/RP0/CPU0:router(config-cgn-map_e)# commit` | Saves configuration changes.<br>• When you issue the end command, the system prompts you to commit changes:<br>`Uncommitted changes found, commit them before exiting (yes/no/cancel)?`<br>`[cancel]:`<br>• Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. |

| | Command or Action | Purpose |
|---|---|---|
| | | • Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>• Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

# Configuring Contiguous Ports

Perform this task to configure the number of contiguous ports for a MAP-E instance

### SUMMARY STEPS

1. **configure**
2. **service cgn** *instance-name*
3. **service-type map-e** *instance-name*
4. **contiguous-ports** *number*
5. **end**or **commit**

### DETAILED STEPS

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **service cgn** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# service cgn cgn1`<br>`RP/0/RP0/CPU0:router(config-cgn)#` | Configures the instance for the CGv6 application and enters CGv6 configuration mode. |
| **Step 3** | **service-type map-e** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# service-type map-e m1`<br>`RP/0/RP0/CPU0:router(config-cgn-map_e)#` | Defines the service type keyword definition for the CGv6 MAP-E application. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | **contiguous-ports** *number*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-map_e)# contiguous-ports 16`<br>`RP/0/RP0/CPU0:router(config-cgn-map_e)#` | Configures the number of contiguous ports. The range is from 1 to 65536.<br>**Note**<br>The value is expressed in powers of 2. |
| **Step 5** | **end**or **commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-map_e)# end`<br>`or`<br>`RP/0/RP0/CPU0:router(config-cgn-map_e)# commit` | Saves configuration changes.<br>• When you issue the end command, the system prompts you to commit changes:<br>`Uncommitted changes found, commit them before exiting (yes/no/cancel)?`<br>`[cancel]:`<br>• Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>• Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>• Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

# Configuring CPE Domain Parameters

Perform this task to configure Customer Premise Equipment (CPE) domain parameters.

### SUMMARY STEPS

1. **configure**
2. **service cgn** *instance-name*
3. **service-type map-e** *instance-name*
4. **cpe-domain ipv4 prefix** *ipv4 address/prefix* or **cpe-domain ipv6 prefix** *ipv6 address/prefix*
5. **end**or **commit**

## DETAILED STEPS

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **service cgn** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# service cgn cgn1`<br>`RP/0/RP0/CPU0:router(config-cgn)#` | Configures the instance for the CGv6 application and enters CGv6 configuration mode. |
| **Step 3** | **service-type map-e** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# service-type map-e m1`<br>`RP/0/RP0/CPU0:router(config-cgn-map_e)#` | Defines the service type keyword definition for the CGv6 MAP-E application. |
| **Step 4** | **cpe-domain ipv4 prefix** *ipv4 address/prefix* or<br>**cpe-domain ipv6 prefix** *ipv6 address/prefix*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-map_e)# cpe-domain ipv4 prefix 10.2.2.24/2`<br>`RP/0/RP0/CPU0:router(config-cgn-map_e)#`<br>`or`<br>`RP/0/RP0/CPU0:router(config-cgn-map_e)# cpe-domain ipv6 prefix 2001:da8:a464::/48`<br>`RP/0/RP0/CPU0:router(config-cgn-map_e)#` | Configures the IPv4 or IPv6 prefixes of the CPE domain. |
| **Step 5** | **end**or **commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-map_e)# end`<br>`or`<br>`RP/0/RP0/CPU0:router(config-cgn-map_e)# commit` | Saves configuration changes.<br>• When you issue the end command, the system prompts you to commit changes:<br>`Uncommitted changes found, commit them before exiting (yes/no/cancel)?`<br>`[cancel]:`<br>  • Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>  • Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>  • Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. |

| | Command or Action | Purpose |
|---|---|---|
| | | • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

# Configuring Path MTU of the Tunnel

Perform this task to configure the path Maximum Transmission Unit (MTU) of the tunnel.

### SUMMARY STEPS

1. **configure**
2. **service cgn** *instance-name*
3. **service-type map-e** *instance-name*
4. **path-mtu** *value*
5. **end**or **commit**

### DETAILED STEPS

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **service cgn** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# service cgn cgn1`<br>`RP/0/RP0/CPU0:router(config-cgn)#` | Configures the instance for the CGv6 application and enters CGv6 configuration mode. |
| **Step 3** | **service-type map-e** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# service-type map-e m1`<br>`RP/0/RP0/CPU0:router(config-cgn-map_e)#` | Defines the service type keyword definition for the CGv6 MAP-E application. |
| **Step 4** | **path-mtu** *value*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-map_e)# path-mtu 1300`<br>`RP/0/RP0/CPU0:router(config-cgn-map_e)#` | Configures the path MTU of the tunnel. The range is from 1280 to 9216. |
| **Step 5** | **end**or **commit**<br>**Example:** | Saves configuration changes. |

| | Command or Action | Purpose |
|---|---|---|
| | `RP/0/RP0/CPU0:router(config-cgn-map_e)# end`<br>or<br>`RP/0/RP0/CPU0:router(config-cgn-map_e)# commit` | • When you issue the end command, the system prompts you to commit changes:<br>`Uncommitted changes found, commit them before exiting (yes/no/cancel)?`<br>`[cancel]:`<br>  • Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>  • Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>  • Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

# Configuring Port Sharing Ratio

Perform this task to configure the sharing ratio of the port.

### SUMMARY STEPS

1. **configure**
2. **service cgn** *instance-name*
3. **service-type map-e** *instance-name*
4. **sharing-ratio** *number*
5. **end**or **commit**

### DETAILED STEPS

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **service cgn** *instance-name*<br>**Example:** | Configures the instance for the CGv6 application and enters CGv6 configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| | `RP/0/RP0/CPU0:router(config)# service cgn cgn1`<br>`RP/0/RP0/CPU0:router(config-cgn)#` | |
| **Step 3** | **service-type map-e** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# service-type map-e m1`<br>`RP/0/RP0/CPU0:router(config-cgn-map_e)#` | Defines the service type keyword definition for the CGv6 MAP-E application. |
| **Step 4** | **sharing-ratio** *number*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-map_e)# sharing-ratio 64`<br>`RP/0/RP0/CPU0:router(config-cgn-map_e)#` | Configures the port sharing ratio. The range is from 1 to 32768.<br>**Note**<br>The value is expressed in powers of 2. |
| **Step 5** | **end**or **commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-map_e)# end`<br>`or`<br>`RP/0/RP0/CPU0:router(config-cgn-map_e)# commit` | Saves configuration changes.<br>• When you issue the end command, the system prompts you to commit changes:<br>`Uncommitted changes found, commit them before exiting (yes/no/cancel)?`<br>`[cancel]:`<br>  • Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>  • Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>  • Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

