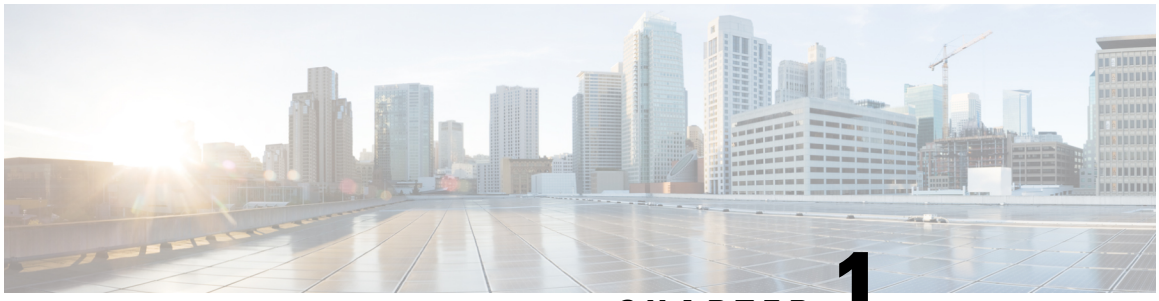
IP Addressing: NAT Configuration Guide, Cisco IOS XE Release 3S

Americas Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
http://www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0883

CONTENTS



CHAPTER 1

Configuring NAT for IP Address Conservation

This module describes how to configure Network Address Translation (NAT) for IP address conservation and how to configure inside and outside source addresses. This module also provides information about the benefits of configuring NAT for IP address conservation.

NAT enables private IP internetworks that use nonregistered IP addresses to connect to the Internet. NAT operates on a device, usually connecting two networks. Before packets are forwarded onto another network, NAT translates the private (not globally unique) addresses in the internal network into legal addresses. NAT can be configured to advertise to the outside world only one address for the entire network. This ability provides more security by effectively hiding the entire internal network behind that one address.

NAT is also used at the enterprise edge to allow internal users access to the Internet. It allows Internet access to internal devices such as mail servers.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Configuring NAT for IP Address Conservation

Access Lists

All access lists that are required for use with the configuration tasks that are described in this module must be configured before initiating a configuration task. For information about how to configure an access list, see the *IP Access List EntrySequence Numbering* document.

Note If you specify an access list with a NAT command, NAT will not support the **permit ip any any** command. This command is commonly used in an access list.

NAT Requirements

Before configuring NAT in your network, ensure that you know the interfaces on which NAT is configured and for what purposes. The following requirements help you decide how to configure and use NAT:

- Define the NAT inside and outside interfaces if:
  - Users exist off multiple interfaces.
  - Multiple interfaces connect to the Internet.
- Define what you need NAT to accomplish:
  - Allow internal users to access the Internet.
  - Allow the Internet to access internal devices such as a mail server.
  - Allow overlapping networks to communicate.
  - Allow networks with different address schemes to communicate.
  - Allow networks with different address schemes to communicate.
  - Redirect TCP traffic to another TCP port or address.
  - Use NAT during a network transition.

From Cisco IOS XE Denali 16.3 release, NAT support is introduced on Bridge Domain Interface (BDI) for enabling NAT configuration on the BDI interface.

Restrictions for Configuring NAT for IP Address Conservation

- When you configure Network Address Translation (NAT) on an interface, that interface becomes optimized for NAT packet flow. Any nontranslated packet that flows through the NAT interface goes through a series of checks to determine whether the packet must be translated or not. These checks result in increased latency for nontranslated packet flows and thus negatively impact the packet processing latency of all packet flows through the NAT interface. We highly recommend that a NAT interface must be used only

for NAT-only traffic. Any non-NAT packets must be separated and these packets must go through an interface that does not have NAT configured on it. You can use Policy-Based Routing (PBR) for separating non-NAT traffic.

- NAT Virtual Interfaces (NVIs) are not supported in the Cisco IOS XE software.
- In Cisco IOS XE software, NAT outside interfaces show up in the translations tables, by default. This view of NAT outside interfaces causes the connection that originates from the outside interface of the device to fail. To restore connectivity, you must explicitly deny the outside Interface within the NAT ACL using the **deny** command. After using the **deny** command, no translation is observed for the outside interface.
- NAT is not practical if large numbers of hosts in the stub domain communicate outside of the domain.
- Some applications use embedded IP addresses in such a way that translation by a NAT device is impractical. These applications may not work transparently or at all through a NAT device.
- In a NAT configuration, addresses configured for any inside mapping must not be configured for any outside mapping.
- Do not configure the interface IP address as part of the IP address NAT pool.
- By default, support for the Session Initiation Protocol (SIP) is enabled on port 5060. Therefore, NAT-enabled devices interpret all packets on this port as SIP call messages. If other applications in the system use port 5060 to send packets, the NAT service may corrupt the packet. This packet corruption is due to its attempt to interpret the packet as a SIP call message.
- NAT hides the identity of hosts, which may be an advantage or a disadvantage depending on the needed result.
- Devices that are configured with NAT must not advertise the local networks to outside the network. However, routing information that NAT receives from the outside can be advertised in the stub domain as usual.
- NAT outside interface is not supported on a VRF. However, NAT outside interface is supported in iWAN and is part of the Cisco Validated Design.
- For VRF-aware NAT, remove the NAT configuration before you remove the VRF configuration.
- If you specify an access list to use with a NAT command, NAT does not support the **permit ip any any** command. This NAT command is commonly used in the access list.
- Cisco ASR 1000 Series Aggregation Services Routers do not support an access list with a port range.
- NAT configuration is not supported on the access side of the Intelligent Services Gateway (ISG).
- Using any IP address that is configured of a device as an address pool or in a NAT static rule is not supported. NAT can share the physical interface address (not any other IP address) of a device only by using the NAT interface overload configuration. A device uses the ports of its physical interface and NAT must receive communication about the ports that it can safely use for translation. This communication happens only when the NAT interface overload is configured.
- The output of the **show ip nat statistics** command displays information about all IP address pools and NAT mappings that you have configured. If your NAT configuration has a high number of IP address pools and NAT mappings (for example, 1000 to 4000), the update rate of the pool and mapping statistics in **show ip nat statistics** is slow.

- Static and dynamic NAT with generic routing encapsulation (generic GRE) and dynamic NAT with Layer 2 do not work when used along with hardware-based Cisco AppNav appliances (for example, Wide Area Application Services [WAAS]). In the context of WAAS, generic GRE is an out-of-path deployment mechanism. It helps to return packets from the WAAS Wide-Area Application Engine (WAE) through the GRE tunnel to the same device from which they were originally redirected after completing optimization.
- Port Address Translation (also called NAT overload) only supports protocols whose port numbers are known; these protocols are Internet Control Message Protocol (ICMP), TCP, and UDP. Other protocols do not work with PAT because they consume the entire address in an address pool. Configure your access control list to only permit ICMP, TCP, and UDP protocols, so that all other protocol traffic is prevented from entering the network.
- NAT, Zone-Based Policy Firewall, and Web Cache Communication Protocol (WCCP) cannot coexist in a network.
- Non-Pattable traffic, is traffic for a protocol where there are no ports. PAT/Overload can only be done on protocols where the ports are known, that is, UDP, TCP, and ICMP.

  When ASR is configured for NAT overload (PAT) and Non-Pattable traffic hits the router, Non-Pattable BIND entry gets created for this traffic. Following is a bind entry in the NAT table:

  ```
  ---  213.252.7.132         172.16.254.242        ---
  ```

  This bind entry consumes an entire address from the pool. In this example, 213.252.7.132 is an address from an overloaded pool.

  That means an inside local IP Address gets bound to the outside global IP which is similar to static NAT. Because of this binding action, new inside local IP Addresses cannot use this global IP Address until the current entry gets timed out. All the translation that is created off this BIND is 1-to-1 translations instead of overload.

  To avoid consumption of an entire address from the pool, make sure that there are not any entries for the Non-Pattable traffic across the router.

Information About Configuring NAT for IP Address Conservation

Benefits of Configuring NAT for IP Address Conservation

NAT allows organizations to resolve the problem of IP address depletion when they have existing networks and must access the Internet. Sites that do not yet possess Network Information Center (NIC)-registered IP addresses must acquire them. If more than 254 clients are present or planned, the scarcity of Class B addresses becomes a serious issue. Cisco IOS XE NAT addresses these issues by mapping thousands of hidden internal addresses to a range of easy-to-get Class C addresses.

Sites that already have registered IP addresses for clients on an internal network may want to hide those addresses from the Internet. This action disable hacker to directly attack the clients. With clients addresses hidden, an extent of security is established. Cisco IOS XE NAT gives LAN administrators complete freedom to expand Class A addressing. The Class A addressing expansion is drawn from the reserve pool of the Internet Assigned Numbers Authority (RFC 1597). This expansion occurs within the organization without concern for addressing changes at the LAN/Internet interface.

The Cisco IOS XE software can selectively or dynamically perform NAT. This flexibility allows the network administrator to use a mix of RFC 1597 and RFC 1918 addresses or registered addresses. NAT is designed for use on various devices for IP address simplification and conservation. In addition, Cisco IOS XE NAT allows the selection of internal hosts that are available for NAT.

A significant advantage of NAT is that it can be configured without requiring changes to hosts or devices other than those few devices on which NAT is configured.

In Cisco IOS XE Denali 16.3 release, Multi-Tenant support for NAT feature was introduced. With Multi-Tenant support, the configuration changes of a Virtual Routing and Forwarding (VRF) instance does not interrupt the traffic flow of other VRFs in the network.

NAT is a feature that allows the IP network of an organization to appear from the outside to use a different IP address space than what it actually uses. Thus, NAT allows an organization with nonglobally routable addresses to connect to the Internet by translating those addresses into a globally routable address space. NAT also allows a graceful renumbering strategy for organizations that are changing service providers or voluntarily renumbering into classless interdomain routing (CIDR) blocks. NAT is described in RFC 1631.

How NAT Works

A device that is configured with NAT has at least one interface to the inside network and one to the outside network. In a typical environment, NAT is configured at the exit device between a stub domain and the backbone. When a packet exits the domain, NAT translates the locally significant source address into a globally unique address. When a packet enters the domain, NAT translates the globally unique destination address into a local address. If more than one exit point exists, each NAT must have the same translation table. If NAT cannot allocate an address because it has run out of addresses, it drops the packet. Then, NAT sends an Internet Control Message Protocol (ICMP) host unreachable packet to the destination.

Uses of NAT

NAT can be used for the following scenarios:

- To connect to the Internet, but not all your hosts have globally unique IP addresses. Network Address Translation (NAT) enables private IP internetworks that use nonregistered IP addresses to connect to the Internet. NAT is configured on a device at the border of a stub domain (mentioned as the *inside network*) and a public network such as the Internet (mentioned as the *outside network*). NAT translates internal local addresses to globally unique IP addresses before sending packets to the outside network. As a solution to the connectivity problem, NAT is practical only when relatively few hosts in a stub domain communicate outside of the domain at the same time. In that case, only a small subset of the IP addresses in the domain must be translated into globally unique IP addresses when outside communication is necessary. tAlso, these addresses can be reused when they are no longer in use.
- Change your internal addresses. Instead of changing the internal addresses, which can be a considerable amount of work, you can translate them by using NAT.
- For basic load-sharing of TCP traffic. You can map a single global IP address to many local IP addresses by using the TCP Load Distribution feature.

Types of NAT

NAT operates on a router—generally connecting only two networks. Before any packets are forwarded to another network, NAT translates the private (inside local) addresses within the internal network into public

(inside global) addresses. This functionality gives you the option to configure NAT so that it advertises only a single address for your entire network to the outside world. Doing this translation, NAT effectively hides the internal network from the world, giving you some additional security.

The types of NAT include:

- Static address translation (static NAT)—Allows one-to-one mapping between local and global addresses.
- Dynamic address translation (dynamic NAT)—Maps unregistered IP addresses to registered IP addresses from a pool of registered IP addresses.
- Overloading—Maps multiple unregistered IP addresses to a single registered IP address (many to one) by using different ports. This method is also known as Port Address Translation (PAT). Thousands of users can be connected to the Internet by using only one real global IP address through overloading.

NAT Inside and Outside Addresses

The term *inside* in a Network Address Translation (NAT) context refers to networks owned by an organization that must be translated. When NAT is configured, hosts within this network have addresses in one space (known as the *local* address space). These hosts appear to those outside the network as being in another space (known as the *global* address space).

Similarly, the term *outside* refers to those networks to which the stub network connects, and which are not under the control of an organization. Also, hosts in outside networks can be subject to translation, and can thus have local and global addresses. NAT uses the following definitions:

- Inside local address—An IP address that is assigned to a host on the inside network. The address that the Network Information Center (NIC) or service provider assigns is probably not a legitimate IP address.
- Inside global address—A legitimate IP address (assigned by the NIC or service provider) that represents one or more inside local IP addresses to the outside world.
- Outside local address—The IP address of an outside host as it appears to the inside network. Not necessarily a legitimate address, it is allocated from the address space that is routable on the inside.
- Outside global address—The IP address that is assigned to a host on the outside network by the owner of the host. The address is allocated from a globally routable address or network space.

NAT supports the following VRFs:

Table 1: VRF NAT Support

NAT Inside Interface	NAT Outside Interface	Condition
Global VRF (also referred to as a non-VRF interface)	Global VRF (also referred to as a non-VRF interface)	Normal
VRF X	Global VRF (also referred to as a non-VRF interface)	When NAT is not configured for Match-in-VRF support. For more details, see the *Match-in-VRF Support for NAT* chapter.

NAT Inside Interface	NAT Outside Interface	Condition
VRF X	VRF X	When both inside and outside interfaces are in the same VRF, and NAT is configured with Match-in-VRF support.

This section describes the following topics:

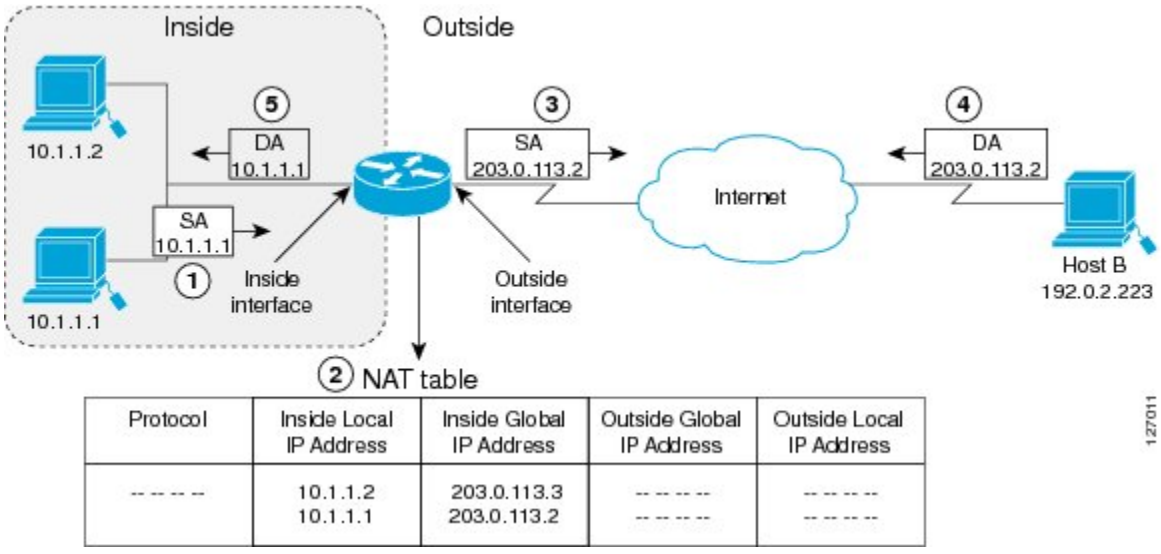
Inside Source Address Translation

You can translate IP addresses into globally unique IP addresses when communicating outside of your network. You can configure inside source address translation of static or dynamic NAT as follows:

- *Static translation* establishes a one-to-one mapping between the inside local address and an inside global address. Static translation is useful when a host on the inside must be accessible by a fixed address from the outside.
- *Dynamic translation* establishes a mapping between an inside local address and a pool of global addresses.

The following figure illustrates a device that is translating a source address inside a network to a source address outside the network.

Figure 1: NAT Inside Source Translation



The following process describes the inside source address translation, as shown in the preceding figure:

1. The user at host 10.1.1.1 opens a connection to Host B in the outside network.
2. The first packet that the device receives from host 10.1.1.1 causes the device to check its Network Address Translation (NAT) table. Based on the NAT configuration, the following scenarios are possible:
   - If a static translation entry is configured, the device goes to Step 3.

- If no translation entry exists, the device determines that the source address (SA) 10.1.1.1 must be translated dynamically. The device selects a legal, global address from the dynamic address pool, and creates a translation entry in the NAT table. This kind of translation entry is called a *simple entry*.

3. The device replaces the inside local source address of host 10.1.1.1 with the global address of the translation entry and forwards the packet.
4. Host B receives the packet and responds to host 10.1.1.1 by using the inside global IP destination address (DA) 203.0.113.2.
5. When the device receives the packet with the inside global IP address, it performs a NAT table lookup by using the inside global address as a key. It then translates the address to the inside local address of host 10.1.1.1 and forwards the packet to host 10.1.1.1.