# MAP-E on VSM Configuration: Example

This example shows a sample MAP-E configuration on VSM:

```
interface ServiceApp1
ipv4 address 30.30.30.1 255.255.255.0
service cgn cgn1 service-type map-e m1
!
interface ServiceApp2
ipv4 address 19.1.1.1 255.255.255.252
ipv6 address 2001:101::/32
```

```
service cgn cgn1 service-type map-e m1
!
interface ServiceInfra1
ipv4 address 200.1.1.1 255.255.255.0
service-location 0/0/CPU0
!
router static
address-family ipv4 unicast
202.38.102.0/24 ServiceApp1 30.30.30.2
!
address-family ipv6 unicast
2001:da8:a464:ffff::/64 ServiceApp2 2001:101::2
!
service cgn cgn1
service-location preferred-active 0/0/CPU0
service-type map-e m1
cpe-domain ipv6 prefix 2001:da8:a464::/48
cpe-domain ipv4 prefix 202.38.102.0/24
aftr-endpoint-address 2001:da8:a464:ffff::/128
sharing-ratio 16
contiguous-ports 32
path-mtu 1300

address-family ipv4
interface ServiceApp1
tcp mss 235

!
address-family ipv6
interface ServiceApp2
tcp mss 1154
!
!
```

# Mapping of Address and Port-Translation Mode

Mapping of Address and Port-Translation Mode (MAP-T) is a CGN solution that enables IPv4-only clients to communicate with IPv6-only resources using address and packet translation. MAP-T is also referred to as Dual IVI (dIVI) or Stateless NAT46. This enables a service provider to offer IPv4 services to IPv6 enabled (customer) sites to which it provides customer premise equipment (CPE). This approach utilizes stateless IPv4 to IPv6 translation (that is NAT64) to transit IPv6-enabled network infrastructure. The provider access network can now be on IPv6, while customers use IPv6 and IPv4 services simultaneously. MAP-T keeps the stateful NAT44 on CPE, as usual, to handle IPv4 address exhaustion, in addition to stateless NAT64 on CPE and Border Router.

MAP-T is attractive to those SPs who have deployed, or are planning to deploy IPv6 end-to-end services, and want to manage IPv4 address exhaustion with utmost predictability.

MAP-T is a preferred alternate to DS-Lite in a sevice provider network when there is no tunneling needed.

**Note**

- MAP-T is offered in stateless mode only.
- If the cumulative sum of EA bit value and cpe-domain ipv6 prefix value is more than 64 for an interface, then the traffic is dropped for that interface. EA bit value is calculated as (32- ipv4 prefix) + X), where 32 is a constant and X is the power value of 2 when the sharing ratio is expressed as $2^x$.

  Ensure that the EA bit value is less than or equal to 32 and sharing ratio is less than or equal to 256 to avoid any traffic drops.
- You should not modify or delete the existing MAP-E and MAP-T configuration. If you modify the existing configuration, the changes are not reflected in the PBR policy.

  To update the configuration, delete the existing MAP-E and MAP-T instance and add new instance with the required changes.
- Do not configure MAP-T and MAP-E on the same CGv6 instance simultaneously.

# Configuring MAP-T

Perform these tasks to configure MAP-T.

**Note** MAP-T is supported only on Cisco ASR 9000 Series 400G and 200G Modular Line Cards and Cisco ASR 9000 Series 4-Port and 8-Port 100 Gigabit Ethernet Line Cards.

## Configuring the Application Service Virtual Interface

This section lists the guidelines for selecting service application interfaces for MAP-T.

- Pair ServiceApp<n> with ServiceApp<n+1>, where <n> is an odd integer. This is to ensure that the ServiceApp pairs works with a maximum throughput. For example, ServiceApp1 with ServiceApp2 or ServiceApp3 with ServiceApp4.
- Pair ServiceApp<n> with ServiceApp<n+5> or ServiceApp<n+9>, and so on, where <n> is an odd integer. For example, ServiceApp1 with ServiceApp6, ServiceApp1 with ServiceApp10, ServiceApp3 with ServiceApp8, or ServiceApp3 with ServiceApp12.
- Pair ServiceApp<n> with ServiceApp<n+4>, where <n> is an integer (odd or even integer). For example, ServiceApp1 with ServiceApp5, or ServiceApp2 with ServiceApp6.

**Warning** Although ServiceApp pairs work, the aggregate throughput for Inside-to-Outside and Outside-to-Inside traffic for the ServiceApp pair is halved.

**Caution** Do not pair ServiceApp<n> with ServiceApp<n+1>, where <n> is an even integer. When used, Outside-to-Inside traffic is dropped because traffic flows in the incorrect dispatcher and core.

Perform this task to configure the application service virtual interface (SVI) to forward data traffic.

## SUMMARY STEPS

1. **configure**
2. **interface ServiceApp** *value*
3. **service cgn** *instance-name* **service-type map-t**
4. **end**or **commit**

## DETAILED STEPS

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **interface ServiceApp** *value*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# interface ServiceApp 1`<br>`RP/0/RP0/CPU0:router(config-if)#` | Configures the application SVI to 1, and enters interface configuration mode. |
| **Step 3** | **service cgn** *instance-name* **service-type map-t**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-if)# service cgn cgn1 service-type map-t map1` | Configures the application SVI to 1, and enters interface configuration mode. |
| **Step 4** | **end**or **commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-if)# end`<br>`or`<br>`RP/0/RP0/CPU0:router(config-if)# commit` | Saves configuration changes.<br>• When you issue the end command, the system prompts you to commit changes:<br>`Uncommitted changes found, commit them before exiting (yes/no/cancel)?`<br>`[cancel]:`<br>• Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>• Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes. |

| | Command or Action | Purpose |
|---|---|---|
| | | • Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

# Configuring a MAP-T Instance

Perform this task to configure a MAP-T instance.

### SUMMARY STEPS

1. **configure**
2. **service cgn** *instance-name*
3. **service-type map-t** *instance-name*
4. **end** or **commit**

### DETAILED STEPS

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **service cgn** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# service cgn cgn1`<br>`RP/0/RP0/CPU0:router(config-cgn)#` | Configures the instance named cgn1 for the CGv6 application and enters CGv6 configuration mode. |
| **Step 3** | **service-type map-t** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# service-type map-t map-t-inst`<br>`RP/0/RP0/CPU0:router(config-cgn-mapt)#` | Configures the service type keyword definition for CGv6 MAP-T application |
| **Step 4** | **end** or **commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-mapt)# end`<br>`or`<br>`RP/0/RP0/CPU0:router(config-cgn-mapt)# commit` | Saves configuration changes.<br>• When you issue the **end** command, the system prompts you to commit changes:<br>*Uncommitted changes found, commit them before exiting (yes/no/cancel)?* |

| | Command or Action | Purpose |
|---|---|---|
| | | *[cancel]:*<br>• Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>• Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>• Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

# Configuring the Policy Functions

Perform these tasks to configure the policy functions.

## Configuring Address Family

Perform these tasks to configure address family.

### Configuring IPv4 Address Family

Perform these tasks configure IPv4 address family for a MAP-T instance.

*Configuring an IPv4 Interface*

Perform this task to configure an IPv4 interface for a MAP-T instance.

**SUMMARY STEPS**

1. **configure**
2. **service cgn** *instance-name*
3. **service-type map-t** *instance-name*
4. **address-family ipv4 interface ServiceApp** *number*
5. **end**or **commit**

## DETAILED STEPS

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **service cgn** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# service cgn cgn1`<br>`RP/0/RP0/CPU0:router(config-cgn)#` | Configures the instance for the CGv6 application and enters CGv6 configuration mode. |
| **Step 3** | **service-type map-t** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# service-type map-t map-t-inst`<br>`RP/0/RP0/CPU0:router(config-cgn-mapt)#` | Configures the service type keyword definition for CGv6 MAP-T application. |
| **Step 4** | **address-family ipv4 interface ServiceApp** *number*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-mapt)#address-family ipv4 interface serviceApp 66`<br>`RP/0/RP0/CPU0:router(config-cgn-mapt-afi)` | Configures the IPv4 interface to divert IPv4 MAP-T traffic. |
| **Step 5** | **end**or **commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-mapt-afi)# end`<br>`or`<br>`RP/0/RP0/CPU0:router(config-cgn-mapt-afi)# commit` | Saves configuration changes.<br>• When you issue the end command, the system prompts you to commit changes:<br>`Uncommitted changes found, commit them before exiting (yes/no/cancel)?`<br>`[cancel]:`<br>  • Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>  • Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>  • Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

*Configuring IPv4 TCP Maximum Segment Size (MSS)*

Perform this task to configure the MSS for TCP in bytes.

**SUMMARY STEPS**

1. **configure**
2. **service cgn** *instance-name*
3. **service-type map-t** *instance-name*
4. **address-family ipv4 tcp mss** *value*
5. **end**or **commit**

**DETAILED STEPS**

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **service cgn** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# service cgn cgn1`<br>`RP/0/RP0/CPU0:router(config-cgn)#` | Configures the instance for the CGv6 application and enters CGv6 configuration mode. |
| **Step 3** | **service-type map-t** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# service-type map-t map-t-inst`<br>`RP/0/RP0/CPU0:router(config-cgn-mapt)#` | Configures the service type keyword definition for CGv6 MAP-T application. |
| **Step 4** | **address-family ipv4 tcp mss** *value*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-mapt)#address-family ipv4 tcp mss 66`<br>`RP/0/RP0/CPU0:router(config-cgn-mapt-afi)` | Configures the MSS for TCP in bytes. |
| **Step 5** | **end**or **commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-mapt-afi)# end`<br>`or`<br>`RP/0/RP0/CPU0:router(config-cgn-mapt-afi)# commit` | Saves configuration changes.<br>• When you issue the end command, the system prompts you to commit changes:<br>`Uncommitted changes found, commit them before exiting (yes/no/cancel)?`<br>`[cancel]:`<br>• Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. |

| | Command or Action | Purpose |
|---|---|---|
| | | • Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>• Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

*Configuring IPv4 Type of Service (ToS)*

Perform this task to configure the configured ToS value to be used when translating a packet from IPv6 to IPv4.

**SUMMARY STEPS**

1. **configure**
2. **service cgn** *instance-name*
3. **service-type map-t** *instance-name*
4. **address-family ipv4 tos** *value*
5. **end**or **commit**

**DETAILED STEPS**

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **service cgn** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# service cgn cgn1`<br>`RP/0/RP0/CPU0:router(config-cgn)#` | Configures the instance for the CGv6 application and enters CGv6 configuration mode. |
| **Step 3** | **service-type map-t** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# service-type map-t map-t-inst`<br>`RP/0/RP0/CPU0:router(config-cgn-mapt)#` | Configures the service type keyword definition for CGv6 MAP-T application. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | **address-family ipv4 tos** *value*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-mapt)#address-family ipv4 tos 66`<br>`RP/0/RP0/CPU0:router(config-cgn-mapt-afi)` | Configures the TOS value. |
| **Step 5** | **end**or **commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-mapt-afi)# end`<br>or<br>`RP/0/RP0/CPU0:router(config-cgn-mapt-afi)# commit` | Saves configuration changes.<br>• When you issue the end command, the system prompts you to commit changes:<br>`Uncommitted changes found, commit them before exiting (yes/no/cancel)?`<br>`[cancel]:`<br>  • Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>  • Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>  • Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

## Configuring IPv6 Address Family

Perform these tasks configure an IPv6 address family.

### *Configuring IPv6 Do not Fragment (DF) Override*

Perform this task to enable DF override configuration.

### SUMMARY STEPS

1. **configure**
2. **service cgn** *instance-name*
3. **service-type map-t** *instance-name*
4. **address-family ipv6 df-override**
5. **end**or **commit**

## DETAILED STEPS

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **service cgn** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# service cgn cgn1`<br>`RP/0/RP0/CPU0:router(config-cgn)#` | Configures the instance for the CGv6 application and enters CGv6 configuration mode. |
| **Step 3** | **service-type map-t** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# service-type map-t map-t-inst`<br>`RP/0/RP0/CPU0:router(config-cgn-mapt)` | Configures the service type keyword definition for CGv6 MAP-T application. |
| **Step 4** | **address-family ipv6 df-override**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-mapt)#address-family ipv6 df-override`<br>`RP/0/RP0/CPU0:router(config-cgn-mapt-afi)` | Configures the DF-Override. |
| **Step 5** | **end**or **commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-mapt-afi)# end`<br>`or`<br>`RP/0/RP0/CPU0:router(config-cgn-mapt-afi)# commit` | Saves configuration changes.<br>• When you issue the end command, the system prompts you to commit changes:<br>`Uncommitted changes found, commit them before exiting (yes/no/cancel)?`<br>`[cancel]:`<br>  • Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>  • Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>  • Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

*Configuring an IPv6 Interface*

Perform this task to configure an IPv6 interface for a stateful NAT64 instance.