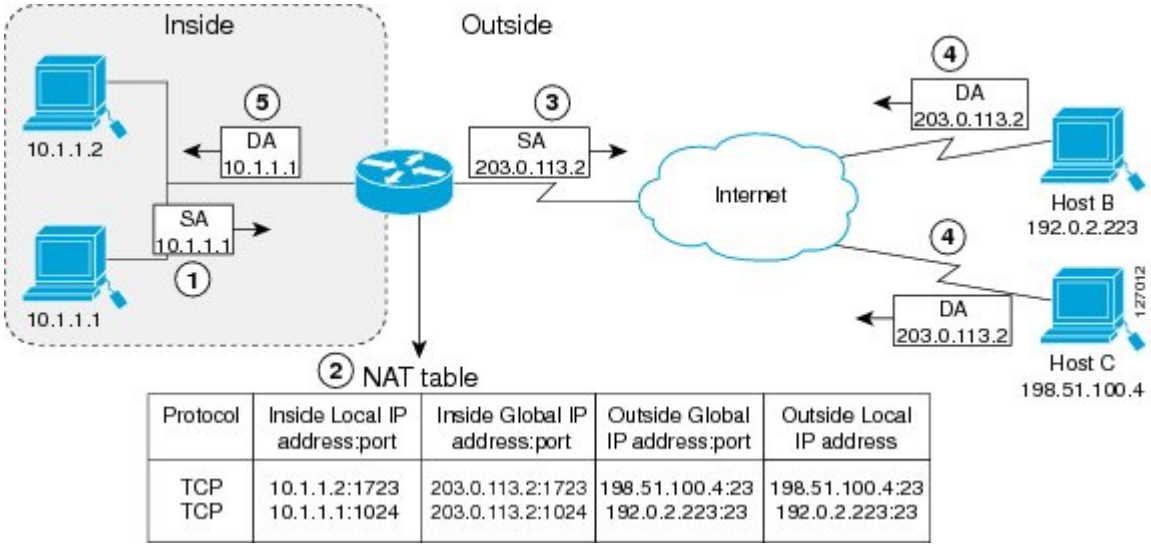
Host 10.1.1.1 receives the packet and continues the conversation. The device performs Steps 2 to 5 for each packet that it receives.

Overloading of Inside Global Addresses

You can conserve addresses in the inside global address pool by allowing a device to use one global address for many local addresses. This type of Network Address Translation (NAT) configuration is called overloading. When overloading is configured, the device maintains enough information from higher-level protocols (for example, TCP or UDP port numbers). This action translates the global address back to the correct local address. When multiple local addresses map to one global address, the TCP or UDP port numbers of each inside host distinguish between local addresses.

The following figure illustrates a NAT operation when an inside global address represents multiple inside local addresses. The TCP port numbers act as differentiators.

Figure 2: NAT Overloading Inside Global Addresses



Protocol	Inside Local IP address:port	Inside Global IP address:port	Outside Global IP address:port	Outside Local IP address
TCP	10.1.1.2:1723	203.0.113.2:1723	198.51.100.4:23	198.51.100.4:23
TCP	10.1.1.1:1024	203.0.113.2:1024	192.0.2.223:23	192.0.2.223:23

The device performs the following process in the overloading of inside global addresses, as shown in the preceding figure. Both Host B and Host C believe that they are communicating with a single host at address 203.0.113.2. Whereas, they are actually communicating with different hosts; the port number is the differentiator. In fact, many inside hosts can share the inside global IP address by using many port numbers.

1. The user at host 10.1.1.1 opens a connection to Host B.
2. The first packet that the device receives from host 10.1.1.1 causes the device to check its NAT table. Based on your NAT configuration the following scenarios are possible:
   - If no translation entry exists, the device determines that IP address 10.1.1.1 must be translated, and translates inside local address 10.1.1.1 to a legal global address.
   - If overloading is enabled and another translation is active, the device reuses the global address from that translation and saves enough information. This saved information can be used to translate the global address back, as an entry in the NAT table. This type of translation entry is called an *extended entry*.
3. The device replaces inside local source address 10.1.1.1 with the selected global address and forwards the packet.
4. Host B receives the packet and responds to host 10.1.1.1 by using the inside global IP address 203.0.113.2.
5. When the device receives the packet with the inside global IP address, it performs a NAT table lookup by using a protocol, the inside global address and port, and the outside address and port as keys. It translates the address to the inside local address 10.1.1.1 and forwards the packet to host 10.1.1.1.

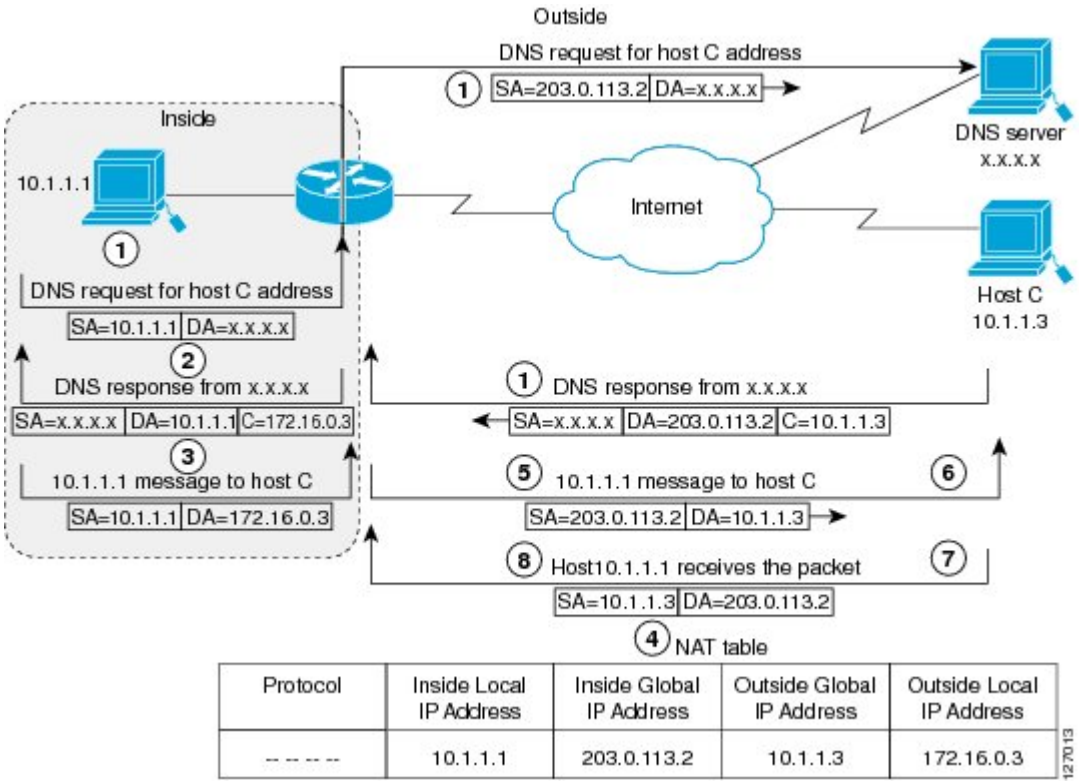
Host 10.1.1.1 receives the packet and continues the conversation. The device performs Steps 2 to 5 for each packet it receives.

Address Translation of Overlapping Networks

Use Network Address Translation (NAT) to translate IP addresses if the IP addresses that you use are not legal or officially assigned. Overlapping networks result when you assign an IP address to a device on your network. This device is already legally owned and assigned to a different device on the Internet or outside the network.

The following figure shows how NAT translates overlapping networks.

Figure 3: NAT Translating Overlapping Addresses



The following steps describe how a device translates overlapping addresses:

1. Host 10.1.1.1 opens a connection to Host C using a name, requesting a name-to-address lookup from a Domain Name System (DNS) server.
2. The device intercepts the DNS reply, and translates the returned address if there is an overlap. That is, the resulting legal address resides illegally in the inside network. To translate the return address, the device creates a simple translation entry. This entry maps the overlapping address, 10.1.1.3 to an address from a separately configured, outside the local address pool.

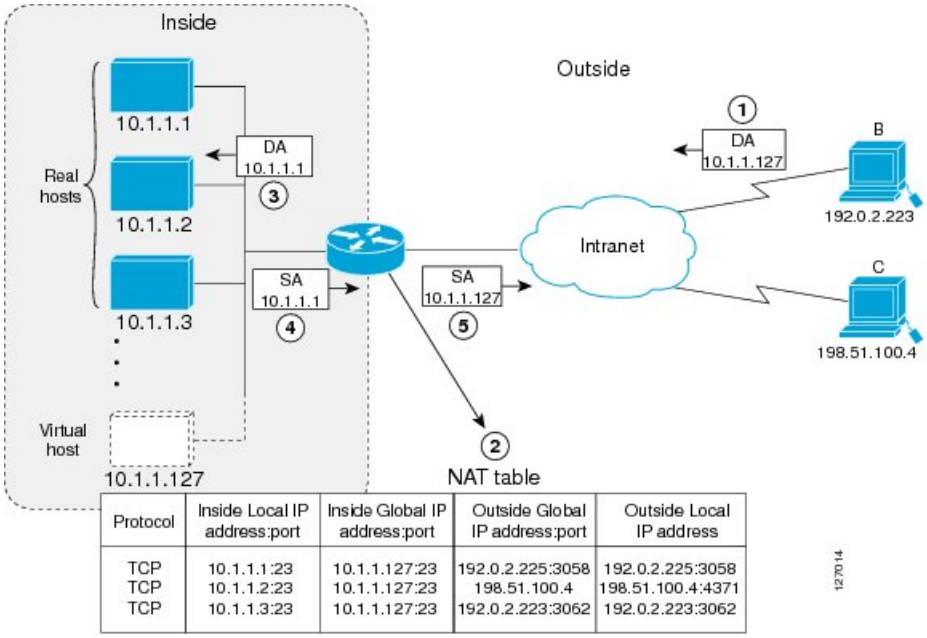
The device examines every DNS reply to ensure that the IP address is not in a stub network. If it is, the device translates the address as described in the following steps:

1. Host 10.1.1.1 opens a connection to 172.16.0.3.
2. The device sets up the translation mapping of the inside local and global addresses to each other. It also sets up the translation mapping of the outside global and local addresses to each other.
3. The device replaces the SA with the inside global address and replaces the DA with the outside global address.
4. Host C receives the packet and continues the conversation.
5. The device does a lookup, replaces the DA with the inside local address, and replaces the SA with the outside local address.
6. Host 10.1.1.1 receives the packet and the conversation continues using this translation process.

TCP Load Distribution for NAT

Your organization may have multiple hosts that must communicate with a heavily used host. By using Network Address Translation (NAT), you can establish a virtual host on the inside network that coordinates load sharing among real hosts. Destination addresses that match an access list are replaced with addresses from a rotary pool. Allocation is done on a round-robin basis and only when a new connection is opened from the outside to inside the network. Non-TCP traffic is passed untranslated (unless other translations are configured). The following figure illustrates how TCP load distribution works.

Figure 4: NAT TCP Load Distribution



A device performs the following process when translating rotary addresses:

1. Host B (192.0.2.223) opens a connection to a virtual host at 10.1.1.127.
2. The device receives the connection request and creates a new translation, allocating the next real host (10.1.1.1) for the inside local IP address.
3. The device replaces the destination address with the selected real host address and forwards the packet.
4. Host 10.1.1.1 receives the packet and responds.
5. The device receives the packet and performs a NAT table lookup by using the inside local address and port number. It also does a NAT table lookup by using the outside address and port number as keys. The device then translates the source address to the address of the virtual host and forwards the packet.
6. The device will allocate IP address 10.1.1.2 as the inside local address for the next connection request.

Static IP Address Support

A public wireless LAN provides users of mobile computing devices with wireless connections to a public network, such as the Internet.

To support users who are configured with a static IP address, the NAT Static IP Address Support feature extends the capabilities of public wireless LAN providers. By configuring a device to support users with a static IP address, public wireless LAN providers extend their services to a greater number of users.

Users with static IP addresses can use services of the public wireless LAN provider without changing their IP address. NAT entries are created for static IP clients and a routable address is provided.

RADIUS

RADIUS is a distributed client/server system that secures networks against unauthorized access. Communication between a network access server (NAS) and a RADIUS server is based on UDP. Generally, the RADIUS protocol is considered a connectionless service. RADIUS-enabled devices handle issues that are related to a server availability, retransmission, and timeouts rather than the transmission protocol.

The RADIUS client is typically a NAS, and the RADIUS server is usually a daemon process running on a UNIX or Windows NT machine. The client passes user information to designated RADIUS servers and acts on the response that is returned. To deliver service to the user, RADIUS servers receive a user connection request, authenticate the user, and then return the configuration information necessary for the client. A RADIUS server can act as a proxy client to other RADIUS servers or other kinds of authentication servers.

Denial-of-Service Attacks

A denial-of-service (DoS) attack typically involves misuse of standard protocols or connection processes. The intent of DoS attack is to overload and disable a target, such as a device or web server. DoS attacks can come from a malicious user or from a computer that is infected with a virus or worm. Distributed DoS attack is an attack that comes from many different sources at once. This attack can be when a virus or worm has infected many computers. Such distributed DoS attacks can spread rapidly and involve thousands of systems.

Viruses and Worms That Target NAT

Viruses and worms are malicious programs that are designed to attack computers and networking equipment. Although viruses are typically embedded in discrete applications and run only when executed, worms self-propagate and can quickly spread by their own. Although a specific virus or worm may not expressly target NAT, it may use NAT resources to propagate itself. The Rate Limiting NAT Translation feature can be used to limit the impact of viruses and worms. These viruses and worms originate from specific hosts, access control lists, and VPN routing and forwarding (VRF) instances.

How to Configure NAT for IP Address Conservation

The tasks that are described in this section configure NAT for IP address conservation. Ensure that you configure at least one of the tasks that are described in this section. Based on your configuration, you may need to configure more than one task.

Configuring Inside Source Addresses

Inside source addresses, can be configured for static or dynamic translations. Based on your requirements, you can configure either static or dynamic translations.

Configuring Static Translation of Inside Source Addresses

Configure static translation of the inside source addresses to allow one-to-one mapping between an inside local address and an inside global address. Static translation is useful when a host on the inside must be accessible by a fixed address from the outside.

Note Configure different IP addresses for an interface on which NAT is configured and for inside addresses that are configured by using the **ip nat inside source static** command.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip nat inside source static** *local-ip global-ip*
4. **interface** *type number*
5. **ip address** *ip-address mask* [**secondary**]
6. **ip nat inside**
7. **exit**
8. **interface** *type number*
9. **ip address** *ip-address mask* [**secondary**]
10. **ip nat outside**
11. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip nat inside source static** *local-ip global-ip* **Example:** `Device(config)# ip nat inside source static 10.10.10.1 172.16.131.1`	Establishes static translation between an inside local address and an inside global address.
Step 4	**interface** *type number* **Example:** `Device(config)# interface ethernet 1`	Specifies an interface and enters the interface configuration mode.
Step 5	**ip address** *ip-address mask* [**secondary**] **Example:**	Sets a primary IP address for an interface.

	Command or Action	Purpose
	`Device(config-if)# ip address 10.114.11.39 255.255.255.0`	
Step 6	**ip nat inside** **Example:** `Device(config-if)# ip nat inside`	Connects the interface to the inside network, which is subject to NAT.
Step 7	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 8	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 0/0/0`	Specifies a different interface and enters the interface configuration mode.
Step 9	**ip address** *ip-address mask* [**secondary**] **Example:** `Device(config-if)# ip address 172.31.232.182 255.255.255.240`	Sets a primary IP address for an interface.
Step 10	**ip nat outside** **Example:** `Device(config-if)# ip nat outside`	Connects the interface to the outside network.
Step 11	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode. **Note** Conditional translation is not supported with **ip nat outside source route-map** configuration.

Configuring Dynamic Translation of Inside Source Addresses

Dynamic translation establishes a mapping between an inside local address and a pool of global addresses. Dynamic translation is useful when multiple users on a private network must access the Internet. The dynamically configured pool IP address may be used as needed. It is released for use by other users when access to the Internet is no longer required.

Note When inside global or outside local addresses belong to a directly connected subnet on a NAT device, the device adds IP aliases for them. This action enables it to answer Address Resolution Protocol (ARP) requests. However, a situation can arise where the device answers packets that are not destined for it, possibly causing a security issue. This security issue can happen when an incoming Internet Control Message Protocol (ICMP) packet or a UDP packet that is destined for one of the aliased addresses does not have a corresponding NAT translation in the NAT table. Also, the device itself runs a corresponding service, for example, Network Time Protocol (NTP). Such a situation can cause minor security risks.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip nat pool** *name start-ip end-ip* {**netmask** *netmask* | **prefix-length** *prefix-length*}
4. **access-list** *access-list-number* **permit** *source* [*source-wildcard*]
5. **ip nat inside source list** *access-list-number* **pool** *name*
6. **interface** *type number*
7. **ip address** *ip-address mask*
8. **ip nat inside**
9. **exit**
10. **interface** *type number*
11. **ip address** *ip-address mask*
12. **ip nat outside**
13. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip nat pool** *name start-ip end-ip* {**netmask** *netmask* | **prefix-length** *prefix-length*} **Example:** `Device(config)# ip nat pool net-208 172.16.233.208 172.16.233.223 prefix-length 28`	Defines a pool of global addresses to be allocated as needed.
Step 4	**access-list** *access-list-number* **permit** *source* [*source-wildcard*] **Example:** `Device(config)# access-list 1 permit 192.168.34.0 0.0.0.255`	Defines a standard access list permitting those addresses that are to be translated.
Step 5	**ip nat inside source list** *access-list-number* **pool** *name* **Example:** `Device(config)# ip nat inside source list 1 pool net-208`	Establishes dynamic source translation, specifying the access list defined in Step 4.
Step 6	**interface** *type number* **Example:** `Device(config)# interface ethernet 1`	Specifies an interface and enters an interface configuration mode.

	Command or Action	Purpose
Step 7	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 10.114.11.39 255.255.255.0`	Sets a primary IP address for the interface.
Step 8	**ip nat inside** **Example:** `Device(config-if)# ip nat inside`	Connects the interface to the inside network, which is subject to NAT.
Step 9	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 10	**interface** *type number* **Example:** `Device(config)# interface ethernet 0`	Specifies an interface and enters an interface configuration mode.
Step 11	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 172.16.232.182 255.255.255.240`	Sets a primary IP address for the interface.
Step 12	**ip nat outside** **Example:** `Device(config-if)# ip nat outside`	Connects the interface to the outside network.
Step 13	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Using NAT to Allow Internal Users Access to the Internet

Perform this task to allow your internal users access to the Internet and conserve addresses in the inside global address pool using overloading of global addresses.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip nat pool** *name start-ip end-ip* {**netmask** *netmask* | **prefix-length** *prefix-length*}
4. **access-list** *access-list-number* **permit** *source* [*source-wildcard*]
5. **ip nat inside source list** *access-list-number* **pool** *name* **overload**
6. **interface** *type number*
7. **ip address** *ip-address mask*
8. **ip nat inside**

9. **exit**
10. **interface** *type number*
11. **ip address** *ip-address mask*
12. **ip nat outside**
13. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip nat pool** *name start-ip end-ip* {**netmask** *netmask* | **prefix-length** *prefix-length*} **Example:** `Device(config)# ip nat pool net-208 192.168.202.129 192.168.202.158 netmask 255.255.255.224`	Defines a pool of global addresses to be allocated as needed.
Step 4	**access-list** *access-list-number* **permit** *source* [*source-wildcard*] **Example:** `Device(config)# access-list 1 permit 192.168.201.30 0.0.0.255`	Defines a standard access list permitting those addresses that are to be translated. • The access list must permit only those addresses that are to be translated. (Remember that there is an implicit "deny all" at the end of each access list.) Use of an access list that is too permissive can lead to unpredictable results.
Step 5	**ip nat inside source list** *access-list-number* **pool** *name* **overload** **Example:** `Device(config)# ip nat inside source list 1 pool net-208 overload`	Establishes dynamic source translation with overloading, specifying the access list defined in Step 4.
Step 6	**interface** *type number* **Example:** `Device(config)# interface ethernet 1`	Specifies an interface and enters the interface configuration mode.
Step 7	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 192.168.201.1 255.255.255.240`	Sets a primary IP address for the interface.