**SUMMARY STEPS**

1. **configure**
2. **service cgn** *instance-name*
3. **service-type nat64 stateful** *instance-name*
4. **address-family ipv6 interface ServiceApp** *number*
5. **end**or **commit**

**DETAILED STEPS**

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **service cgn** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# service cgn cgn1`<br>`RP/0/RP0/CPU0:router(config-cgn)#` | Configures the instance for the CGv6 application and enters CGv6 configuration mode. |
| **Step 3** | **service-type nat64 stateful** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# service-type nat64 stateful nat64-inst`<br>`RP/0/RP0/CPU0:router(config-cgn-nat64-stateful)` | Configures the service type keyword definition for CGv6 Stateful NAT64 application. |
| **Step 4** | **address-family ipv6 interface ServiceApp** *number*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-nat64-stateful)#address-family ipv4 interface ServiceApp 66`<br>`RP/0/RP0/CPU0:router(config-cgn-nat64-stful-afi)` | Configures the IPv6 interface to divert IPv6 nat64 traffic. |
| **Step 5** | **end**or **commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-nat64-stful-afi)# end`<br>`or`<br>`RP/0/RP0/CPU0:router(config-cgn-nat64-stful-afi)# commit` | Saves configuration changes.<br>• When you issue the end command, the system prompts you to commit changes:<br>`Uncommitted changes found, commit them before exiting (yes/no/cancel)?`<br>`[cancel]:`<br>• Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode. |

| | Command or Action | Purpose |
|---|---|---|
| | | • Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>• Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

*Configuring IPv6 TCP Maximum Segment Size (MSS)*

Perform this task to configure the MSS for TCP in bytes.

**SUMMARY STEPS**

1. **configure**
2. **service cgn** *instance-name*
3. **service-type map-t** *instance-name*
4. **address-family ipv6 tcp mss** *value*
5. **end**or **commit**

**DETAILED STEPS**

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **service cgn** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# service cgn cgn1`<br>`RP/0/RP0/CPU0:router(config-cgn)#` | Configures the instance for the CGv6 application and enters CGv6 configuration mode. |
| **Step 3** | **service-type map-t** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# service-type map-t map-t-inst`<br>`RP/0/RP0/CPU0:router(config-cgn-mapt)` | Configures the service type keyword definition for CGv6 MAP-T application. |
| **Step 4** | **address-family ipv6 tcp mss** *value*<br>**Example:** | Configures the MSS for TCP in bytes. |

| | Command or Action | Purpose |
|---|---|---|
| | `RP/0/RP0/CPU0:router(config-cgn-mapt)#address-family ipv6 tcp mss 66`<br>`RP/0/RP0/CPU0:router(config-cgn-mapt-afi)` | |
| **Step 5** | **end**or **commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-mapt-afi)# end`<br>`or`<br>`RP/0/RP0/CPU0:router(config-cgn-mapt-afi)# commit` | Saves configuration changes.<br>• When you issue the end command, the system prompts you to commit changes:<br>`Uncommitted changes found, commit them before exiting (yes/no/cancel)?`<br>`[cancel]:`<br>  • Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>  • Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>  • Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

*Configuring IPv6 Traffic-Class*

Perform this task to configure a traffic-class.

**SUMMARY STEPS**

1. **configure**
2. **service cgn** *instance-name*
3. **service-type nat64 stateful** *instance-name*
4. **address-family ipv6 traffic-class** *value*
5. **end**or **commit**

**DETAILED STEPS**

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 2** | **service cgn** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# service cgn cgn1`<br>`RP/0/RP0/CPU0:router(config-cgn)#` | Configures the instance for the CGv6 application and enters CGv6 configuration mode. |
| **Step 3** | **service-type nat64 stateful** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# service-type nat64 stateful nat64-inst`<br>`RP/0/RP0/CPU0:router(config-cgn-nat64-stateful)` | Configures the service type keyword definition for CGv6 Stateful NAT64 application. |
| **Step 4** | **address-family ipv6 traffic-class** *value*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-nat64-stateful)#address-family ipv6 traffic-class 66`<br>`RP/0/RP0/CPU0:router(config-cgn-nat64-stful-afi)` | Configures the traffic class to be set. |
| **Step 5** | **end**or **commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-nat64-stful-afi)# end`<br>`or`<br>`RP/0/RP0/CPU0:router(config-cgn-nat64-stful-afi)# commit` | Saves configuration changes.<br>• When you issue the end command, the system prompts you to commit changes:<br>`Uncommitted changes found, commit them before exiting (yes/no/cancel)?`<br>`[cancel]:`<br>  • Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>  • Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>  • Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

### *Configuring Contiguous Ports*

Perform this task to configure contiguous ports.

**SUMMARY STEPS**

1. **configure**
2. **service cgn** *instance-name*
3. **service-type map-t** *instance-name*

4. **contiguous-ports** *number*
5. **end**or **commit**

## DETAILED STEPS

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **service cgn** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# service cgn cgn1`<br>`RP/0/RP0/CPU0:router(config-cgn)#` | Configures the instance for the CGv6 application and enters CGv6 configuration mode. |
| **Step 3** | **service-type map-t** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# service-type map-t map-t-inst`<br>`RP/0/RP0/CPU0:router(config-cgn-mapt)` | Configures the service type keyword definition for CGv6 MAP-T application. |
| **Step 4** | **contiguous-ports** *number*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-mapt)#contiguous-ports 14`<br>`RP/0/RP0/CPU0:router(config-cgn-mapt)` | Configures the number of ports and the value is expressed in powers of 2. The range is from 1 to 65536. |
| **Step 5** | **end**or **commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-mapt)# end`<br>`or`<br>`RP/0/RP0/CPU0:router(config-cgn-mapt)# commit` | Saves configuration changes.<br>• When you issue the end command, the system prompts you to commit changes:<br>`Uncommitted changes found, commit them before exiting (yes/no/cancel)?`<br>`[cancel]:`<br>  • Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>  • Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>  • Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes. |

| | Command or Action | Purpose |
|---|---|---|
| | | • Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

*Configuring Customer Premise Equipment Domain Parameters*

Perform this task to configure Customer Premise Equipment (CPE) domain parameters.