	Command or Action	Purpose
Step 8	**ip nat inside** **Example:** `Device(config-if)# ip nat inside`	Connects the interface to the inside network, which is subject to NAT.
Step 9	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 10	**interface** *type number* **Example:** `Device(config)# interface ethernet 0`	Specifies an interface and enters the interface configuration mode.
Step 11	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 192.168.201.29 255.255.255.240`	Sets a primary IP address for the interface.
Step 12	**ip nat outside** **Example:** `Device(config-if)# ip nat outside`	Connects the interface to the outside network.
Step 13	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring Address Translation Timeouts

You can configure address translation timeouts that is based on your NAT configuration.

By default, dynamic address translations time out after a period of nonuse. You can change the default values on timeouts, if necessary. When overloading is not configured, simple translation entries time out after 24 hours. Use the **ip nat translation timeout** command to change the timeout value for dynamic address translations.

You can use the **ip nat translation max-entries** command to change the default global NAT translation limit.

Note On Catalyst 6500 Series Switches, when the NAT translation is done in the hardware, timers are reset every 100 seconds or once the set timeout value is reached.

Changing the Translation Timeout

By default, dynamic address translations time out after some period of nonuse. You can change the default values on timeouts, if necessary. When overloading is not configured, simple translation entries time out after 24 hours. Configure the **ip nat translation timeout** *seconds* command to change the timeout value for dynamic address translations that do not use overloading.

Changing the Timeouts When Overloading Is Configured

If you have configured overloading, you can control the translation entry timeout, because each translation entry contains more context about the traffic using it.

Based on your configuration, you can change the timeouts that are described in this section. If you must quickly free your global IP address for a dynamic configuration, configure a shorter timeout than the default timeout. You can do it by using the **ip nat translation timeout** command. However, the configured timeout is longer than the other timeouts configured using commands specified in the following task. If a finish (FIN) packet does not close a TCP session properly from both sides or during a reset, change the default TCP timeout. You can do it by using the **ip nat translation tcp-timeout** command.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip nat translation** *seconds*
4. **ip nat translation udp-timeout** *seconds*
5. **ip nat translation dns-timeout** *seconds*
6. **ip nat translation tcp-timeout** *seconds*
7. **ip nat translation finrst-timeout** *seconds*
8. **ip nat translation icmp-timeout** *seconds*
9. **ip nat translation syn-timeout** *seconds*
10. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip nat translation** *seconds* **Example:** `Device(config)# ip nat translation 300`	(Optional) Changes the amount of time after which NAT translations time out. • The default timeout is 24 hours, and it applies to the aging time for half-entries.
Step 4	**ip nat translation udp-timeout** *seconds* **Example:** `Device(config)# ip nat translation udp-timeout 300`	(Optional) Changes the UDP timeout value.
Step 5	**ip nat translation dns-timeout** *seconds* **Example:**	(Optional) Changes the Domain Name System (DNS) timeout value.

	Command or Action	Purpose
	`Device(config)# ip nat translation dns-timeout 45`	
Step 6	**ip nat translation tcp-timeout** *seconds* **Example:** `Device(config)# ip nat translation tcp-timeout 2500`	(Optional) Changes the TCP timeout value. • The default is 24 hours.
Step 7	**ip nat translation finrst-timeout** *seconds* **Example:** `Device(config)# ip nat translation finrst-timeout 45`	(Optional) Changes the finish and reset timeout value. • **finrst-timeout**—The aging time after a TCP session receives both finish-in (FIN-IN) and finish-out (FIN-OUT) requests or after the reset of a TCP session.
Step 8	**ip nat translation icmp-timeout** *seconds* **Example:** `Device(config)# ip nat translation icmp-timeout 45`	(Optional) Changes the ICMP timeout value.
Step 9	**ip nat translation syn-timeout** *seconds* **Example:** `Device(config)# ip nat translation syn-timeout 45`	(Optional) Changes the synchronous (SYN) timeout value. • The synchronous timeout or the aging time is used only when a SYN request is received on a TCP session. When a synchronous acknowledgment (SYNACK) request is received, the timeout changes to TCP timeout.
Step 10	**end** **Example:** `Device(config)# end`	(Optional) Exits global configuration mode and returns to privileged EXEC mode.

Allowing Overlapping Networks to Communicate Using NAT

Tasks in this section are grouped because they perform the same action. However, the tasks are executed differently depending on the type of translation that is implemented—static or dynamic. Perform the task that applies to the translation type that you have implemented.

This section contains the following tasks:

- Configuring Static Translation of Overlapping Networks
- Configuring Dynamic Translation of Overlapping Networks
- What to Do Next

Configuring Static Translation of Overlapping Networks

Configure static translation of overlapping networks that are based on the following requirements:

- If your IP addresses in the stub network are legitimate IP addresses belonging to another network.
- If you want to communicate with those hosts or routers by using static translation.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip nat inside source static** *local-ip global-ip*
4. **interface** *type number*
5. **ip address** *ip-address mask*
6. **ip nat inside**
7. **exit**
8. **interface** *type number*
9. **ip address** *ip-address mask*
10. **ip nat outside**
11. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip nat inside source static** *local-ip global-ip* **Example:** `Device(config)# ip nat inside source static 192.168.121.33 10.2.2.1`	Establishes static translation between an inside local address and an inside global address.
Step 4	**interface** *type number* **Example:** `Device(config)# interface ethernet 1`	Specifies an interface and enters the interface configuration mode.
Step 5	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 10.114.11.39 255.255.255.0`	Sets a primary IP address for the interface.
Step 6	**ip nat inside** **Example:** `Device(config-if)# ip nat inside`	Marks the interface as connected to the inside.

	Command or Action	Purpose
Step 7	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 8	**interface** *type number* **Example:** `Device(config)# interface ethernet 0`	Specifies an interface and enters the interface configuration mode.
Step 9	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 172.16.232.182 255.255.255.240`	Sets a primary IP address for the interface.
Step 10	**ip nat outside** **Example:** `Device(config-if)# ip nat outside`	Marks the interface as connected to the outside.
Step 11	**end** **Example:** `Device(config-if)# end`	(Optional) Exits interface configuration mode and returns to privileged EXEC mode.

Configuring Dynamic Translation of Overlapping Networks

Configure dynamic translation of overlapping networks:

- If your IP addresses in the stub network are legitimate IP addresses belonging to another network.
- You want to communicate with those hosts or routers by using dynamic translation.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip nat pool** *name start-ip end-ip* {**netmask** *netmask* | **prefix-length** *prefix-length*}
4. **access-list** *access-list-number* **permit** *source* [*source-wildcard*]
5. **ip nat outside source list** *access-list-number* **pool** *name*
6. **interface** *type number*
7. **ip address** *ip-address mask*
8. **ip nat inside**
9. **exit**
10. **interface** *type number*
11. **ip address** *ip-address mask*
12. **ip nat outside**
13. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip nat pool** *name start-ip end-ip* {**netmask** *netmask* | **prefix-length** *prefix-length*} **Example:** `Device(config)# ip nat pool net-10 10.0.1.0 10.0.1.255 prefix-length 24`	Defines a pool of global addresses to be allocated as needed.
Step 4	**access-list** *access-list-number* **permit** *source* [*source-wildcard*] **Example:** `Device(config)# access-list 1 permit 10.114.11.0 0.0.0.255`	Defines a standard access list permitting those addresses that are to be translated. • The access list must permit only those addresses that are to be translated. (Remember that there is an implicit "deny all" at the end of each access list.) Use of an access list that is too permissive can lead to unpredictable results.
Step 5	**ip nat outside source list** *access-list-number* **pool** *name* **Example:** `Device(config)# ip nat outside source list 1 pool net-10`	Establishes dynamic outside source translation, specifying the access list defined in Step 4.
Step 6	**interface** *type number* **Example:** `Device(config)# interface ethernet 1`	Specifies an interface and enters the interface configuration mode.
Step 7	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 10.114.11.39 255.255.255.0`	Sets a primary IP address for the interface.
Step 8	**ip nat inside** **Example:** `Device(config-if)# ip nat inside`	Marks the interface as connected to the inside.
Step 9	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.

	Command or Action	Purpose
Step 10	**interface** *type number* **Example:** `Device(config)# interface ethernet 0`	Specifies an interface and enters the interface configuration mode.
Step 11	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 172.16.232.182 255.255.255.240`	Sets a primary IP address for the interface.
Step 12	**ip nat outside** **Example:** `Device(config-if)# ip nat outside`	Marks the interface as connected to the outside.
Step 13	**end** **Example:** `Device(config-if)# end`	(Optional) Exits interface configuration mode and returns to privileged EXEC mode.

What to Do Next

When you have completed the required configuration, go to the "Monitoring and Maintaining NAT" module.

Configuring Server TCP Load Balancing

Perform this task to configure a server TCP load balancing by way of destination address rotary translation. The commands that are specified in the task allow you to map one virtual host with many real hosts. Each new TCP session opened with the virtual host is translated into a session with a different real host.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip nat pool** *name start-ip end-ip* {**netmask** *netmask* | **prefix-length** *prefix-length*} **type rotary**
4. **access-list** *access-list-number* **permit** *source* [*source-wildcard*]
5. **ip nat inside destination-list** *access-list-number* **pool** *name*
6. **interface** *type number*
7. **ip address** *ip-address mask*
8. **ip nat inside**
9. **exit**
10. **interface** *type number*
11. **ip address** *ip-address mask*
12. **ip nat outside**
13. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip nat pool** *name start-ip end-ip* {**netmask** *netmask* | **prefix-length** *prefix-length*} **type rotary** **Example:** `Device(config)# ip nat pool real-hosts 192.168.201.2 192.168.201.5 prefix-length 28 type rotary`	Defines a pool of addresses containing the addresses of the real hosts.
Step 4	**access-list** *access-list-number* **permit** *source* [*source-wildcard*] **Example:** `Device(config)# access-list 1 permit 192.168.201.30 0.0.0.255`	Defines an access list permitting the address of the virtual host.
Step 5	**ip nat inside destination-list** *access-list-number* **pool** *name* **Example:** `Device(config)# ip nat inside destination-list 2 pool real-hosts`	Establishes dynamic inside destination translation, specifying the access list defined in the prior step.
Step 6	**interface** *type number* **Example:** `Device(config)# interface ethernet 0`	Specifies an interface and enters the interface configuration mode.
Step 7	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 192.168.201.1 255.255.255.240`	Sets a primary IP address for the interface.
Step 8	**ip nat inside** **Example:** `Device(config-if)# ip nat inside`	Marks the interface as connected to the inside.
Step 9	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.

	Command or Action	Purpose
Step 10	**interface** *type number* **Example:** `Device(config)# interface serial 0`	Specifies a different interface and enters the interface configuration mode.
Step 11	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 192.168.15.129 255.255.255.240`	Sets a primary IP address for the interface.
Step 12	**ip nat outside** **Example:** `Device(config-if)# ip nat outside`	Marks the interface as connected to the outside.
Step 13	**end** **Example:** `Device(config-if)# end`	(Optional) Exits interface configuration mode and returns to privileged EXEC mode.

Enabling Route Maps on Inside Interfaces

Before you begin

All route maps required for use with this task must be configured before you begin the configuration task.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip nat inside source** {**list** {*access-list-number* | *access-list-name*} **pool** *pool-name* [**overload**]| **static** *local-ip global-ip* [**route-map** *map-name*]}
4. **exit**
5. **show ip nat translations** [**verbose**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**ip nat inside source** {**list** {*access-list-number* | *access-list-name*} **pool** *pool-name* [**overload**]| **static** *local-ip global-ip* [**route-map** *map-name*]} **Example:** `Device(config)# ip nat inside source static 192.168.201.6 192.168.201.21 route-map isp2`	Enables route mapping with static NAT configured on the NAT inside interface.
Step 4	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.
Step 5	**show ip nat translations** [**verbose**] **Example:** `Device# show ip nat translations`	(Optional) Displays active NAT.

Enabling NAT Route Maps Outside-to-Inside Support

The NAT Route Maps Outside-to-Inside Support feature enables you to configure a Network Address Translation (NAT) route map configuration. It allows IP sessions to be initiated from the outside to the inside. Perform this task to enable the NAT Route Maps Outside-to-Inside Support feature.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip nat pool** *name start-ip end-ip* **netmask** *netmask*
4. **ip nat pool** *name start-ip end-ip* **netmask** *netmask*
5. **ip nat inside source route-map** *name* **pool** *name* [**reversible**]
6. **ip nat inside source route-map** *name* **pool** *name* [**reversible**]
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device(config)# configure terminal`	Enters global configuration mode.
Step 3	**ip nat pool** *name start-ip end-ip* **netmask** *netmask* **Example:**	Defines a pool of network addresses for NAT.

	Command or Action	Purpose
	`Device(config)# ip nat pool POOL-A 192.168.201.4 192.168.201.6 netmask 255.255.255.128`	
Step 4	**ip nat pool** *name start-ip end-ip* **netmask** *netmask* **Example:** `Device(config)# ip nat pool POOL-B 192.168.201.7 192.168.201.9 netmask 255.255.255.128`	Defines a pool of network addresses for NAT.
Step 5	**ip nat inside source route-map** *name* **pool** *name* [**reversible**] **Example:** `Device(config)# ip nat inside source route-map MAP-A pool POOL-A reversible`	Enables outside-to-inside initiated sessions to use route maps for destination-based NAT.
Step 6	**ip nat inside source route-map** *name* **pool** *name* [**reversible**] **Example:** `Device(config)# ip nat inside source route-map MAP-B pool POOL-B reversible`	Enables outside-to-inside initiated sessions to use route maps for destination-based NAT.
Step 7	**end** **Example:** `Device(config)# end`	(Optional) Exits global configuration mode and returns to privileged EXEC mode.

Configuring NAT of External IP Addresses Only

When you configure NAT of external IP addresses, NAT can be configured to ignore all embedded IP addresses for any application and traffic type. Traffic between a host and the traffic outside an enterprise's network flows through the internal network. A device that is configured for NAT translates the packet to an address that can be routed inside the internal network. If the intended destination is outside an enterprise's network, the packet gets translated back to an external address and is sent out.

Note When you configure the **ip nat outside source static** command to add static routes for outside local addresses, there is a delay in the translation of packets and packets are dropped. Packets are dropped because a shortcut is not created for the initial synchronization (SYN) packet when NAT is configured for static translation. To avoid dropped packets, configure either the **ip nat outside source static add-route** command or the **ip route** command.

Benefits of configuring NAT of external IP addresses only are:

- Allows an enterprise to use the Internet as its enterprise backbone network.
- Allows the use of network architecture that requires only the header translation.
- Gives the end client a usable IP address at the starting point. This address is the address that is used for IPsec connections and for traffic flows.
- Supports public and private network architecture with no specific route updates.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip nat inside source** {**list** {*access-list-number* | *access-list-name*} **pool** *pool-name* [**overload**] | **static network** *local-ip global-ip* [**no-payload**]}
4. **ip nat inside source** {**list** {*access-list-number* | *access-list-name*} **pool** *pool-name* [**overload**] | **static** {**tcp** | **udp**} *local-ip local-port global-ip global-port* [**no-payload**]}
5. **ip nat inside source** {**list** {*access-list-number* | *access-list-name*} **pool** *pool-name* [**overload**] | **static** [**network**] *local-network-mask global-network-mask* [**no-payload**]}
6. **ip nat outside source** {**list** {*access-list-number* | *access-list-name*} **pool** *pool-name* | **static** *local-ip global-ip* [**no-payload**]}
7. **ip nat outside source** {**list** {*access-list-number* | *access-list-name*} **pool** *pool-name* | **static** {**tcp** | **udp**} *local-ip local-port global-ip global-port* [**no-payload**]}
8. **ip nat outside source** {**list** {*access-list-number* | *access-list-name*} **pool** *pool-name* | **static** [**network**] *local-network-mask global-network-mask* [**no-payload**]}
9. **exit**
10. **show ip nat translations** [**verbose**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip nat inside source** {**list** {*access-list-number* | *access-list-name*} **pool** *pool-name* [**overload**] | **static network** *local-ip global-ip* [**no-payload**]} **Example:** `Device(config)# ip nat inside source static network 10.1.1.1 192.168.251.0/24 no-payload`	Disables the network packet translation on the inside host device.
Step 4	**ip nat inside source** {**list** {*access-list-number* | *access-list-name*} **pool** *pool-name* [**overload**] | **static** {**tcp** | **udp**} *local-ip local-port global-ip global-port* [**no-payload**]} **Example:** `Device(config)# ip nat inside source static tcp 10.1.1.1 2000 192.168.1.1 2000 no-payload`	Disables port packet translation on the inside host device.
Step 5	**ip nat inside source** {**list** {*access-list-number* | *access-list-name*} **pool** *pool-name* [**overload**] | **static**	Disables packet translation on the inside host device.

	Command or Action	Purpose
	[**network**] *local-network-mask global-network-mask* [**no-payload**]} **Example:** `Device(config)# ip nat inside source static 10.1.1.1 192.168.1.1 no-payload`	
Step 6	**ip nat outside source** {**list** {*access-list-number* | *access-list-name*} **pool** *pool-name* | **static** *local-ip global-ip* [**no-payload**]} **Example:** `Device(config)# ip nat outside source static 10.1.1.1 192.168.1.1 no-payload`	Disables packet translation on the outside host device.
Step 7	**ip nat outside source** {**list** {*access-list-number* | *access-list-name*} **pool** *pool-name* | **static** {**tcp** | **udp**} *local-ip local-port global-ip global-port* [**no-payload**]} **Example:** `Device(config)# ip nat outside source static tcp 10.1.1.1 20000 192.168.1.1 20000 no-payload`	Disables port packet translation on the outside host device.
Step 8	**ip nat outside source** {**list** {*access-list-number* | *access-list-name*} **pool** *pool-name* | **static** [**network**] *local-network-mask global-network-mask* [**no-payload**]} **Example:** `Device(config)# ip nat outside source static network 10.1.1.1 192.168.251.0/24 no-payload`	Disables network packet translation on the outside host device.
Step 9	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.
Step 10	**show ip nat translations** [**verbose**] **Example:** `Device# show ip nat translations`	Displays active NAT.

Configuring the NAT Default Inside Server Feature

The NAT Default Inside Server feature helps forward packets from the outside to a specified inside local address. Traffic that does not match any existing dynamic translations or static port translations are redirected, and packets are not dropped.

Dynamic mapping and interface overload can be configured for gaming devices. For online games, outside traffic comes on a different UDP port. If a packet is destined for an interface from outside an enterprise's network, and there is no match in the NAT table for fully extended entry or static port entry, the packet is forwarded to the gaming device using a simple static entry.

Note

- You can use this feature to configure gaming devices with an IP address different from the IP address of the PC. To avoid unwanted traffic or DoS attacks, use access lists.
- For traffic going from the PC to the outside, it is better to use a route map so that extended entries are created.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip nat inside source static** *local-ip* **interface** *type number*
4. **ip nat inside source static tcp** *local-ip local-port* **interface** *global-port*
5. **exit**
6. **show ip nat translations** [**verbose**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip nat inside source static** *local-ip* **interface** *type number* **Example:** `Device(config)# ip nat inside source static 10.1.1.1 interface Ethernet 1/1`	Enables static NAT on the interface.
Step 4	**ip nat inside source static tcp** *local-ip local-port* **interface** *global-port* **Example:** `Device(config)# ip nat inside source static tcp 10.1.1.1 23 interface 23`	(Optional) Enables the use of telnet to the device from the outside.
Step 5	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.
Step 6	**show ip nat translations** [**verbose**] **Example:** `Device# show ip nat translations`	(Optional) Displays active NAT.

Reenabling RTSP on a NAT Router

The Real Time Streaming Protocol (RTSP) is a client/server multimedia presentation control protocol that supports multimedia application delivery. Some of the applications that use RTSP include Windows Media Services (WMS) by Microsoft, QuickTime by Apple Computer, and RealSystem G2 by RealNetworks.

When the RTSP protocol passes through a NAT router, the embedded address and port must be translated for the connection to be successful. NAT uses Network Based Application Recognition (NBAR) architecture to parse the payload and translate the embedded information in the RTSP payload.

RTSP is enabled by default. Use the **ip nat service rtsp port** *port-number* command to reenable RTSP on a NAT router if this configuration has been disabled.

Configuring Support for Users with Static IP Addresses

Configuring support for users with static IP addresses enables those users to establish an IP session in a public wireless LAN environment.

Before you begin

Before configuring support for users with static IP addresses, you must first enable NAT on your router and configure a RADIUS server host.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ip nat inside**
5. **exit**
6. **ip nat allow-static-host**
7. **ip nat pool** *name start-ip end-ip* **netmask** *netmask* **accounting** *list-name*
8. **ip nat inside source list** *access-list-number* **pool** *name*
9. **access-list** *access-list-number* **deny ip** *source*
10. **end**
11. **show ip nat translations verbose**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**interface** *type number* **Example:** `Device(config)# interface ethernet 1`	Configures an interface and enters an interface configuration mode.
Step 4	**ip nat inside** **Example:** `Device(config-if)# ip nat inside`	Marks the interface as connected to the inside.
Step 5	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 6	**ip nat allow-static-host** **Example:** `Device(config)# ip nat allow-static-host`	Enables static IP address support. • Dynamic Address Resolution Protocol (ARP) learning will be disabled on this interface, and NAT will control the creation and deletion of ARP entries for the static IP host.
Step 7	**ip nat pool** *name start-ip end-ip* **netmask** *netmask* **accounting** *list-name* **Example:** `Device(config)# ip nat pool pool1 172.16.0.0 172.16.0.254 netmask 255.255.255.0 accounting WLAN-ACCT`	Specifies an existing RADIUS profile name to be used for authentication of the static IP host.
Step 8	**ip nat inside source list** *access-list-number* **pool** *name* **Example:** `Device(config)# ip nat inside source list 1 pool net-208`	Specifies the access list and pool to be used for static IP support. • The specified access list must permit all traffic.
Step 9	**access-list** *access-list-number* **deny ip** *source* **Example:** `Device(config)# access-list 1 deny ip 192.168.196.51`	Removes the traffic of the device from NAT. • The *source* argument is the IP address of the device that supports the NAT Static IP Support feature.
Step 10	**end** **Example:** `Device(config)# end`	(Optional) Exits global configuration mode and returns to privileged EXEC mode.
Step 11	**show ip nat translations verbose** **Example:** `Device# show ip nat translations verbose`	(Optional) Displays active NAT translations and additional information for each translation table entry, including how long ago the entry was created and used.

Examples

The following is sample output from the **show ip nat translations verbose** command:

```
Device# show ip nat translations verbose

--- 172.16.0.0 10.1.1.1           ---                ---
create 00:05:59, use 00:03:39, left 23:56:20, Map-Id(In): 1, flags: none wlan-flags: Secure
 ARP added, Accounting Start sent Mac-Address:0010.7bc2.9ff6 Input-IDB:Ethernet1/2, use_count:
 0, entry-id:7, lc_entries: 0
```

Configuring the Rate Limiting NAT Translation Feature

SUMMARY STEPS

1. **enable**
2. **show ip nat translations**
3. **configure terminal**
4. **ip nat translation max-entries** {*number* | **all-vrf** *number* | **host** *ip-address number* | **list** *listname number* | **vrf** *name number*}
5. **end**
6. **show ip nat statistics**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show ip nat translations** **Example:** `Device# show ip nat translations`	(Optional) Displays active NAT. • A specific host, access control list, or VRF instance generating an unexpectedly high number of NAT requests may be the source of a malicious virus or worm attack.
Step 3	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 4	**ip nat translation max-entries** {*number* | **all-vrf** *number* | **host** *ip-address number* | **list** *listname number* | **vrf** *name number*} **Example:** `Device(config)# ip nat translation max-entries 300`	Configures the maximum number of NAT entries that are allowed from the specified source. • The maximum number of allowed NAT entries is 2147483647, although a typical range for a NAT rate limit is 100 to 300 entries.

	Command or Action	Purpose
		• When you configure a NAT rate limit for all VRF instances, each VRF instance is limited to the maximum number of NAT entries that you specify. • When you configure a NAT rate limit for a specific VRF instance, you can specify a maximum number of NAT entries for the named VRF instance that is greater than or less than that allowed for all VRF instances.
Step 5	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.
Step 6	**show ip nat statistics** **Example:** `Device# show ip nat statistics`	(Optional) Displays current NAT usage information, including NAT rate limit settings. • After setting a NAT rate limit, use the **show ip nat statistics** command to verify the current NAT rate limit settings.

Configuring Bypass NAT Functionality

The Bypass NAT functionality feature reduces the TCAM size by resolving the deny jump issue. To enable the Bypass NAT functionality feature, you must:

- Create a NAT bypass pool by using a reserved loopback address (127.0.0.1).
- Create a new NAT mapping containing a new ACL with all existing deny statements that are converted to permit statements.

You can enable the Bypass NAT functionality by creating new NAT mapping with new ACL mapped to a bypass pool.

To configure the bypass-pool with 127.0.0.1 as reserved loopback address:

```
enable
configure terminal
access-list 60 permit 25.33.0.0 0.0.255.255
ip nat pool bypass-pool 127.0.0.1 127.0.0.1 prefix-length 24
ip nat inside source list 60 pool bypass-pool
end
```

To convert existing configuration with deny statements:

```
enable
configure terminal
ip access list extended nat-acl
deny ip host 10.10.10.10 host 10.77.64.17
permit ip any 10.77.64.0 0.0.15.255
ip nat inside source list nat-acl pool nat-pool
end
```

New converted configuration using bypass pool with permit statements:

```
enable
configure terminal
ip nat pool bypass-pool 127.0.0.1 127.0.0.1 prefix-length 24
ip access list extended nat-bypass-acl
permit ip host 10.10.10.10 host 10.77.64.17
ip nat inside source list nat-bypass-acl pool bypass-pool
ip access list extended nat-acl
permit ip any 10.77.64.0 0.0.15.255
ip nat inside source list nat-acl pool nat-pool
end
```

Configuration Examples for Configuring NAT for IP Address Conservation

Example: Configuring Static Translation of Inside Source Addresses

The following example shows how inside hosts addressed from the 10.114.11.0 network are translated to the globally unique 172.31.233.208/28 network. Further, packets from outside hosts that are addressed from the 10.114.11.0 network (the true 10.114.11.0 network) are translated to appear from the 10.0.1.0/24 network.

```
ip nat pool net-208 172.31.233.208 172.31.233.223 prefix-length 28
ip nat pool net-10 10.0.1.0 10.0.1.255 prefix-length 24
ip nat inside source list 1 pool net-208
ip nat outside source list 1 pool net-10
!
interface gigabitethernet 0/0/0
 ip address 172.31.232.182 255.255.255.240
 ip nat outside
!
interface gigabitethernet 1/1/1
 ip address 10.114.11.39 255.255.255.0
 ip nat inside
!
access-list 1 permit 10.114.11.0 0.0.0.255
```

The following example shows NAT configured on the provider edge (PE) device with a static route to the shared service for the vrf1 and vrf2 VPNs. NAT is configured as inside source static one-to-one translation.

```
ip nat pool outside 10.4.4.1 10.4.4.254 netmask 255.255.255.0
ip nat outside source list 1 pool mypool
access-list 1 permit 172.16.18.0 0.0.0.255
ip nat inside source static 192.168.121.33 10.2.2.1 vrf vrf1
ip nat inside source static 192.168.121.33.10.2.2.2 vrf vrf2
```

Example: Configuring Dynamic Translation of Inside Source Addresses

The following example shows how inside hosts addressed from either the 192.168.1.0 or the 192.168.2.0 network are translated to the globally unique 172.31.233.208/28 network:

```
ip nat pool net-208 172.31.233.208 172.31.233.223 prefix-length 9
ip nat inside source list 1 pool net-208
```

```
!
interface gigabitethernet 0/0/0
 ip address 172.31.232.182 255.255.255.240
 ip nat outside
!
interface gigabitethernet 1/1/1
 ip address 192.168.1.94 255.255.255.0
 ip nat inside
!
access-list 1 permit 192.168.1.0 0.0.0.255
access-list 1 permit 192.168.2.0 0.0.0.255
!
```

The following example shows how only traffic local to the provider edge (PE) device running NAT is translated:

```
ip nat inside source list 1 interface gigabitethernet 0/0/0 vrf vrf1 overload
ip nat inside source list 1 interface gigabitethernet 0/0/0 vrf vrf2 overload
!
ip route vrf vrf1 0.0.0.0 0.0.0.0 192.168.1.1
ip route vrf vrf2 0.0.0.0 0.0.0.0 192.168.1.1
!
access-list 1 permit 10.1.1.1.0 0.0.0.255
!
ip nat inside source list 1 interface gigabitethernet 1/1/1 vrf vrf1 overload
ip nat inside source list 1 interface gigabitethernet 1/1/1 vrf vrf2 overload
!
ip route vrf vrf1 0.0.0.0 0.0.0.0 172.16.1.1 global
ip route vrf vrf2 0.0.0.0 0.0.0.0 172.16.1.1 global
access-list 1 permit 10.1.1.0 0.0.0.255
!
```

Example: Using NAT to Allow Internal Users Access to the Internet

The following example shows how to create a pool of addresses that is named net-208. The pool contains addresses from 172.31.233.208 to 172.31.233.233. Access list 1 allows packets with SA from 192.168.1.0 to 192.168.1.255. If no translation exists, packets matching access list 1 is translated to an address from the pool. The router allows multiple local addresses (192.168.1.0 to 192.168.1.255) to use the same global address. The router retains port numbers to differentiate the connections.

```
ip nat pool net-208 172.31.233.208 172.31.233.233 netmask 255.255.255.240
access-list 1 permit 192.168.1.0 0.0.0.255
ip nat inside source list 1 pool net-208 overload
interface gigabitethernet 1/1/1
 ip address 192.168.201.1 255.255.255.240
 ip nat inside
!
interface gigabitethernet 0/0/0
 ip address 192.168.201.29 255.255.255.240
 ip nat outside
!
```

Example: Allowing Overlapping Networks to Communicate Using NAT

Example: Configuring Static Translation of Overlapping Networks

```
ip nat inside source static 192.168.121.33 10.2.2.1
!
interface gigabitethernet 1/1/1
 ip address 10.114.11.39 255.255.255.0
 ip nat inside
!
interface gigabitethernet 0/0/0
 ip address 172.16.232.182 255.255.255.240
 ip nat outside
!
```

Example: Configuring Dynamic Translation of Overlapping Networks

In the following example, the addresses in the local network are being used legitimately by someone else on the Internet. An extra translation is required to access the external network. The pool net-10 is a pool of outside local IP addresses. The **ip nat outside source list 1 pool** *net-10* command translates the addresses of hosts from the outside overlapping network to addresses in that pool.

```
ip nat pool net-208 172.31.233.208 172.31.233.223 prefix-length 28
ip nat pool net-10 10.0.1.0 10.0.1.255 prefix-length 24
access-list 1 permit 10.114.11.0 0.0.0.255
ip nat inside source list 1 pool net-208
ip nat outside source list 1 pool net-10
!
interface gigabitethernet 1/1/1
 ip address 10.114.11.39 255.255.255.0
 ip nat inside
!
interface gigabitethernet 0/0/0
 ip address 172.16.232.182 255.255.255.240
 ip nat outside
!
```

Example: Configuring Server TCP Load Balancing

In the following example, the goal is to define a virtual address, connections to which are distributed among a set of real hosts. The pool defines addresses of real hosts. The access list defines the virtual address. If a translation does not exist, TCP packets from serial interface 0 (the outside interface), whose destination matches the access list, are translated to an address from the pool.

```
ip nat pool real-hosts 192.168.15.2 192.168.15.15 prefix-length 28 type rotary
access-list 2 permit 192.168.15.1
ip nat inside destination list 2 pool real-hosts
interface gigabitethernet 0/0/0
 ip address 192.168.15.129 255.255.255.240
 ip nat inside
!
interface serial 0
 ip address 192.168.15.17 255.255.255.240
```

```
 ip nat outside
!
```

Example: Enabling Route Maps on Inside Interfaces

```
ip nat inside source static 192.168.201.6 192.168.201.21
!
```

Example: Enabling NAT Route Maps Outside-to-Inside Support

The following example shows how to configure a route map A and route map B to allow outside-to-inside translation for a destination-based Network Address Translation (NAT):

```
ip nat pool POOL-A 192.168.201.4 192.168.201.6 netmask 255.255.255.128
ip nat pool POOL-B 192.168.201.7 192.168.201.9 netmask 255.255.255.128
ip nat inside source route-map MAP-A pool POOL-A reversible
ip nat inside source route-map MAP-B pool POOL-B reversible
```

Example: Configuring NAT of External IP Addresses Only

```
ip nat inside source static network 10.1.1.1 192.168.2510/24 no-payload
ip nat inside source static tcp 10.1.1.1 2000 192.168.1.1 2000 no-payload
ip nat inside source static 10.1.1.1 192.168.1.1 no-payload
ip nat outside source static 10.1.1. 192.168.1.1 no-payload
ip nat outside source static tcp 10.1.1.1 20000 192.168.1.1 20000 no-payload
ip nat outside source static network 10.1.1.1 192.168.251.0/24 no-payload
```

Example: Configuring Support for Users with Static IP Addresses

```
interface gigabitethernet 1/1/1
 ip nat inside
!
ip nat allow-static-host
ip nat pool pool1 172.16.0.0 172.16.0.254 netmask 255.255.255.0 accounting WLAN-ACCT
ip nat inside source list 1 pool net-208
access-list 1 deny ip 192.168.196.51
```

Example: Configuring NAT Static IP Support

The following example shows how to enable static IP address support for the device at 192.168.196.51:

```
interface gigabitethernet 1/1/1
 ip nat inside
!
ip nat allow-static-host
ip nat pool net-208 172.16.1.1 172.16.1.10 netmask 255.255.255.0 accounting WLAN-ACCT
ip nat inside source list 1 pool net-208
access-list 1 deny ip 192.168.196.51
```

Example: Creating a RADIUS Profile for NAT Static IP Support

The following example shows how to create a RADIUS profile for use with the NAT Static IP Support feature:

```
aaa new-model
!
aaa group server radius WLAN-RADIUS
 server 172.16.88.1 auth-port 1645 acct-port 1645
 server 172.16.88.1 auth-port 1645 acct-port 1646
!
aaa accounting network WLAN-ACCT start-stop group WLAN-RADIUS
aaa session-id common
ip radius source-interface gigabitethernet3/0
radius-server host 172.31.88.1 auth-port 1645 acct-port 1646
radius-server key cisco
```

Example: Configuring the Rate Limiting NAT Translation Feature

The following example shows how to limit the maximum number of allowed NAT entries to 300:

```
ip nat translation max-entries 300
```

The following example shows how to limit the VRF instance named “vrf1” to 150 NAT entries:

```
ip nat translation max-entries vrf vrf1 150
```

The following example shows how to limit each VRF instance to 200 NAT entries:

```
ip nat translation max-entries all-vrf 200
```

The following example shows how to limit the VRF instance, “vrf2” to 225 NAT entries, but limit all other VRF instances to 100 NAT entries each:

```
ip nat translation max-entries all-vrf 100
ip nat translation max-entries vrf vrf2 225
```

The following example shows how to limit the access control list named “vrf3” to 100 NAT entries:

```
ip nat translation max-entries list vrf3 100
```

The following example shows how to limit the host at IP address 10.0.0.1 to 300 NAT entries:

```
ip nat translation max-entries host 10.0.0.1 300
```

Example: Setting a Global NAT Rate Limit

The following example shows how to limit the maximum number of allowed NAT entries to 300:

```
ip nat translation max-entries 300
```

Example: Setting NAT Rate Limits for a Specific VRF Instance

The following example shows how to limit the VRF instance named “vrf1” to 150 NAT entries:

```
ip nat translation max-entries vrf vrf1 150
```

Example: Setting NAT Rate Limits for All VRF Instances

The following example shows how to limit each VRF instance to 200 NAT entries:

```
ip nat translation max-entries all-vrf 200
```

The following example shows how to limit the VRF instance, “vrf2” to 225 NAT entries, but limit all other VRF instances to 100 NAT entries each:

```
ip nat translation max-entries all-vrf 100
ip nat translation max-entries vrf vrf2 225
```

Example: Setting NAT Rate Limits for Access Control Lists

The following example shows how to limit the access control list named “vrf3” to 100 NAT entries:

```
ip nat translation max-entries list vrf3 100
```

Example: Setting NAT Rate Limits for an IP Address

The following example shows how to limit the host at IP address 10.0.0.1 to 300 NAT entries:

```
ip nat translation max-entries host 10.0.0.1 300
```

Where to Go Next

- To configure NAT for use with application-level gateways, see the “Using Application Level Gateways with NAT” module.
- To verify, monitor, and maintain NAT, see the “Monitoring and Maintaining NAT” module.
- To integrate NAT with Multiprotocol Label Switching (MPLS) VPNs, see the “Integrating NAT with MPLS VPNs” module.
- To configure NAT for high availability, see the “Configuring NAT for High Availability” module.