**SUMMARY STEPS**

1. **configure**
2. **service cgn** *instance-name*
3. **service-type map-t** *instance-name*
4. **cpe-domain ipv4 prefix** *ipv4 address/prefix* or **cpe-domain ipv6 prefix** *ipv6 address/prefix*
5. **end**or **commit**

**DETAILED STEPS**

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **service cgn** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# service cgn cgn1`<br>`RP/0/RP0/CPU0:router(config-cgn)#` | Configures the instance for the CGv6 application and enters CGv6 configuration mode. |
| **Step 3** | **service-type map-t** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# service-type map-t map-t-inst`<br>`RP/0/RP0/CPU0:router(config-cgn-mapt)` | Configures the service type keyword definition for CGv6 MAP-T application. |
| **Step 4** | **cpe-domain ipv4 prefix** *ipv4 address/prefix* or **cpe-domain ipv6 prefix** *ipv6 address/prefix*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-mapt)#acpe-domain ipv4 prefix 10.2.2.24/2`<br>`RP/0/RP0/CPU0:router(config-cgn-mapt)`<br>`or`<br>`RP/0/RP0/CPU0:router(config-cgn-mapt)#acpe-domain ipv6 prefix 10:2::2/24`<br>`RP/0/RP0/CPU0:router(config-cgn-mapt)` | Configures the cpe domain parameters. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 5** | **end**or **commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-mapt)# end`<br>`or`<br>`RP/0/RP0/CPU0:router(config-cgn-mapt)# commit` | Saves configuration changes.<br>• When you issue the end command, the system prompts you to commit changes:<br>`Uncommitted changes found, commit them before exiting (yes/no/cancel)?`<br>`[cancel]:`<br>  • Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>  • Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>  • Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

*Configuring External Domain Parameters*

Perform this task to configure external domain parameters.

## SUMMARY STEPS

1. **configure**
2. **service cgn** *instance-name*
3. **service-type map-t** *instance-name*
4. **external-domain ipv6 prefix** *ipv6 address/prefix*
5. **end**or **commit**

## DETAILED STEPS

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **service cgn** *instance-name*<br>**Example:** | Configures the instance for the CGv6 application and enters CGv6 configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| | `RP/0/RP0/CPU0:router(config)# service cgn cgn1`<br>`RP/0/RP0/CPU0:router(config-cgn)#` | |
| **Step 3** | **service-type map-t** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# service-type map-t map-t-inst`<br>`RP/0/RP0/CPU0:router(config-cgn-mapt)` | Configures the service type keyword definition for CGv6 MAP-T application. |
| **Step 4** | **external-domain ipv6 prefix** *ipv6 address/prefix*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-mapt)#external-domain ipv6 prefix 10:2::2/24`<br>`RP/0/RP0/CPU0:router(config-cgn-mapt)` | Configures the external domain parameters. |
| **Step 5** | **end**or **commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-mapt)# end`<br>`or`<br>`RP/0/RP0/CPU0:router(config-cgn-mapt)# commit` | Saves configuration changes.<br>• When you issue the end command, the system prompts you to commit changes:<br>`Uncommitted changes found, commit them before exiting (yes/no/cancel)?`<br>`[cancel]:`<br>  • Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>  • Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>  • Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

*Configuring Port Sharing Ratio*

Perform this task to configure port sharing ratio.

**SUMMARY STEPS**

1. **configure**
2. **service cgn** *instance-name*
3. **service-type map-t** *instance-name*
4. **sharing-ratio** *number*
5. **end**or **commit**

## DETAILED STEPS

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **service cgn** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# service cgn cgn1`<br>`RP/0/RP0/CPU0:router(config-cgn)#` | Configures the instance for the CGv6 application and enters CGv6 configuration mode. |
| **Step 3** | **service-type map-t** *instance-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn)# service-type map-t map-t-inst`<br>`RP/0/RP0/CPU0:router(config-cgn-mapt)` | Configures the service type keyword definition for CGv6 MAP-T application. |
| **Step 4** | **sharing-ratio** *number*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-mapt)#sharing-ratio 14`<br>`RP/0/RP0/CPU0:router(config-cgn-mapt)` | Configures the port sharing ratio and the value is expressed in powers of 2. The range is from 1 to 32768. |
| **Step 5** | **end**or **commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-cgn-mapt)# end`<br>`or`<br>`RP/0/RP0/CPU0:router(config-cgn-mapt)# commit` | Saves configuration changes.<br>• When you issue the end command, the system prompts you to commit changes:<br>`Uncommitted changes found, commit them before exiting (yes/no/cancel)?`<br>`[cancel]:`<br>  • Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>  • Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>  • Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

# MAP-T Configuration on VSM: Example

```
interface ServiceApp4
 ipv4 address 30.30.30.1 255.255.255.0
 service cgn test service-type map-t
!
interface ServiceApp6
 ipv4 address 19.1.1.1 255.255.255.252
 ipv6 address 2001:101::/32
 service cgn test service-type map-t
!
interface ServiceInfra1
 ipv4 address 200.1.1.1 255.255.255.0
 service-location 0/0/CPU0
!
router static
 address-family ipv4 unicast
202.38.102.0/24 ServiceApp4 30.30.30.2
  !
 address-family ipv6 unicast
  2001:da8:a464:ffff::/64 ServiceApp6 2001:101::2
!
service cgn test
 service-location preferred-active 0/0/CPU0
service-type map-t xlat1
  cpe-domain ipv6 prefix 2001:da8:a464::/48
  cpe-domain ipv4 prefix  202.38.102.0/24
  external-domain ipv6 prefix 2001:da8:a464:ffff::/64
  sharing-ratio 64
  contiguous-ports 128

       address-family ipv4
   interface ServiceApp4
 tcp mss 235
 tos 100
  !
      address-family ipv6
   interface ServiceApp6
  tcp mss 1154
 traffic-class 100
 df-override
;
 !
!
```

# Configuration Examples for Implementing the Carrier Grade NAT

This section provides the following configuration examples for CGN:

**Note** Do not configure multiple outside address-pools to be mapped to a single inside-vrf. If you have multiple outside address-pools to be mapped, then create multiple inside-vrfs and map each outside address-pool to a single inside-vrf inside the NAT44 configuration.