Additional References for Configuring NAT for IP Address Conservation

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
NAT commands: complete command syntax, command mode command history, defaults, usage guidelines, and examples	Cisco IOS IP Addressing Services Command Reference
Application-level gateways	*Using Application Level Gateways with NAT* module
IP access list sequence numbering	IP Access List Entry Sequence Numbering document
RADIUS attributes overview	*RADIUS Attributes Overview and RADIUS IETF Attributes* module

Standards and RFCs

Standard/RFC	Title
IETF Behave Draft NAT MIB	Definitions of Managed Objects for Network Address Translators (NAT) draft-ietf-behave-nat-mib-11
RFC 1597	Internet Assigned Numbers Authority
RFC 1631	The IP Network Address Translation (NAT)
RFC 1918	Address Allocation for Private Internets
RFC 2663	IP Network Address Translation (NAT) Terminology and Considerations
RFC 3022	Traditional IP Network Address Translation (Traditional NAT)

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services. These services are the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/support

Feature Information for Configuring NAT for IP Address Conservation

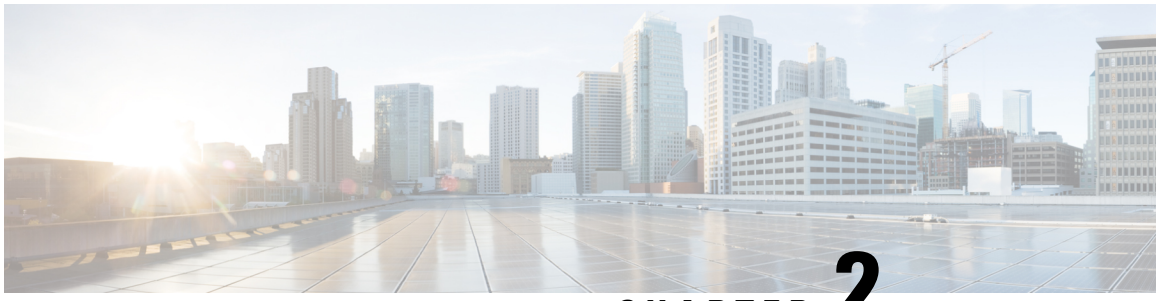
Table 2: Feature Information for Configuring NAT for IP Address Conservation

Feature Name	Releases	Feature Information
Destination-Based NAT Using Route Maps	Cisco IOS XE Release 2.1	The Destination-Based NAT Using Route Maps feature adds support for destination-based NAT using route maps.
NAT Duplicate Inside Global Address	Cisco IOS XE Release 2.1	The Cisco IOS XE software supports the NAT Duplicate Inside Global Addresses feature.

Feature Name	Releases	Feature Information
NAT Host Number Preservation	Cisco IOS XE Release 2.1	For ease of network management, some sites prefer to translate prefixes rather than addresses. These sites want the translated address to have the same host number as the original address. The two prefixes must be of the same length. The NAT Host Number Preservation feature can be enabled by configuring dynamic translation with the address pool of the type, match-host.
NAT MIB Support	Cisco IOS XE Release 3.15S	The NAT MIB Support feature supports, IETF Behave Draft, Definitions of Managed Objects for Network Address Translators (NAT). Only regular NAT is supported, NAT 64 is not supported.
NAT Performance Enhancement—Translation Table Optimization	Cisco IOS XE Release 2.1	The NAT Performance Enhancement—Translation Table Optimization feature provides greater structure for storing translation table entries and an optimized lookup in the table. The optimized lookup table enables associating table entries to IP connections.
NAT Route Maps Outside-to-Inside Support	Cisco IOS XE Release 2.2	The NAT Route Maps Outside-to-Inside Support feature enables the deployment of a NAT route map configuration that will allow IP sessions to be initiated from the outside to the inside.
NAT Static IP Support	Cisco IOS XE Release 2.1	The NAT Static IP Support feature provides support for users with static IP addresses. It enables those users to establish an IP session in a public wireless LAN environment.
NAT Timers	Cisco IOS XE Release 2.1	The NAT Timers feature allows you to change the amount of time after which NAT translations time out.
NAT Translation of External IP Addresses Only	Cisco IOS XE Release 2.1	To configure NAT for ignoring all embedded IP addresses of any application and traffic type, use the NAT Translation of External IP Address Only feature .
Rate Limiting NAT Translation	Cisco IOS XE Release 2.1	The Rate Limiting NAT Translation feature provides the ability to limit the maximum number of concurrent Network Address Translation (NAT) operations on a router. In addition to giving users more control over how NAT addresses are used, the Rate Limiting NAT Translation feature can be used to limit the effects of viruses, worms, and denial-of-service attacks.

Feature Name	Releases	Feature Information
VRF Scale Increase in NAT	Cisco IOS XE Release 3.10S	The VRF Scale Increase in NAT feature provides the ability to increase the number of virtual routing and forwarding (VRF) instances that are supported on NAT to 4000. This feature is enabled by default when NAT is configured. You cannot disable this configuration. No commands were introduced or modified for this feature. In Cisco IOS XE Release 3.10S, support was added for Cisco CSR 1000V Series Routers. In Cisco IOS XE Release 3.10S, support was added for Cisco ISR 4400 Series Routers.
NAT support on BDI interface	Cisco IOS XE Denali 16.3.1	The NAT support on BDI interface feature enables you to configure NAT on Bridge Domain Interface (BDI). No commands were introduced or modified for this feature.
Multi-Tenant support for NAT	Cisco IOS XE Denali 16.3.1	With Multi-Tenant support for NAT feature, the configuration changes of a VRF instance do not interrupt the traffic flow of other VRFs in the network. No commands were introduced or modified for this feature.
Bypass NAT functionality	Cisco IOS XE Denali 16.3.2 Cisco IOS XE Everest 16.4.1	The Bypass NAT functionality feature enables you to permit an ACL with deny statements using a bypass pool. The Bypass NAT functionality feature reduces the TCAM size by resolving the deny jump issue. No commands were introduced or modified for this feature. In Cisco IOS XE Denali 16.3 everest Bypass NAT functionality .2, support was added for Cisco Cloud Services Router 1000V Series, Cisco ASR 1000 Series Routers with Route Processors (RP2 and RP3), Cisco 4000 Series Integrated Services Routers. In Cisco IOS XE Everest 16.4.1, support was extended to Cisco ASR 1001-HX Router, Cisco ASR 1001-X Router, Cisco ASR 1002-HX Router, Cisco ASR 1002-X Router.

Feature Name	Releases	Feature Information
IP Address Port Parity and Conservation	Cisco IOS XE Everest 16.5.1b	Added information on NAT pool overload of RTP packets during ALG processing. No commands were introduced or modified for this feature.



CHAPTER 2

Using Application-Level Gateways with NAT

This module describes the basic tasks to configure an application-level gateway (ALG) with Network Address Translation (NAT). This module also provides information about the protocols that use ALGs for IP header translation.

NAT performs translation services on any TCP/UDP traffic that does not carry source and destination IP addresses in the application data stream. Protocols that do not carry the source and destination IP addresses include HTTP, TFTP, telnet, archie, finger, Network Time Protocol (NTP), Network File System (NFS), remote login (rlogin), remote shell (rsh) protocol, and remote copy (rcp).

Specific protocols that embed the IP address information within the payload require the support of an ALG. NAT requires a variety of ALGs to handle application data stream (Layer 7) protocol-specific services such as translating embedded IP addresses and port numbers in the packet payload and extracting new connection/session information from control channels.

NAT supports virtual routing and forwarding (VRF) for protocols that have a supported ALG.

The Support for IPsec ESP Through NAT feature provides the ability to support multiple concurrent IPsec Encapsulating Security Payload (ESP) tunnels or connections through a NAT device configured in Overload or Port Address Translation (PAT) mode. You can use the **ip nat service dns-v6** command to control processing of IPv6 DNS packets by ALG

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Using Application Level Gateways with NAT

- Before performing the tasks in this module, you should be familiar with the concepts described in the "Configuring NAT for IP Address Conservation" module.
- All access lists required for use with the tasks in this module should be configured prior to beginning the configuration task. For information about how to configure an access list, see the "IP Access List Sequence Numbering" document.
- Before performing the tasks in this module, you should verify that the Session Initiation Protocol (SIP) and H.323 are not disabled. SIP and H.323 are enabled by default.

Information About Using Application-Level Gateways with NAT

IPsec

IPsec is a set of extensions to the IP protocol family in a framework of open standards for ensuring secure private communications over the Internet. Based on standards developed by the IETF, IPsec ensures confidentiality, integrity, and authenticity of data communications across the public network and provides cryptographic security services.

Secure tunnels between two peers, such as two routers, are provided and decisions are made as to which packets are considered sensitive and should be sent through these secure tunnels, and which parameters should be used to protect these sensitive packets by specifying characteristics of these tunnels. When the IPsec peer receives a sensitive packet, it sets up the appropriate secure tunnel and sends the packet through the tunnel to the remote peer.

IPsec using Encapsulating Security Payload (ESP) can pass through a router running NAT without any specific support from it as long as Network Address Port Translation (NAPT) or address overloading is not configured. You can enable IPsec packet processing using ESP with the **ip nat service ipsec-esp enable** command.

There are a number of factors to consider when attempting an IPsec VPN connection that traverses a NAPT device that represents multiple private internal IP addresses as a single public external IP address. Such factors include the capabilities of the VPN server and client, the capabilities of the NAPT device, and whether more than one simultaneous connection is attempted across the NAPT device.

There are two possible methods for configuring IPsec on a router with NAPT:

- Encapsulate IPsec in a Layer 4 protocol such as TCP or UDP. In this case, IPsec is *sneaking* through NAT. The NAT device is unaware of the encapsulation.
- Add IPsec-specific support to NAPT. IPsec works with NAT in this case as opposed to *sneaking* through NAT. The NAT Support for IPsec ESP-- Phase II feature provides support for Internet Key Exchange (IKE) and ESP without encapsulation in tunnel mode through a Cisco IOS router configured with NAPT.

We recommend that TCP and UDP be used when conducting IPsec sessions that traverse a NAPT device. However, not all VPN servers or clients support TCP or UDP.

SPI Matching

SPI matching is used to establish VPN connections between multiple pairs of destinations. NAT entries will immediately be placed in the translation table for endpoints matching the configured access list..

Benefits of Configuring NAT IPsec

- NAT enables customers to deploy private IP addresses within their network and translate the private IP addresses to public IP addresses when connecting to the Internet or when interconnecting with another corporate network.
- NAT support for the Session Initiation Protocol (SIP) adds the ability to deploy NAT on VoIP solutions based on SIP.
- With NAT ALGs, customers can control their IP address scheme and include complete support for H.323 v2 gatekeeper designs.
- ESP entries in the translation table are normally delayed from being transmitted until a reply is received from the destination. With predictable security parameter indexes (SPIs) and SPI matching, the delay can be eliminated because SPI entries are matched. Some third-party concentrators require both source ports and incoming ports to use port 500. Use the **ip nat service preserve-port** command to preserve the ports rather than changing them, which is required with regular NAT.

Voice and Multimedia over IP Networks

SIP is a protocol developed by the IETF Multiparty Multimedia Session Control (MMUSIC) Working Group. The Cisco SIP functionality equips Cisco routers to signal the setup of voice and multimedia calls over IP networks. SIP provides an alternative to H.323 within the VoIP internetworking software.

Session Description Protocol (SDP) is a protocol that describes multimedia sessions. SDP may be used in SIP message bodies to describe multimedia sessions used for creating and controlling multimedia sessions with two or more participants.

The NAT Support for SIP feature allows SIP embedded messages passing through a router configured with NAT to be translated and encoded back to the packet. An ALG is used with NAT to translate the SIP or SDP messages.

Note By default support for SIP is enabled on port 5060. Therefore, NAT-enabled devices interpret all packets on this port as SIP call messages. If other applications in the system use port 5060 to send packets, the NAT service may corrupt the packet as it attempts to interpret the packet as a SIP call message.

NAT Support of H.323 v2 RAS

Cisco IOS NAT supports all H.225 and H.245 message types, including those sent in the Registration, Admission, and Status (RAS) protocol. RAS provides a number of messages that are used by software clients and VoIP devices to register their location, request assistance in call setup, and control bandwidth. The RAS messages are directed toward an H.323 gatekeeper.

Some RAS messages include IP addressing information in the payload, typically meant to register a user with the gatekeeper or learn about another user already registered. If these messages are not known to NAT, they cannot be translated to an IP address that will be visible to the public.

In Cisco IOS Release 12.2(2)T and later releases, embedded IP addresses can be inspected for potential address translation. Prior to Cisco IOS Release 12.2(2)T, NAT did not support H.323 v2 RAS messages.

NAT Support for H.323 v3 and v4 in v2 Compatibility Mode

H.323 is an ITU-T specification for transmitting audio, video, and data across packet networks. NAT supports four versions of the H.323 protocols: Version 1, Version 2, Version 3, and Version 4. The NAT Support for H.323 v3 and v4 in v2 Compatibility Mode feature enables NAT routers to support messages coded in H.323 Version 3 and Version 4 when these messages contain fields that are compatible with H.323 Version 2. This feature does not support H.323 capabilities introduced in H.323 Version 3 and Version 4, such as new message types or new fields that require address translation.

NAT H.245 Tunneling Support

The NAT H.245 Tunneling Support feature supports H.245 tunneling in H.323 ALGs. The H.245 tunneling supports H.245 tunnel messages that are needed to create a media channel setup.

For an H.323 call to take place, an H.225 connection on TCP port 1720 must be opened. When the H.225 connection is opened, the H.245 session is initiated and established. The H.323 connection can take place on a separate channel other than the H.225 or it can be done by using H.245 tunneling on the same H.225 channel whereby the H.245 messages are embedded in H.225 messages and sent on the previously established H.225 channel.

If the H.245 tunneled message is not understood by NAT, the media address or the port number is left untranslated by NAT, resulting in media traffic failure. The H.245 FastConnect procedures will not help if the H.245 tunneled message is not understood by NAT because FastConnect is terminated as soon as an H.245 tunneled message is sent.

NAT Support of Skinny Client Control Protocol

Cisco IP phones use the SCCP to connect with and register to Cisco CallManager.

To be able to configure Cisco IOS NAT between the IP phone and Cisco CallManager in a scalable environment, NAT needs to be able to detect the SCCP and understand the information passed within the messages. Messages flow back and forth that include IP address and port information used to identify other IP phone users with which a call can be placed.

The SCCP client to Cisco CallManager communication typically flows from inside to outside. Domain Name System (DNS) should be used to resolve the Cisco CallManager IP address connection when the Cisco CallManager is on the inside (behind the NAT device), or static NAT should be configured to reach the Cisco CallManager in the inside.

When an IP phone attempts to connect to the Cisco CallManager and it matches the configured NAT rules, NAT will translate the original source IP address and replace it with one from the configured pool. This new address will be reflected in the Cisco CallManager and be visible to other IP phone users.

NAT Support of SCCP Fragmentation

Skinny Client Control Protocol (SCCP) messages, also called Skinny control messages, are exchanged over TCP. If either the IP phone or the Cisco Unified CallManager is configured to have a TCP maximum segment size (MSS) lower than the Skinny control message payload, the Skinny control message is segmented across multiple TCP segments. Prior to the introduction of this feature, Skinny control message exchanges used to fail during TCP segmentation because the NAT Skinny ALG was not able to reassemble Skinny control messages. The NAT SCCP Fragmentation Support feature adds support for TCP segments for the NAT Skinny ALG and fragmented payloads that requires an IP translation or a port translation is no longer dropped.

Skinny control messages can also be IP fragmented by using Virtual Fragmentation Reassembly (VFR).

In Cisco IOS Release 15.1(3)T and later releases, NAT works with SCCP phones Version 17 and higher.

NAT Segmentation with Layer 4 Forwarding

The NAT Segmentation with Layer 4 Forwarding feature is implemented for the H.323, Skinny Client Control Protocol (SCCP), and the TCP Domain Name System (DNS) protocol. NAT supports the processing of segmented H.323, SCCP, or TCP DNS messages that are split across multiple packets.

Layer 4 forwarding or TCP proxy is responsible for session handling that includes setting sequence numbers in order, acknowledging the numbers in a packet, resegmenting the translated packet if it is larger than the maximum segment size (MSS), and handling retransmissions in case of packet loss. Layer 4 forwarding also handles out-of-order packets and these packets are buffered and not dropped. Layer 4 forwarding buffers received packets and notifies the NAT ALG when an in-order packet is available, sends acknowledgments to end hosts for received packets, and sends translated packets that it receives from the NAT ALG back into the output packet path.

Restrictions

The NAT Segmentation with Layer 4 Forwarding feature does not work when:

- Firewalls are configured using the **ip inspect name** command. (Context-Based Access Control (CBAC) firewalls are not supported. Zone-based firewalls are supported.)
- H.323, SCCP, or TCP DNS messages are larger than 18 KB.
- Multiprotocol Label Switching (MPLS) is configured.
- NAT and the Cisco Unified CallManager are configured on the same device. In this case, a colocated solution in Call Manager Express is used.
- NAT Virtual Interface (NVI) is configured.
- Stateful Network Address Translation (SNAT) is enabled.

Note Effective January 31, 2014, Stateful NAT is not available in Cisco IOS software. For more information, see End-of-Sale and End-of-Life Announcement for the Cisco IOS Stateful Failover of Network Address Translation (SNAT).

- The **match-in-vrf** keyword is configured along with the **ip nat inside source** command for packet translation.

- The packets are IPv6 packets.

How to Configure Application-Level Gateways with NAT

Configuring IPsec Through NAT

Configuring IPsec ESP Through NAT

IPsec ESP Through NAT provides the ability to support multiple concurrent IPsec ESP tunnels or connections through a Cisco IOS NAT device configured in Overload or PAT mode.

Perform this task to configure IPsec ESP through NAT.

Note IPsec can be configured for any NAT configuration, not just static NAT configurations.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip nat** [**inside** | **outside**] **source static** *local-ip global-ip* [**vrf** *vrf-name*]
4. **exit**
5. **show ip nat translations**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ip nat** [**inside** | **outside**] **source static** *local-ip global-ip* [**vrf** *vrf-name*] **Example:** `Router(config)# ip nat inside source static 10.10.10.10 192.168.30.30`	Enables static NAT.

	Command or Action	Purpose
Step 4	**exit** **Example:** `Router(config)# exit`	Returns to privileged EXEC mode.
Step 5	**show ip nat translations** **Example:** `Router# show ip nat translations`	(Optional) Displays active NATs.

Enabling the Preserve Port

This task is used for IPsec traffic using port 500 for the source port. Perform this task to enable port 500 to be preserved for the source port.

Note This task is required by certain VPN concentrators. Cisco VPN devices generally do not use this feature.

>

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip nat service list** *access-list-number* **IKE preserve-port**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ip nat service list** *access-list-number* **IKE preserve-port** **Example:** `Router(config)# ip nat service list 10 IKE preserve-port`	Specifies IPsec traffic that matches the access list to preserve the port.

Enabling SPI Matching on the NAT Device

Note SPI matching is disabled by default.

Security parameter index (SPI) matching is used to establish VPN connections between multiple pairs of destinations. NAT entries are immediately placed in the translation table for endpoints matching the configured access list. SPI matching is available only for endpoints that choose SPIs according to the predictive algorithm implemented in Cisco IOS Release 12.2(15)T.

The generation of SPIs that are predictable and symmetric is enabled. SPI matching should be used in conjunction with NAT devices when multiple ESP connections across a NAT device are desired.

Before you begin

Cisco IOS software must be running on both the source router and the remote gateway enabling parallel processing.

Note SPI matching must be configured on the NAT device and both endpoint devices.

>

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip nat service list** *access-list-number* **ESP spi-match**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ip nat service list** *access-list-number* **ESP spi-match** **Example:** `Router(config)# ip nat service list 10 ESP spi-match`	Specifies an access list to enable SPI matching. • This example shows how to enter ESP traffic matching list 10 into the NAT table, making the assumption that both devices are Cisco devices and are configured to provide matchable SPIs.

Enabling SPI Matching on Endpoints

Before you begin

Cisco software must be running on both the source device and the remote gateway, enabling parallel processing.

Note Security parameter index (SPI) matching must be configured on the Network Address Translation (NAT) device and on both endpoint devices.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto ipsec nat-transparency spi-matching**
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**crypto ipsec nat-transparency spi-matching** **Example:** `Device(config)# crypto ipsec nat-transparency spi-matching`	Enables SPI matching on both endpoints.
Step 4	**end** **Example:** `Device(config)# end`	Exits global configuration mode and enters privileged EXEC mode.

Enabling MultiPart SDP Support for NAT

The MultiPart SDP Support for NAT feature provides support for the multipart Session Description Protocol (SDP) in a SIP ALG. MultiPart SDP support for NAT is disabled by default.

Note NAT translates only embedded IPv4 addresses.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip nat service allow-multipart**
4. **exit**
5. **show ip nat translations**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip nat service allow-multipart** **Example:** `Device(config)# ip nat service allow-multipart`	Enables multipart SDP.
Step 4	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and enters privileged EXEC mode.
Step 5	**show ip nat translations** **Example:** `Device# show ip nat translations`	(Optional) Displays active NATs.

Configuring NAT Between an IP Phone and Cisco CallManager

This section describes configuring Cisco's Skinny Client Control Protocol (SCCP) for Cisco IP phone to Cisco CallManager communication. The task in this section configures NAT between an IP phone and Cisco CallManager.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip nat service skinny tcp port** *number*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ip nat service skinny tcp port** *number* **Example:** `Router(config)# ip nat service skinny tcp port 20002`	Configures the skinny protocol on the specified TCP port.

Configuration Examples for Using Application-Level Gateways with NAT

Example: Specifying a Port for NAT Translation

```
ip nat service skinny tcp port 20002
```

Example: Enabling the Preserve Port

The following example shows how to configure TCP port 500 of the third-party concentrator. Access list 10 is configured.

```
ip nat service list 10 IKE preserve-port
access-list 10 permit 10.1.1.1
```

Example Enabling SPI Matching

The following example shows how to enable SPI matching. Access list 10 is configured:

```
ip nat service list 10 ESP spi-match
access-list 10 permit 10.1.1.1
```

Example: Enabling SPI Matching on Endpoints

```
crypto ipsec nat-transparency spi-matching
```

Example: Enabling MultiPart SDP Support for NAT

```
ip nat service allow-multipart
```

Example: Specifying a Port for NAT Translation

```
ip nat service skinny tcp port 20002
```

Where to Go Next

- To learn about NAT and configure NAT for IP address conservation, see the "Configuring NAT for IP Address Conservation" module.
- To verify monitor, and maintain NAT, see the "Monitoring and Maintaining NAT" module.
- To integrate NAT with MPLS VPNs, see the "Integrating NAT with MPLS VPNs" module.
- To configure NAT for high availability, see the "Configuring NAT for High Availability" module.

Additional References for Using Application-Level Gateways with NAT

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
NAT commands: complete command syntax, command mode, defaults, usage guidelines, and examples	Cisco IOS IP Addressing Services Command Reference
IP access list sequence numbering	*IP Access List Sequence Numbering*
NAT IP address conservation	*Configuring NAT for IP Address Conservation*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

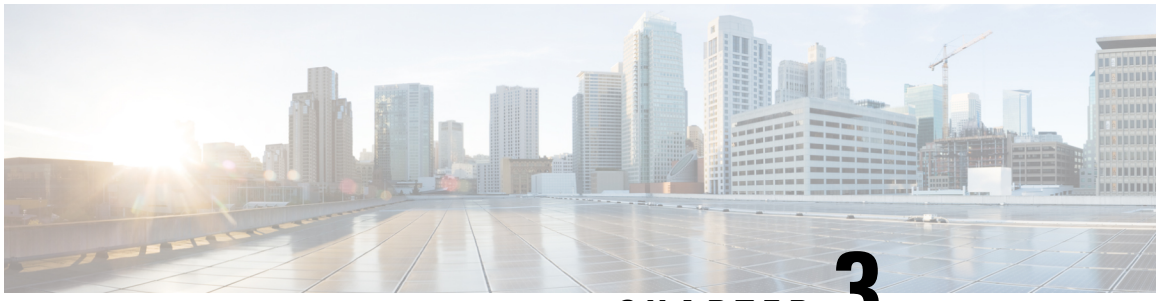
Feature Information for Using Application-Level Gateways with NAT

Table 3: Feature Information for Using Application-Level Gateways with NAT

Feature Name	Releases	Feature Configuration Information
ALG—H.323 v6 Support	Cisco IOS XE Release 3.6S	The ALG—H.323 v6 supports the parsing of H.323 v6 packets and the inspection and translation of IPv4 address information in H.323 messages.
ALG—SCCP Version 17 Support	Cisco IOS XE Release 3.5S	The ALG—SCCP Version 17 Support feature enables the SCCP ALG to parse SCCP Version 17 packets. Cisco Unified Communications Manager 7.0 and IP phones that use Cisco Unified Communications Manager 7.0 support only SCCP Version 17 messages. The SCCP Version 17 packets support IPv6 packets. The SCCP ALG supports the inspection and translation of IPv4 address information in SCCP messages.
NAT ALG—SIP REFER Method	Cisco IOS XE Release 3.2S	The NAT ALG—SIP REFER method feature supports two types of call transfers, unattended (blind) transfer and attended (consultative) transfer.
NAT ALG—SIP Trunking Support	Cisco IOS XE Release 3.2S	The NAT ALG—SIP Trunking Support feature uses a local database to store all media-related information within a SIP trunk. Call IDs of each call are used to index this local database.
NAT Basic H.323 ALG Support	Cisco IOS XE Release 2.1	NAT requires a variety of ALGs to handle Layer 7 protocol-specific services such as translating embedded IP addresses and port numbers in the packet payload and extracting new connection/session information from control channels. The NAT Basic H.323 ALG support feature provides these specific services for H.323 messages.
NAT DNS ALG Support	Cisco IOS XE Release 2.1	The NAT DNS ALG Support feature supports translation of DNS packets.
NAT FTP ALG Support	Cisco IOS XE Release 2.1	The NAT FTP ALG Support feature supports translation of FTP packets.

Feature Name	Releases	Feature Configuration Information
NAT H.323 RAS	Cisco IOS XE Release 2.4	NAT supports all H.225 and H. 245 message types, including those sent in the Registration, Admission, and Status (RAS) protocol. RAS provides a number of messages that are used by software clients and VoIP devices to register their location, request assistance in call setup, and control bandwidth. The RAS messages are directed toward an H.323 gatekeeper.
NAT ICMP ALG Support	Cisco IOS XE Release 2.1	The NAT ICMP ALG Support feature supports translation of ICMP packets.
NAT NetBIOS ALG Support	Cisco IOS XE Release 3.1S	NAT provides Network Basic Input Output System (NetBIOS) message translation support. The NAT NetBIOS ALG Support feature introduced the following command to display NetBIOS-specific information for a device: **show platform hardware qfp [active | standby] feature alg statistics netbios**.
NAT NetMeeting Directory (LDAP)	Cisco IOS XE Release 2.4	The NAT NetMeeting Directory (LDAP) feature provides ALG support for NetMeeting directory LDAP messages.
NAT RCMD ALG Support	Cisco IOS XE Release 3.1S	NAT provides remote command execution service (RCMD) message translation support.The NAT RCMD ALG Support feature introduced the following command to display RCMD-specific information for a device: **show platform software trace message process qfp active**.
NAT RTSP ALG Support	Cisco IOS XE Release 3.1S	The NAT RTSP ALG Support feature provides RTSP message translation support.
NAT—SCCP for Video	Cisco IOS XE Release 2.4	The NAT—SCCP for Video feature provides SCCP video message translation support.
NAT—SIP ALG Enhancement for T.38 Fax Relay	Cisco IOS XE Release 2.4.1	The NAT—SIP ALG Enhancement for T.38 Fax Relay feature provides translation support for SIP ALG support of T.38 Fax Relay over IP.
NAT—SIP Extended Methods	Cisco IOS XE Release 2.4	The NAT—SIP Extended Methods feature supports extended methods for SIP.
NAT Support of IP Phone to Cisco CallManager	Cisco IOS XE Release 2.1	The NAT Support of IP Phone to Cisco CallManager feature adds NAT support for configuring Cisco SCCP for a Cisco IP phone-to-Cisco CallManager communication.

Feature Name	Releases	Feature Configuration Information
NAT Support for IPsec ESP—Phase II	Cisco IOS XE Release 2.1	The NAT Support for IPsec ESP-- Phase II feature provides support for Internet Key Exchange (IKE) and ESP without encapsulation in tunnel mode through a device configured with NAPT.
NAT Support for SIP	Cisco IOS XE Release 2.1 Cisco IOS XE Release 3.2S	The NAT Support for SIP feature adds the ability to deploy NAT between VoIP solutions based on SIP.
NAT TFTP ALG Support	Cisco IOS XE Release 2.1	The NAT TFTP ALG Support feature supports translation of TFTP packets.
NAT VRF-Aware ALG Support	Cisco IOS XE Release 2.5	The NAT VRF-Aware ALG Support feature supports VPN routing and forwarding (VRF) for protocols that have a supported ALG.
NAT vTCP ALG Support	Cisco IOS XE Release 3.1S Cisco IOS XE Release 3.2S	The NAT vTCP ALG Support feature provides vTCP support to handle TCP segmentation and reassembling for ALG.
Support for IPsec ESP Through NAT	Cisco IOS XE Release 2.1	The Support for IPsec ESP Through NAT feature provides the ability to support multiple, concurrent IPsec ESP tunnels or connections through a NAT device configured in Overload or PAT mode.



CHAPTER 3

Carrier Grade Network Address Translation

Carrier Grade Network Address Translation (CGN) is a large-scale NAT that translates private IPv4 addresses into public IPv4 addresses. CGN employs Network Address and Port Translation methods to aggregate multiple private IPv4 addresses into fewer public IPv4 addresses.

This module provides an overview of CGN and describes how to configure CGN.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for Carrier Grade Network Address Translation

- Asymmetric routing with box-to-box (B2B) redundancy is not supported in Carrier Grade Network Address Translation (CGN) mode.
- B2B redundancy is not supported on broadband with CGN; B2B is supported on standalone CGN.
- Broadband is not supported with traditional NAT.
- CGN does not support IP sessions.
- NAT outside mappings are disabled automatically when CGN operating mode is configured using the **ip nat settings mode cgn** command.

- CGN does not support integration with Cisco Performance Routing (PfR). Commands with the **oer** keyword are not supported. For example, the **ip nat inside source route-map pool overload oer** and the **ip nat inside source list pool overload oer** commands are not supported.
- The **match-in-vrf** keyword for intra-VPN NAT is not supported with CGN.
- If you specify a destination port to configure timeout in CGN mode, the destination port is ignored and the local port is considered for timeout.

Information About Carrier Grade Network Address Translation

Carrier Grade NAT Overview

Network Address Translation (NAT) is positioned between a private and public IP network and uses nonglobal, private IP addresses and a public IP address for translation. NAT dynamically maps one or more private IP addresses into one or more public (globally routable) IP addresses that use Network Address and Port Translation (NAPT) techniques. Traditionally, NAT boxes are deployed in residential home gateways (HGWs) to translate multiple private IP addresses that are configured on multiple devices inside the home to a single public IP address that is configured and provisioned on the HGW by the service provider. Service providers deploy NAT in such a way that multiple subscribers can share a single global IP address. The service provider NAT scales to several millions of NAT translations, making it a Carrier Grade NAT (CGN).

In CGN, packets that traverse from inside the network to outside require only the source address port translation; destination address port translation is not required. CGN can be standalone like traditional NAT or you can use it along with broadband access aggregation. CGN coexits with Intelligent Services Gateway (ISG) features such as Layer 4 Redirect and subscriber services such as traffic classes.

You can configure CGN by using the **ip nat settings mode cgn** command. Use the **ip nat settings mode default** command to change to the default or traditional NAT operating mode. In the CGN mode, you cannot configure any NAT outside mappings. However, when you change from the default NAT mode to CGN mode, all existing outside mappings have to be removed. Use the **no ip nat settings support mapping outside** command to remove all outside mappings and to prevent any new outside mappings from being configured. You can also remove outside mappings by using the **no** form of commands used to configure NAT outside.

CGN increases the scalability of the number of NAT translations that can be supported because destination information is not stored.

CGN supports the following:

- All application-level gateways (ALGs) that are supported by traditional NAT. For more information about supported ALGs, see the *Using Application-Level Gateways with NAT* module of the *IP Addressing: NAT Configuration Guide*.
- Endpoint independent mapping and endpoint independent filtering.
- Hairpinning by using VRF-Aware Software Infrastructure (VASI) and policy-based routing (PBR). Hairpinning occurs when two subscribers are behind the same NAT device but can see each other only by using the global IP address.
- Interbox and intrabox redundancy.
- Lawful intercept.

- Logging of NAT high-speed logging (HSL) records. For more information about HSL, see the section "High-Speed Logging for NAT" in the *Maintaining and Monitoring NAT* module of the *IP Addressing: NAT Configuration Guide*.
- Multihoming, which is the ability to support multiple outside interfaces to provide connectivity through redundant or standby exit points. Depending on the configured routing topology, any exit interface that is marked as an outside interface can use a translation that was created previously.
- TCP timeout value of 2 hours and 4 minutes.
- VPN routing and forwarding (VRF)-aware NAT.
- CGN NAT can scale to higher number of translations on ESP200 using the **ip nat settings scale bind** command.

Carrier Grade NAT Support for Broadband Access Aggregation

You can configure Carrier Grade Network Address Translation (CGN) as an independent feature or use CGN along with broadband access aggregation.

Broadband access aggregation enables connections between multiple technologies such as cable, digital subscriber line (DSL), Ethernet, ISDN, and wireless devices that are connected to corporate VPNs, third-party applications, and the Internet.

PPP over Ethernet (PPPoE) connects hosts on a network over a simple bridging device to a remote aggregation concentrator. PPPoE is the predominant access protocol in broadband networks worldwide.

For PPPoE to work with CGN, either the virtual templates or the RADIUS server must provide the Network Address Translation (NAT) inside configuration. The NAT inside configuration can be downloaded as part of the RADIUS authentication or alternatively configure the **ip nat inside** command on the virtual template. This gets cloned into a virtual access interface that inherits the ip nat inside configuration. For the RADIUS server to provide the NAT inside configuration, configure the **aaa policy interface-config allow-subinterface** global command or configure the Cisco attribute-value pairs (AV pairs) lcp:allow-subinterface=yes and then include lcp:interface-config=ip nat inside in the RADIUS profile on a per-subscriber basis.