# Configuring a Different Inside VRF Map to a Different Outside VRF: Example

This example shows how to configure a different inside VRF map to a different outside VRF and different outside address pools:

```
service cgn cgn1
inside-vrf insidevrf1
map outside-vrf outsidevrf1 address-pool 100.1.1.0/24
!
!
inside-vrf insidevrf2
map outside-vrf outsidevrf2 address-pool 100.1.2.0/24
!
service-location preferred-active 0/2/cpu0 preferred-standby 0/3/cpu0
!
interface ServiceApp 1
vrf insidevrf1
ipv4 address 210.1.1.1 255.255.255.0
service cgn cgn1
!
router static
vrf insidevrf1
0.0.0.0/0 serviceapp 1
!
!
interface ServiceApp 2
vrf insidevrf2
ipv4 address 211.1.1.1 255.255.255.0
service cgn cgn1
service-type nat44 nat1
!
router static
vrf insidevrf2
0.0.0.0/0 serviceapp 2
!
!
interface ServiceApp 3
vrf outsidevrf1
ipv4 address 1.1.1.1 255.255.255.0
service cgn cgn1
service-type nat44 nat1
!
router static
vrf outsidevrf1
100.1.1.0/24 serviceapp 3
!
!
interface ServiceApp 4
vrf outsidevrf2
ipv4 address 2.2.2.1 255.255.255.0
service cgn cgn1
service-type nat44 nat1
!
router static
vrf outsidevrf2
100.1.2.0/24 serviceapp 4
!
```

# Configuring a Different Inside VRF Map to a Same Outside VRF: Example

This example shows how to configure a different inside VRF map to the same outside VRF but with different outside address pools:

```
service cgn cgn1
inside-vrf insidevrf1
map outside-vrf outsidevrf1 address-pool 100.1.1.0/24
!
inside-vrf insidevrf2
map outside-vrf outsidevrf1 address-pool 200.1.1.0/24
!
!
service-location preferred-active 0/2/cpu0 preferred-standby 0/3/cpu0
!
interface ServiceApp 1
vrf insidevrf1
ipv4 address 1.1.1.1 255.255.255.0
service cgn cgn1
!
router static
vrf insidevrf1
0.0.0.0/0 serviceapp 1
!
!
interface ServiceApp 2
vrf insidevrf2
ipv4 address 2.1.1.1 255.255.255.0
service cgn cgn1
!
router static
vrf insidevrf2
0.0.0.0/0 serviceapp 2
!
!
interface ServiceApp 3
vrf outsidevrf1
ipv4 address 100.1.1.1 255.255.255.0
service cgn cgn1
!
router static
vrf outsidevrf1
100.1.1.0/24 serviceapp 3
200.1.1.0/24 serviceapp 3
!
```

# NAT44 Configuration: Example

This example shows a NAT44 sample configuration:

```
IPv4: 40.22.22.22/16
!
interface Loopback40
 description IPv4 Host for NAT44
 ipv4 address 40.22.22.22 255.255.0.0
!
interface Loopback41
 description IPv4 Host for NAT44
 ipv4 address 41.22.22.22 255.255.0.0
!
interface GigabitEthernet0/3/0/0.1
 description Connected to P2_CRS-8 GE 0/6/5/0.1
```

```
 ipv4 address 10.222.5.22 255.255.255.0
 encapsulation dot1q 1
!
router static
 address-family ipv4 unicast
  180.1.0.0/16 10.222.5.2
  181.1.0.0/16 10.222.5.2
!
!
Hardware Configuration for CSGE:
!
vrf InsideCustomer1
 address-family ipv4 unicast
 !
!
vrf OutsideCustomer1
 address-family ipv4 unicast
 !
!
hw-module service cgn location 0/3/CPU0
!
service-plim-ha location 0/3/CPU0 datapath-test
service-plim-ha location 0/3/CPU0 core-to-core-test
service-plim-ha location 0/3/CPU0 pci-test
service-plim-ha location 0/3/CPU0 coredump-extraction
!
!
interface GigabitEthernet0/6/5/0.1
 vrf InsideCustomer1
 ipv4 address 10.222.5.2 255.255.255.0
 encapsulation dot1q 1
!
interface GigabitEthernet0/6/5/1.1
 vrf OutsideCustomer1
 ipv4 address 10.12.13.2 255.255.255.0
 encapsulation dot1q 1
!
interface ServiceApp1
 vrf InsideCustomer1
 ipv4 address 1.1.1.1 255.255.255.252
 service cgn cgn1 service-type nat44
!
interface ServiceApp2
 vrf OutsideCustomer1
 ipv4 address 2.1.1.1 255.255.255.252
 service cgn cgn1 service-type nat44
!
interface ServiceInfra1
 ipv4 address 75.75.75.75 255.255.255.0
 service-location 0/3/CPU0
!
!
router static
 !
vrf InsideCustomer1
  address-family ipv4 unicast
   0.0.0.0/0 ServiceApp1
   40.22.0.0/16 10.222.5.22
   41.22.0.0/16 10.222.5.22
   181.1.0.0/16 vrf OutsideCustomer1 GigabitEthernet0/6/5/1.1 10.12.13.1
  !
 !
 vrf OutsideCustomer1
  address-family ipv4 unicast
```

```
   40.22.0.0/16 vrf InsideCustomer1 GigabitEthernet0/6/5/0.1 10.222.5.22
   41.22.0.0/16 vrf InsideCustomer1 GigabitEthernet0/6/5/0.1 10.222.5.22
   100.0.0.0/24 ServiceApp2
   180.1.0.0/16 10.12.13.1
   181.1.0.0/16 10.12.13.1
  !
 !
!
CGSE Configuration:
service cgn cgn1
 service-location preferred-active 0/3/CPU0
 service-type nat44 nat44
  portlimit 200
  alg ActiveFTP
  inside-vrf InsideCustomer1
   map outside-vrf OutsideCustomer1 address-pool 100.0.0.0/24
   protocol tcp
    static-forward inside
     address 41.22.22.22 port 80
    !
   !
   protocol icmp
    static-forward inside
     address 41.22.22.22 port 80
    !
   !
   external-logging netflow version 9
    server
     address 172.29.52.68 port 2055
     refresh-rate 600
     timeout 100 !
   !
  !
 !
!
IPv4: 180.1.1.1/16
!
interface Loopback180
 description IPv4 Host for NAT44
 ipv4 address 180.1.1.1 255.255.0.0
!
interface Loopback181
 description IPv4 Host for NAT44
 ipv4 address 181.1.1.1 255.255.0.0
!
interface GigabitEthernet0/6/5/1.1
 ipv4 address 10.12.13.1 255.255.255.0
 encapsulation dot1q 1
!
router static
 address-family ipv4 unicast
  40.22.0.0/16 10.12.13.2
  41.22.0.0/16 10.12.13.2
  100.0.0.0/24 10.12.13.2 !
!
```

# NAT64 Configuration: Example

This example shows a NAT64 sample configuration:

```
! Defines the Location for CGN Translation
! ---------------------------------
hw-module service cgn location 0/1/cpu0
```

```
! Defines the Service Infra Interface
! -----------------------------
interface ServiceInfra 1

  ! Defines the IP address and netmask
  ipv4 address 3.1.1.2 255.255.255.252

! Defines IPv4 to IPv6 Direction Service Application Interface
! --------------------------------------------
interface ServiceApp 1

  ! Assigns the IPv4 Address and Netmask
  ipv4 address 211.1.1.1 255.255.255.0

  ! Indicates the CGN Instance and Service-type
  service cgn cgn1 service-type nat64 stateful

! Defines IPv6 to IPv4 direction Service Application interface
! --------------------------------------------
interface ServiceApp 2

  ! Assigns IPv6 address and netmask
  ipv6 address 5005::5555/96

  ! Indicates the CGN instance and Service-type
  service cgn cgn1 service-type nat64 stateful

! Define the CGN Instance
! --------------------
service cgn cgn1

  ! Defines the Location
  service-location preferred-active 0/1/CPU0

  ! Defines the Service-type and Name
  service-type nat64 stateful nat64_1

    ! Enables RTSP ALG
    alg RTSP

    ! Defines the Port Limit
    portlimit 65535

    ! Defines IPv6 Prefix
    ipv6-prefix 3301:db8:1::/96

    ! Defines the IPv4 Address Pool
    ipv4 address-pool 52.1.64.0/22

    ! Defines the Dynamic Port Range
    dynamic-port-range start 1

    ! Defines the IPv4 Address Family
    address-family ipv4
        interface ServiceApp1

    ! Defines IPv6 Address Family
    address-family ipv6
      interface ServiceApp2

    ! Defines the UDP/TCP/ICMP Protocol
    protocol udp
      timeout 65535
```

```
    protocol tcp
      session initial timeout 65535
      session active timeout 65535

    protocol icmp
      timeout 65535

    ! Defines the Netflow and Associated Server.
    external-logging netflow version 9
      server
        address 10.64.81.232 port 44444

! Defines the Static Route for IPv4 to IPv6 direction
address-family ipv4 unicast

  ! This route is the same as the IPv4 CPE Domain Prefix
  52.1.64.0/22 ServiceApp1

! Defines Static Route for IPv6 to IPv4 direction
address-family ipv6 unicast

  ! This route is same as the IPv6 External Domain Prefix
  3301:db8:1::/96 ServiceApp2
```

# Bulk Port Allocation and Syslog Configuration: Example

```
service cgn cgn2
 service-type nat44 natA
  inside-vrf broadband
   map address-pool 100.1.2.0/24
   external-logging syslog
    server
     address 20.1.1.2 port 514
    !
   !
   bulk-port-alloc size 64
  !
 !
```

# Predefined NAT Configuration: Example

This example shows how to configure the predefined NAT for NAT44:

```
service cgn cgn1
service-location preferred-active 0/2/CPU0
service-type nat44 nat1
 inside-vrf red
   map outside-vrf blue address-pool 100.0.0.0/24
   nat-mode
    predefined private-pool 103.1.106.0/24
    predefined private-pool 103.1.107.0/26
    predefined private-pool 103.1.107.128/26
    predefined private-pool 103.1.108.0/23
    predefined private-pool 103.1.112.0/22
    predefined private-pool 103.1.116.0/24
    predefined private-pool 103.1.117.64/26
    predefined private-pool 103.1.117.192/26
```

# PPTP ALG Configuration: Example

## NAT44 Instance

```
service cgn cgn1
 service-location preferred-active 0/1/CPU0
 service-type nat44 inst1
  alg pptpAlg
```

•

# DBL Configuration: Example

## NAT44 Instance

```
service cgn cgn1
 service-type nat44 nat1
  inside-vrf ivrf
   external-logging netflow version 9
    server
     address x.x.x.x port x
      session-logging
```

# Configuring TCP Sequence-Check: Example

This example shows how to configure sequence check for TCP sessions.

```
configure
service cgn cgn1
service-type nat44 nat1
inside-vrf vrf1
firewall protocol tcp
sequence-check
```

# Configuring Address and Port-Dependent Filtering: Example

This example shows how to configure address and port-dependent filtering.

```
configure
service cgn cgn1
service-type nat44 nat1
inside-vrf vrf1
filter-policy ignore-port
```

# NAT0 Mode Configuration: Example

This example shows how to configure the NAT0 mode:

```
service cgn cgn1
 service-location preferred-active 0/2/CPU0
  service-type nat44 nat1
    inside-vrf Inside_1
      map outside-vrf ovrf outsideServiceApp ServiceApp2 address-pool 0.0.0.0/0
```

```
      nat-mode no-nat
```

# Configuration of Multiple NetFlow Servers: Example

```
service cgn cgn1
 service-location preferred-active 0/1/CPU0
 service-type nat44 nat1
  inside-vrf ivrf
   map outside-vrf ovrf outsideServiceApp ServiceApp2 address-pool 100.1.1.0/24
    external-logging netflow version 9
    server
     address 111.1.1.1 port 6000
      path-mtu 1200
      refresh-rate 600
      timeout 1000
      session-logging
     !
     address 111.1.1.1 port 9000
      path-mtu 1100
      refresh-rate 500
      timeout 1000
      session-logging
     !
     address 122.1.1.1 port 9000
      path-mtu 1200
      refresh-rate 500
      timeout 1100
      session-logging
     !
    !
   !
```

# Configuration of Multiple Syslog Servers: Servers

```
service cgn cgn1
 service-location preferred-active 0/1/CPU0
 service-type nat44 nat1
  inside-vrf ivrf
   map outside-vrf ovrf outsideServiceApp ServiceApp2 address-pool 100.1.1.0/24
   external-logging syslog
    server
     address 211.1.1.1 port 6000
      path-mtu 1200
      session-logging
     !
     address 211.1.1.1 port 9000
      path-mtu 1200
      session-logging
     !
     address 212.1.1.1 port 6000
      path-mtu 1200
      session-logging
     !
    !
   !
```

# CGN Sequential Allocation Algorithm

In classic NAT, the process of mapping a private IP to a public IP or a private port to an outside port is random. Therefore, it becomes difficult to track the subscribers using an IP and a port at a given time. Predefined NAT avoids this random process by mapping a private IP address to a range of ports associated with the corresponding public IP address. This is done through an algorithm that helps the user to recognize a private IP address without having to refer to the massive CGN logs. The address and port translation is done in accordance with the algorithm.