You can terminate a PPPoE session either in the global routing table or at a VRF instance.

CGN supports dual-stack (IPv4 and IPv6) PPP sessions. However, only IPv4 traffic is subject to NAT. The IPv6 traffic is not translated; it is routed as per the IPv6 routing configuration.

How to Configure Carrier Grade Network Address Translation

Based on your network configuration, you can configure static, dynamic, or dynamic PAT Carrier Grade NAT.

Note You must use at least one of the configurations described in the following tasks for Carrier Grade NAT to work.

Configuring Static Carrier Grade NAT

Static address translation (static NAT) allows one-to-one mapping between local and global addresses. Use the **ip nat inside source static** command to enable static NAT of the inside source address.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip nat settings mode cgn**
4. **ip nat inside source static** *local-ip global-ip*
5. **interface gigabitethernet** *card/spaslot/port.subinterface-number*
6. **ip nat inside**
7. **exit**
8. **interface** *type number*
9. **ip nat outside**
10. **end**
11. **show ip nat translations** [**verbose**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip nat settings mode cgn** **Example:** `Device(config)# ip nat settings mode cgn`	Enables CGN operating mode.
Step 4	**ip nat inside source static** *local-ip global-ip* **Example:** `Device(config)# ip nat inside source static 192.168.2.1 192.168.34.2`	Enables static Carrier Grade NAT of the inside source address.
Step 5	**interface gigabitethernet** *card/spaslot/port.subinterface-number* **Example:** `Device(config)# interface gigabitethernet 0/0/4`	Configures an interface and enters interface configuration mode. **Note** The NAT inside network can be applied to interface virtual-template when the router is used for broadband aggregation deployment.

	Command or Action	Purpose
Step 6	**ip nat inside** **Example:** `Device(config-if)# ip nat inside`	Indicates that the interface is connected to the inside network (the network that is subject to NAT translation).
Step 7	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and enters global configuration mode.
Step 8	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 0/0/0`	Configures an interface and enters interface configuration mode.
Step 9	**ip nat outside** **Example:** `Device(config-if)# ip nat outside`	Indicates that the interface is connected to the outside network.
Step 10	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and enters privileged EXEC mode.
Step 11	**show ip nat translations** [**verbose**] **Example:** `Device# show ip nat translations`	Displays active NAT translations.

Example

The following is sample output from the **show ip nat translations** command:

```
Device# show ip nat translations

Pro  Inside global         Inside local          Outside local         Outside global
udp  10.5.5.1:1025          192.0.2.1:4000        ---                    ---
udp  10.5.5.1:1024          192.0.2.3:4000        ---                    ---
udp  10.5.5.1:1026          192.0.2.2:4000        ---                    ---

Total number of translations: 3
```

The following is sample output from the **show ip nat translations verbose** command:

```
Device# show ip nat translations verbose

Pro  Inside global         Inside local          Outside local         Outside global
udp  10.5.5.1:1025          192.0.2.1:4000        ---                    ---
  create: 02/15/12 11:38:01, use: 02/15/12 11:39:02, timeout: 00:00:00
  Map-Id(In): 1
  Mac-Address: 0000.0000.0000    Input-IDB: TenGigabitEthernet1/1/0
  entry-id: 0x0, use_count:1

udp  10.5.5.1:1024          192.0.2.3:4000        ---                    ---
  create: 02/15/12 11:38:00, use: 02/15/12 11:39:02, timeout: 00:00:00
  Map-Id(In): 1
```

```
  Mac-Address: 0000.0000.0000    Input-IDB: TenGigabitEthernet1/1/0
  entry-id: 0x0, use_count:1

udp  10.5.5.1:1026          192.0.2.2:4000        ---                   ---
  create: 02/15/12 11:38:00, use: 02/15/12 11:39:02, timeout: 00:00:00
  Map-Id(In): 1
  Mac-Address: 0000.0000.0000    Input-IDB: TenGigabitEthernet1/1/0
  entry-id: 0x0, use_count:1

Total number of translations: 3
```

Configuring Dynamic Carrier Grade NAT

Dynamic address translation (dynamic NAT) maps unregistered IP addresses to registered IP addresses from a pool of registered IP addresses.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip nat settings mode cgn**
4. **access-list** *standard-access-list-number* **permit** *source wildcard*
5. **access-list** *standard-access-list-number* **permit** *source wildcard*
6. **route-map** *map-tag*
7. **match ip address** [*access-list-number*]
8. **match ip next-hop** [*access-list-number*]
9. **exit**
10. **ip nat pool** *name start-ip end-ip* **prefix-length** *prefix-length*
11. **ip nat inside source route-map** *name* **pool** *name*
12. **interface gigabitethernet** *card/spaslot/port.subinterface-number*
13. **ip nat inside**
14. **exit**
15. **interface** *type number*
16. **ip nat outside**
17. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**ip nat settings mode cgn** **Example:** `Device(config)# ip nat settings mode cgn`	Enables CGN operating mode.
Step 4	**access-list** *standard-access-list-number* **permit** *source wildcard* **Example:** `Device(config)# access-list 1 permit 10.1.1.0 0.0.0.255`	Defines a standard access list and specifies a host. • Access list 1 defined in this step is used by the **match ip address** command.
Step 5	**access-list** *standard-access-list-number* **permit** *source wildcard* **Example:** `Device(config)# access-list 2 permit 10.5.5.0 0.0.0.255`	Defines a standard access list and specifies a host. • Access list 2 defined in this step is used by the **match ip next-hop** command.
Step 6	**route-map** *map-tag* **Example:** `Device(config)# route-map nat-route-map`	Defines conditions for redistributing routes from one routing protocol into another or enables policy routing and enters route-map configuration mode.
Step 7	**match ip address** [*access-list-number*] **Example:** `Device(config-route-map)# match ip address 1`	Distributes any routes that have a destination network number address that is permitted by a standard access list, an extended access list, or a prefix list or performs policy routing on packets.
Step 8	**match ip next-hop** [*access-list-number*] **Example:** `Device(config-route-map)# match ip next-hop 2`	Redistributes any routes that have a next-hop router address passed by one of the specified access lists.
Step 9	**exit** **Example:** `Device(config-route-map)# exit`	Exits route-map configuration mode and enters global configuration mode.
Step 10	**ip nat pool** *name start-ip end-ip* **prefix-length** *prefix-length* **Example:** `Device(config)# ip nat pool nat-pool 10.1.1.1 10.1.254.254 prefix-length 16`	Defines a pool of IP addresses for NAT.
Step 11	**ip nat inside source route-map** *name* **pool** *name* **Example:** `Device(config)# ip nat inside source route-map nat-route-map pool nat-pool`	Enables dynamic NAT of the inside source address.
Step 12	**interface gigabitethernet** *card/spaslot/port.subinterface-number*	Configures an interface and enters interface configuration mode.

	Command or Action	Purpose
	Example: `Device(config)# interface gigabitethernet 0/0/5`	**Note** The NAT inside network can be applied to interface virtual-template when the router is used for broadband aggregation deployment.
Step 13	**ip nat inside** **Example:** `Device(config-if)# ip nat inside`	Indicates that the interface is connected to the inside network (the network that is subject to NAT translation).
Step 14	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and enters global configuration mode.
Step 15	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 0/0/1`	Configures an interface and enters interface configuration mode.
Step 16	**ip nat outside** **Example:** `Device(config-if)# ip nat outside`	Indicates that the interface is connected to the outside network.
Step 17	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and enters privileged EXEC mode.

Configuring Dynamic Port Address Carrier Grade NAT

Port Address Translation (PAT) or overloading is a form of dynamic NAT that maps multiple unregistered IP addresses to a single registered IP address (many-to-one mapping) by using different ports. PAT enables thousands of users to connect to the Internet by using only one real global IP address.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip nat settings mode cgn**
4. **ip nat inside source list** *number* **pool** *name* [**overload**]
5. **ip nat pool** *name start-ip end-ip* **netmask** *netmask*
6. **access-list** *standard-access-list-number* **permit** *source wildcard*
7. **interface gigabitethernet** *card/spaslot/port.subinterface-number*
8. **ip nat inside**
9. **exit**
10. **interface** *type number*
11. **ip nat outside**
12. **end**

13. **show ip nat statistics**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip nat settings mode cgn** **Example:** `Device(config)# ip nat settings mode cgn`	Enables CGN operating mode.
Step 4	**ip nat inside source list** *number* **pool** *name* [**overload**] **Example:** `Device(config)# ip nat inside source list 1 pool nat-pool overload`	Enables the router to use one global address for many local addresses. • When you configure the **overload** keyword, the TCP or UDP port number of each inside host distinguishes between multiple conversations using the same local IP address. • The **overload** keyword configures overloading or PAT.
Step 5	**ip nat pool** *name start-ip end-ip* **netmask** *netmask* **Example:** `Device(config)# ip nat pool nat-pool 10.1.1.1 10.1.254.254 netmask 255.255.0.0`	Defines a pool of IP addresses for NAT.
Step 6	**access-list** *standard-access-list-number* **permit** *source wildcard* **Example:** `Device(config)# access-list 1 permit 172.16.0.0 255.255.0.0`	Defines a standard access list and specifies a host.
Step 7	**interface gigabitethernet** *card/spaslot/port.subinterface-number* **Example:** `Device(config)# interface gigabitethernet 0/0/6`	Configures an interface and enters interface configuration mode. **Note** The NAT inside network can be applied to interface virtual-template when the router is used for broadband aggregation deployment.
Step 8	**ip nat inside** **Example:** `Device(config-if)# ip nat inside`	Indicates that the interface is connected to the inside network (the network that is subject to NAT translation).

	Command or Action	Purpose
Step 9	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and enters global configuration mode.
Step 10	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 0/0/2`	Configures an interface and enters interface configuration mode.
Step 11	**ip nat outside** **Example:** `Device(config-if)# ip nat outside`	Indicates that the interface is connected to the outside network.
Step 12	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and enters privileged EXEC mode.
Step 13	**show ip nat statistics** **Example:** `Device# show ip nat statistics`	Displays NAT statistics.

Example

The following is sample output from the **show ip nat statistics** command:

```
Device# show ip nat statistics

Total active translations: 3 (0 static, 3 dynamic; 3 extended)
Outside interfaces:
  TenGigabitEthernet2/0/0, TenGigabitEthernet2/1/0, TenGigabitEthernet2/2/0
  TenGigabitEthernet2/3/0
Inside interfaces:
  TenGigabitEthernet1/0/0, TenGigabitEthernet1/1/0, TenGigabitEthernet1/2/0
  TenGigabitEthernet1/3/0
Hits: 59230465  Misses: 3
CEF Translated packets: 0, CEF Punted packets: 0
Expired translations: 0
Dynamic mappings:
-- Inside Source
[Id: 1] access-list 102 pool mypool refcount 3
 pool mypool: netmask 255.255.255.0
        start 10.5.5.1 end 10.5.5.5
        type generic, total addresses 5, allocated 1 (20%), misses 0
nat-limit statistics:
 max entry: max allowed 2147483647, used 3, missed 0
Pool stats drop: 0  Mapping stats drop: 0
Port block alloc fail: 0
IP alias add fail: 0
Limit entry add fail: 0
```

Configuration Examples for Carrier Grade Network Address Translation

Example: Configuring Static Carrier Grade NAT

```
Device# configure terminal
Device(config)# ip nat settings mode cgn
Device(config)# ip nat inside source static 192.168.2.1 192.168.34.2
Device(config)# interface gigabitethernet 0/0/6
Device(config-if)# ip nat inside
Device(config-if)# exit
Device(config)# interface gigabitethernet 0/0/0
Device(config-if)# ip nat outside
Device(config-if)# end
```

Example: Configuring Dynamic Carrier Grade NAT

```
Device# configure terminal
Device(config)# ip nat settings mode cgn
Device(config)# access-list 1 permit 10.1.1.0 0.0.0.255
Device(config)# access-list 2 permit 10.5.5.0 0.0.0.255
Device(config)# route-map nat-route-map
Device(config-route-map)# match ip address 1
Device(config-route-map)# match ip next-hop 2
Device(config-route-map)# exit
Device(config)# ip nat pool nat-pool 10.1.1.1 10.1.254.254 prefix-length 16
Device(config)# ip nat inside source route-map nat-route-map pool nat-pool
Device(config)# interface gigabitethernet 0/0/5
Device(config-if)# ip nat inside
Device(config-if)# exit
Device(config)# interface gigabitethernet 0/0/1
Device(config-if)# ip nat outside
Device(config-if)# end
```

Example: Configuring Dynamic Port Address Carrier Grade NAT

```
Device# configure terminal
Device(config)# ip nat settings mode cgn
Device(config)# ip nat inside source list 1 pool nat-pool overload
Device(config)# ip nat pool nat-pool 10.1.1.1 10.1.254.254 netmask 255.255.0.0
Device(config)# access-list 1 permit 172.16.0.0 255.255.0.0
Device(config)# interface gigabitethernet 0/0/4
Device(config-if)# ip nat inside
Device(config-if)# exit
Device(config)# interface gigabitethernet 0/0/3
Device(config-if)# ip nat outside
Device(config-if)# end
```

Additional References for Carrier Grade Network Address Translation

Related Documents

Related Topic	Document Title
Cisco IOS commands	Master Command List, All Releases
NAT commands	IP Addressing Command Reference
NAT ALGs	"Using Application-Level Gateways with NAT"
HSL messages	"Monitoring and Maintaining NAT"

Standards and RFCs

Standard/RFC	Title
RFC 4787	*Network Address Translation (NAT) Behavioral Requirements for Unicast UDP*
RFC 5582	*Location-to-URL Mapping Architecture and Framework*

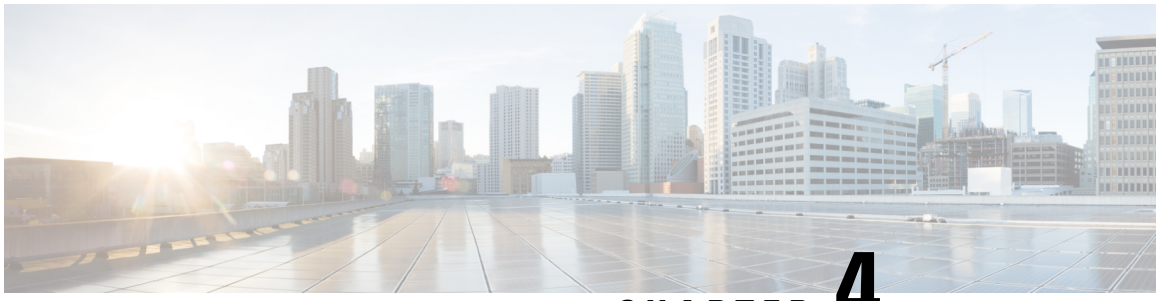
Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/support

Feature Information for Carrier Grade Network Address Translation

Table 4: Feature Information for Carrier Grade Network Address Translation

Feature Name	Releases	Feature Information
Carrier Grade Network Address Translation	Cisco IOS XE Release 3.6S	Carrier Grade Network Address Translation (CGN) is a large-scale NAT that translates private IPv4 addresses into public IPv4 addresses. CGN employs Network Address and Port Translation methods to aggregate multiple private IPv4 addresses into fewer public IPv4 addresses. The following commands were introduced or modified: **ip nat settings mode** and **ip nat settings support mapping outside**. **Note** This feature is not supported on ISR 4000 platform.



CHAPTER 4

Static NAT Mapping with HSRP

This module contains procedures for configuring Network Address Translation (NAT) to support the increasing need for highly resilient IP networks. This network resiliency is required where application connectivity needs to continue unaffected by failures to links and routers at the NAT border.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Static NAT Mapping with HSRP

To understand how high availability is implemented on Cisco ASR 1000 Series Aggregation Services Routers, see the "High Availability Overview" module in the *Cisco ASR 1000 Series Aggregation Services Routers Software Configuration Guide*.

Restrictions for Static NAT Mapping with HSRP

- Using any IP address configured on a device IP address as an address pool or in a NAT static rule is not supported. NAT can share the physical interface address (not any other IP address) of a device only by

using the NAT interface overload configuration. A device uses the ports of its physical interface and NAT must receive communication about the ports that it can safely use for translation. This communication happens only when the NAT interface overload is configured.

- Virtual routing and forwarding (VRF) NAT with Hot Standby Router Protocol (HSRP) is not supported. Effective with Cisco IOS XE Denali 16.3.3, this restriction is not applicable. Upgrade to this release if you want your device to support VRF NAT with HSRP.
- Static NAT mappings must be mirrored on two or more HSRP devices, because the NAT state is not exchanged between devices running NAT in an HSRP group.
- If you configure both HSRP devices with the same static NAT and the **hsrp** keyword to link these devices to the same HSRP group is not configured, the behavior of the devices will be unpredictable.

Information About Static NAT Mapping with HSRP

Static Mapping Support with HSRP for High Availability Feature Overview

When an Address Resolution Protocol (ARP) query is triggered for an address that is configured with NAT static mapping and owned by the device, NAT responds with the burned in MAC (BIA MAC) address on the interface to which the ARP is pointing. Two devices act as the Hot Standby Router Protocol (HSRP) active and standby. You must enable and configure the NAT inside interfaces of the active and standby devices to belong to a group.

Address Resolution with ARP

A device in IP can have both a local address (which uniquely identifies the device on its local segment or LAN) and a network address (which identifies the network to which the device belongs). The local address is known as a data link address because it is contained in the data link layer (Layer 2 of the OSI model) part of the packet header and is read by data-link devices such as bridges, all device interfaces and so on. The local address is referred to as the MAC address, because the MAC sublayer within the data-link layer processes addresses for the layer.

To communicate with a device on Ethernet, for example, the Cisco IOS software must first determine the 48-bit MAC or local data-link address of that device. The process of determining the local data-link address from an IP address is called address resolution. The process of determining the IP address from a local data-link address is called reverse address resolution.