CGN sequential algorithm is based on RFC 7422 support for CGNAT44 deployment with A9K-VSM-500.

With CGN sequential algorithm, you can perform the following:

- Configure **dynamic port start range**, **port block allocation size**, **inside IPv4 pool**, **outside IPv4 pool**.
- Specify inside and outside pools by a start address and end address, instead of IPv4 prefix.

## Limitations

- Public Address Pool and Private address Pool distribution across cores is now dependent on port limit and dynamic port start values
- Default value of Dynamic port start is 1024 and Port Limit default value is 2048
- Port Limit minimum value is 256 and Maximum Value is 16384 for Sequential NAT
- Only One Sequential Pool can be configured per inside VRF. You have to mention ServiceApp number with Outside VRF in Public Pool configuration
- Show Output Translation command in sequential NAT 44 displays entries as per cores instead of blocks



**Note** All the limitations of Predefined DET NAT are applicable to Sequential Predefined NAT.

### Limitations of Predefined DET NAT

- The Bulk Port Allocation configuration is not available in the predefined mode. If you try to configure, Bulk Port Allocation on an inside VRF that has the predefined mode enabled, the configuration is rejected during verification.
- The port-preservation option is not available in the predefined mode.
- The global port limit parameter is not available for the predefined mode. Even though you will be allowed to configure the global port limit, the inside VRF, which has predefined mode enabled, ignores that port limit and uses the port limit configured by the algorithm.
- If you turn the predefined mode on or off for an inside VRF during the active translations, all the translations on that VRF are deleted.
- Ensure that you configure NetFlow or syslog only if it is very much required.

- Any configuration change that results in changes in mapping deletes the existing translations. Therefore, ensure that you record such configuration changes. You might need this information to trace the port usage by a subscriber.
- Ensure uniform port allocation uniform for all subscribers.

# Configuring Sequential Predefined NAT

1. configure terminal
2. service cgn instance-name
3. service-location preferred-active 0/x/CPU0 (‘x’ location of VSM card)
4. service-type nat44 nat1
5. dynamic-port-range start 2048
6. inside-vrf vrf-name
7. nat-mode
8. predefined seq-private-pool start-address end-address
9. map outside-vrf vrf-name seq-address-pool start-address end-address
10. portlimit 2048
11. commit

### Sequential Predefined NAT Configuration: Example

```
service cgn cgn123
service-location preferred-active 0/3/CPU0
service-type nat44 nat1
dynamic-port-range start 2048
   inside-vrf red
   nat-mode
   predefined seq-private-pool 12.0.0.0 12.0.0.255
   !
   map outside-vrf blue outsideServiceApp ServiceApp2 seq-address-pool 100.0.0.0 100.0.0.255

   portlimit 2048
  !
```

### Verification

```
RP/0/RSP0/CPU0:router#sh run service cgn cgn123
service cgn cgn123
 service-location preferred-active 0/3/CPU0
 service-type nat44 nat1
  dynamic-port-range start 2048
  inside-vrf red
   nat-mode
    predefined seq-private-pool 12.0.0.0 12.0.0.255
   !
   map outside-vrf blue seq-address-pool 100.0.0.0 100.0.0.255
   portlimit 2048
  !
```

```
 !

RP/0/RSP0/CPU0:router#sh cgn nat44 nat1 statistics
Statistics summary of NAT44 instance: 'nat1'
Number of active translations: 10496
Number of sessions: 0
Translations create rate: 0
Translations delete rate: 0
Inside to outside forward rate: 4078
Outside to inside forward rate: 4396
Inside to outside drops port limit exceeded: 0
Inside to outside drops system limit reached: 0
Inside to outside drops resource depletion: 0
No translation entry drops: 0
PPTP active tunnels: 0
PPTP active channels: 0
PPTP ctrl message drops: 0
Number of subscribers: 255
Drops due to session db limit exceeded: 0
Drops due to source ip not configured: 0

Pool address totally free: 0
Pool address used: 256
Pool address usage:
------------------------------------------------
  External Address       Ports Used
------------------------------------------------
  100.0.0.0              41
  100.0.0.0              41
  100.0.0.1              41
  100.0.0.2              41
  100.0.0.3              41
  100.0.0.3              41
  100.0.0.4              41
  100.0.0.5              41
……..
RP/0/RSP0/CPU0:router#show cgn nat44 nat1 mapping inside-address inside-vrf red start-addr
 12.0.0.0 end-addr 12.0.0.255
Mapping details for address pool inside a Vrf
-------------------------------------------------------------------------------------------------
NAT44 instance    : nat1
VRF               : red
-------------------------------------------------------------------------------------------------
Inside Ip          Outside IP       Type                Port Range         Ports Used
Address             Address
-------------------------------------------------------------------------------------------------
12.0.0.0         100.0.0.0            Predefined       2048-4095              41
12.0.0.1         100.0.0.0            Predefined       4096-6143              41
12.0.0.2         100.0.0.0            Predefined       6144-8191              41
12.0.0.3         100.0.0.0            Predefined       8192-10239             41
12.0.0.4         100.0.0.0            Predefined      10240-12287             41
12.0.0.5         100.0.0.0            Predefined      12288-14335             41
12.0.0.6         100.0.0.0            Predefined      14336-16383             41
12.0.0.7         100.0.0.0            Predefined      16384-18431             41
12.0.0.8         100.0.0.0            Predefined      18432-20479             41
12.0.0.9         100.0.0.0            Predefined      20480-22527             41
12.0.0.10        100.0.0.0            Predefined      22528-24575             41
12.0.0.11        100.0.0.0            Predefined      24576-26623             41
12.0.0.12        100.0.0.0            Predefined      26624-28671             41
12.0.0.13        100.0.0.0            Predefined      28672-30719             41
12.0.0.14        100.0.0.0            Predefined      30720-32767             41
……..
```

When **o2i-vrf-override** keyword is used for SEQ-NAT44, **ipv4 forwarding-enable** has to be configured on 12 TenGige (VNIC) interfaces on the VSM location that will receive reverse translated traffic. Refer to the config below:

```
interface TenGigE0/1/1/0
description virtual-service interface
mtu 9126
ipv4 forwarding-enable
!
```



**Note** The configuration is same for all the other 12 VNIC interfaces (interface TenGigE0/1/1/1, interface TenGigE0/1/1/2, and so on, to interface TenGigE0/1/1/12).