The software uses three forms of address resolution: Address Resolution Protocol (ARP), proxy ARP, and Probe (similar to ARP). The software also uses the Reverse Address Resolution Protocol (RARP). ARP, proxy ARP, and RARP are defined in RFCs 826, 1027, and 903, respectively. Probe is a protocol developed by the Hewlett-Packard Company (HP) for use on IEEE-802.3 networks.

ARP is used to associate IP addresses with media or MAC addresses. Taking an IP address as input, ARP determines the associated media address. Once a media or MAC address is determined, the IP address or media address association is stored in an ARP cache for rapid retrieval. Then the IP datagram is encapsulated in a link-layer frame and sent over the network. Encapsulation of IP datagrams and ARP requests and replies on IEEE 802 networks other than Ethernet is specified by the Subnetwork Access Protocol (SNAP).

Gratuitous ARP

When a host sends an ARP request to resolve its own IP address, it is called gratuitous ARP. In the ARP request packet, the source and destination IP addresses are filled with the same source IP address itself. The destination MAC address is the Ethernet broadcast address.

When a router becomes active, it broadcasts a gratuitous ARP packet with the Hot Standby Router Protocol (HSRP) virtual MAC address to the affected LAN segment. If the segment uses an Ethernet switch, this allows the switch to change the location of the virtual MAC address so that packets flow to the new router instead of the one that is no longer active. End devices do not actually need gratuitous ARP if routers use the default HSRP MAC address.

How to Configure Static NAT Mapping with HSRP

Configuring NAT Static Mapping Support for HSRP

When an Address Resolution Protocol (ARP) query is triggered for an address that is configured with NAT static mapping and owned by the router, NAT responds with the burned in MAC (BIA MAC) address on the interface to which the ARP is pointing. Two routers are acting as HSRP active and standby. Their NAT inside interfaces must be enabled and configured to belong to a group.

Benefits of Configuring Static Mapping Support for HSRP are the following:

- Using static mapping support for HSRP, failover is ensured without having to time out and repopulate upstream ARP caches in a high-availability environment, where HSRP router pairs have identical NAT configuration for redundancy.
- Static mapping support for HSRP allows the option of having only the HSRP active router respond to an incoming ARP for a router configured with a NAT address.

Both of the following tasks are required and must be performed on both the active and standby routers to configure NAT static mapping support for HSRP:

Enabling HSRP on the NAT Interface

Perform this task to enable HSRP on the NAT interface of both the active and standby routers.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ip address** *ip-address mask*
5. **no ip redirects**
6. **ip nat** {**inside** | **outside**}
7. **standby** [*group-number*] **ip** [*ip-address* [**secondary**]]
8. **standby** [*group-number*] **preempt**
9. **standby** [*group-number*] **ip** [*ip-address* | **secondary**]
10. **standby** [*group-number*] **name** [*group-name*]
11. **standby** [*group-number*] **track** *interface-number*

12. **end**
13. **show standby**
14. **show ip nat translations** [**verbose**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables higher privilege levels, such as privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 1/1/1`	Configures an interface and enters interface configuration mode.
Step 4	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 192.168.1.27 255.255.255.0`	Sets the primary IP address on the interface.
Step 5	**no ip redirects** **Example:** `Device(config-if)# no ip redirects`	Disables the sending of redirect messages
Step 6	**ip nat** {**inside** | **outside**} **Example:** `Device(config)# ip nat inside`	Connects the interface to the inside network.
Step 7	**standby** [*group-number*] **ip** [*ip-address* [**secondary**]] **Example:** `Device(config-if)# standby 10 priority 105`	Enables the HSRP protocol.
Step 8	**standby** [*group-number*] **preempt** **Example:** `Device(config-if)# standby 10 preempt`	Configures HSRP preemption.
Step 9	**standby** [*group-number*] **ip** [*ip-address* | **secondary**] **Example:** `Device(config-if)# standby 10 ip 192.168.5.30`	Enables the HSRP protocol.
Step 10	**standby** [*group-number*] **name** [*group-name*] **Example:**	Sets the HSRP group name.

	Command or Action	Purpose
	`Device(config-if)# standby 10 name HSRP1`	
Step 11	**standby** [*group-number*] **track** *interface-number* **Example:** `Device(config-if)# standby 10 track gigabitethernet1/1/1`	Configures HSRP to track an object and to change the hot standby priority on the basis of the state of the object.
Step 12	**end** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to privileged EXEC mode.
Step 13	**show standby** **Example:** `Device# show standby`	(Optional) Displays HSRP information
Step 14	**show ip nat translations** [**verbose**] **Example:** `Device# show ip nat translations verbose`	(Optional) Displays active NAT translations.

Enabling Static NAT for HSRP

Before you begin

To enable static mapping support with HRSP for high availability, perform this task on both the active and standby devices.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip nat inside source static** *local-ip global-ip* **redundancy** *group-name*
4. **ip classless**
5. **ip route prefix mask** *interface-type interface-number*
6. **no ip http server**
7. **end**
8. **show ip nat translations** [**verbose**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode. • Enter your password if prompted.
Step 3	**ip nat inside source static** *local-ip global-ip* **redundancy** *group-name* **Example:** `Device(config)# ip nat inside source static 192.168.5.33 10.10.10.5 redundancy HSRP1`	Enables a device to respond to Address Resolution Protocol (ARP) queries using BIA MAC, if HSRP is configured on the NAT inside interface.
Step 4	**ip classless** **Example:** `Device(config)# ip classless`	Enables a device to forward packets that are destined for a subnet of a network that has no network default route, to the best supernet route possible.
Step 5	**ip route prefix mask** *interface-type interface-number* **Example:** `Device(config)# ip route 10.10.10.0 255.255.255.0 gigabitethernet 0/0/0`	Establishes static routes.
Step 6	**no ip http server** **Example:** `Device(config)# no ip http server`	Enables the HTTP server on your IP system.
Step 7	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.
Step 8	**show ip nat translations** [**verbose**] **Example:** `Device# show ip nat translations verbose`	(Optional) Displays active NAT translations.

Configuration Example for Static NAT Mapping with HSRP

Example: Configuring Static NAT in an HSRP Environment

The following example shows support for NAT with a static configuration in an HSRP environment. Two devices act as HSRP active and standby, and the NAT inside interfaces are HSRP enabled and configured to belong to group HSRP1.

Active Device Configuration

```
interface BVI10
 ip address 192.168.5.54 255.255.255.255.0
 no ip redirects
```

```
 ip nat inside
 standby 10 priority 105 preempt
 standby 10 name HSRP1
 standby 10 ip 192.168.5.30
 standby 10 track gigabitethernet1/1/1
!
!
 ip default-gateway 10.0.18.126
 ip nat inside source static 192.168.5.33 10.10.10.5 redundancy HSRP1
 ip classless
 ip route 10.10.10.0 255.255.255.0 gigabitethernet1/1/1
 ip route 172.22.33.0 255.255.255.0 gigabitethernet1/1/1
 no ip http server
```

Standby Device Configuration

```
interface BVI10
 ip address 192.168.5.56 255.255.255.255.0
 no ip redirects
 ip nat inside
 standby 10 priority 100 preempt
 standby 10 name HSRP1
 standby 10 ip 192.168.5.30
 standby 10 track gigabitethernet0/0/1
!
 ip default-gateway 10.0.18.126
 ip nat inside source static 192.168.5.33 3.3.3.5 redundancy HSRP1
 ip classless
 ip route 10.0.32.231 255.255.255 gigabitethernet0/0/1
 ip route 10.10.10.0 255.255.255.0 gigabitethernet0/0/1
 no ip http server
```

Additional References for Static NAT Mapping with HSRP

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
NAT commands: complete command syntax, command mode, command history, usage guidelines, and examples	*Cisco IOS IP Addressing Services Command Reference*
IP Access List Sequence Numbering	*IP Access List Sequence Numbering* document
NAT configuration tasks	“Configuring NAT for IP Address Conservation” module
NAT maintenance	“Monitoring and Maintaining NAT” module
Using NAT with MPLS VPNs	“Integrating NAT with MPLS VPNs” module

Standards and RFCs

Standard/RFC	Title
RFC 903	*Reverse Address Resolution Protocol*
RFC 826	*Ethernet Address Resolution Protocol: Or converting network protocol addresses to 48.bit Ethernet address for transmission on Ethernet hardware*
RFC 1027	*Using ARP to implement transparent subnet gateways*

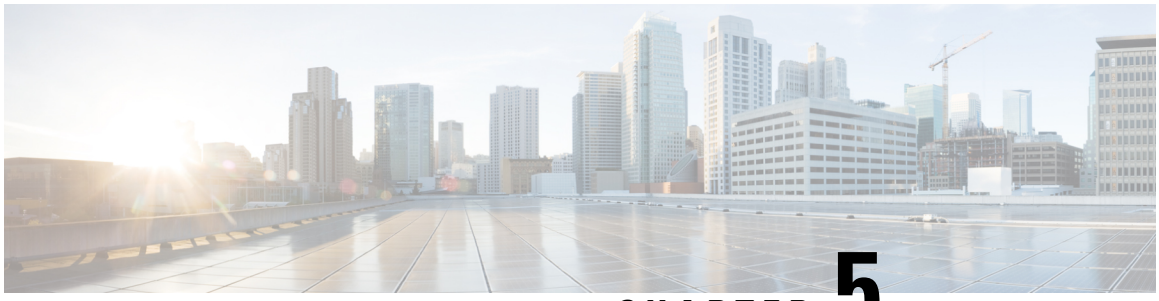
Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Static NAT Mapping with HSRP

Table 5: Feature Information for Static NAT Mapping with HSRP

Feature Name	Releases	Feature Configuration Information
NAT—Static Mapping Support with HSRP for High Availability	Cisco IOS XE Release 2.1	Static mapping support for HSRP allows the option of having only the HSRP active router respond to an incoming ARP for a router configured with a NAT address.



CHAPTER 5

VRF-Aware Dynamic NAT Mapping with HSRP

The VRF-Aware Dynamic NAT Mapping with HSRP feature supports stateless redundancy using HSRP with dynamic Network Address Translation (NAT), Port Address Translation (PAT), and interface overload configuration. Dynamic NAT, PAT and interface overload support HSRP with and without virtual routing and forwarding (VRF) instances. All these configurations are supported in the Carrier Grade NAT (CGN) mode.

This module describes the feature and explains how to configure it.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for VRF-Aware Dynamic NAT Mapping with HSRP

- Both the active and standby devices must be configured with the same Network Address Translation (NAT) rules.
- Hot Standby Router Protocol (HSRP) must be configured between the active and standby devices.

Restrictions for VRF-Aware Dynamic NAT Mapping with HSRP

- During failovers, NAT translated IP addresses on devices may be different from the IP address before the failover, because no state information is exchanged between active and standby devices.
- During a failover, all existing NAT sessions are destroyed and new sessions are established in the active device.
- HSRP Virtual IP Address (VIP) cannot be used by NAT pools.
- Active/active configuration is not supported; only active/standby configuration is supported.
- IPv6 is not supported; only IPv4 is supported.

Information About VRF-Aware Dynamic NAT Mapping with HSRP

VRF-Aware Dynamic NAT Mapping with HSRP Overview

The VRF-Aware Dynamic NAT Mapping with HSRP feature supports stateless redundancy using HSRP with dynamic Network Address Translation (NAT), Port Address Translation (PAT), and interface overload configuration. Dynamic NAT, PAT and interface overload support HSRP with and without virtual routing and forwarding (VRF) instances. All these configurations are supported in the Carrier Grade NAT (CGN) mode.

Hot Standby Router Protocol (HSRP) provides high network availability by providing first-hop routing redundancy for IP hosts on networks configured with a default gateway IP address. HSRP is used in a group of routers for selecting an active device and a standby device. HSRP provides redundancy for routing IP traffic without being dependent on the availability of a single router. In a group of device interfaces, the active device is the device of choice for routing packets; the standby device is the device that takes over when the active device fails or when preset conditions are met.

Devices running HSRP send and receive multicast UDP-based hello packets to detect router failure and to designate active and standby devices. Selection of active and standby devices is based on the assigned priority. The device with the highest priority is selected as the active device. After failover, a new active device sends a gratuitous Address Resolution Protocol (ARP) request to LAN users to notify about the change in MAC address for the virtual IP address (VIP).

To enable this feature, both the active and standby devices must be configured with the same NAT rules and HSRP must be configured on both the devices. Based on the configured priority one of the devices will be active and the other standby. This feature supports VRF-aware NAT translation and Carrier Grade NAT (CGN) mode.

This feature supports the LAN-LAN topology as well as the LAN-WAN topology. In the LAN-WAN topology, only symmetric routing is supported.

When an Address Resolution Protocol (ARP) query is triggered for an address that is configured with dynamic NAT mapping and owned by the device, NAT responds with the burned-in MAC (BIA MAC) address on the interface to which the ARP is pointing. You must enable and configure the NAT inside interfaces of the active and standby devices to belong to a group.

In Cisco IOS XE Denali 16.3 release, the Allow same ACL/router-map on multiple NAT statements feature was introduced to support usage of same ACL for configuring both dynamic mapping and static mapping in NAT. Dynamic mapping is given the precedence over static mapping regardless of the configuration order. The precedence of dynamic mapping over static mapping using the sequence number of the class ensures class order consistency in NAT.

Address Resolution with ARP

A device in IP can have both a local address (which uniquely identifies the device on its local segment or LAN) and a network address (which identifies the network to which the device belongs). The local address is known as a data link address because it is contained in the data link layer (Layer 2 of the OSI model) part of the packet header and is read by data-link devices such as bridges, all device interfaces and so on. The local address is referred to as the MAC address, because the MAC sublayer within the data-link layer processes addresses for the layer.

To communicate with a device on Ethernet, for example, the Cisco IOS software must first determine the 48-bit MAC or local data-link address of that device. The process of determining the local data-link address from an IP address is called address resolution. The process of determining the IP address from a local data-link address is called reverse address resolution.

The software uses three forms of address resolution: Address Resolution Protocol (ARP), proxy ARP, and Probe (similar to ARP). The software also uses the Reverse Address Resolution Protocol (RARP). ARP, proxy ARP, and RARP are defined in RFCs 826, 1027, and 903, respectively. Probe is a protocol developed by the Hewlett-Packard Company (HP) for use on IEEE-802.3 networks.

ARP is used to associate IP addresses with media or MAC addresses. Taking an IP address as input, ARP determines the associated media address. Once a media or MAC address is determined, the IP address or media address association is stored in an ARP cache for rapid retrieval. Then the IP datagram is encapsulated in a link-layer frame and sent over the network. Encapsulation of IP datagrams and ARP requests and replies on IEEE 802 networks other than Ethernet is specified by the Subnetwork Access Protocol (SNAP).

Gratuitous ARP

When a host sends an ARP request to resolve its own IP address, it is called gratuitous ARP. In the ARP request packet, the source and destination IP addresses are filled with the same source IP address itself. The destination MAC address is the Ethernet broadcast address.

When a router becomes active, it broadcasts a gratuitous ARP packet with the Hot Standby Router Protocol (HSRP) virtual MAC address to the affected LAN segment. If the segment uses an Ethernet switch, this allows the switch to change the location of the virtual MAC address so that packets flow to the new router instead of the one that is no longer active. End devices do not actually need gratuitous ARP if routers use the default HSRP MAC address.

How to Configure VRF-Aware Dynamic NAT Mapping with HSRP

Enabling HSRP for VRF-Aware Dynamic NAT

SUMMARY STEPS

1. **enable**

2. **configure terminal**
3. **track** *object-number* **interface** *type number* {**ip** | **ipv6** | **line-protocol**}
4. **exit**
5. **interface** *type number*
6. **ip nat inside**
7. **ip address** *ip-address mask*
8. **standby** *group-number* **ip** [*ip-address*]
9. **standby use-bia**
10. **standby** *group-number* **priority** *priority*
11. **standby** *group-number* **preempt** [*delay*]
12. **standby** *group-number* **track** *object-number* [**decrement** *priority-decrement*]
13. **exit**
14. **ip nat pool** *pool-name start-ipend-ip* **netmask** *netmask*
15. **access-list** *standard-access-list* **permit** *ip-address mask*
16. **ip nat inside source list** *list-name* **pool** *pool-name* [**overload**]
17. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**track** *object-number* **interface** *type number* {**ip** | **ipv6** | **line-protocol**} **Example:** `Device(config)# track 10 interface gigabitethernet 0/0/0 line-protocol`	Configures an interface to be tracked where the Gateway Load Balancing Protocol (GLBP) weighting changes based on the state of the interface
Step 4	**exit** **Example:** `Device(config-track)# exit`	Exits tracking configuration mode and returns to global configuration mode.
Step 5	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 1/2/1`	Configures an interface and enters interface configuration mode.
Step 6	**ip nat inside** **Example:** `Device(config-f)# ip nat inside`	Connects the interface to the inside network, which is subject to Network Address Translation (NAT).

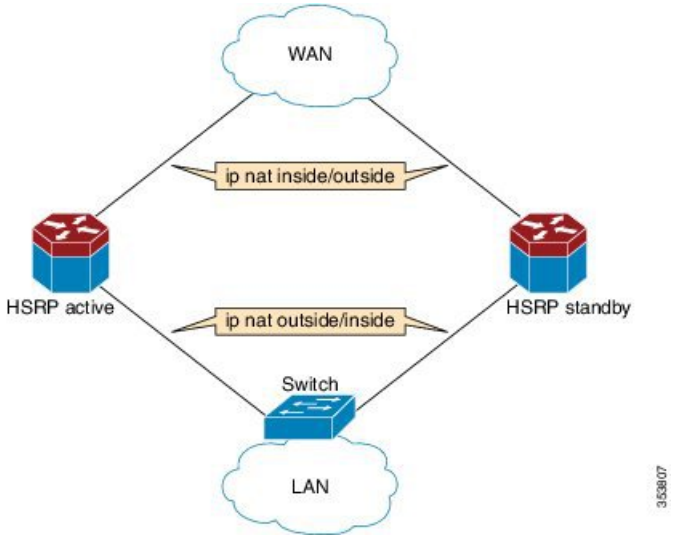
	Command or Action	Purpose
Step 7	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 192.0.0.2 255.0.0.0`	Sets a primary or secondary IP address for an interface.
Step 8	**standby** *group-number* **ip** [*ip-address*] **Example:** `Device(config-if)# standby 1 ip 192.0.0.1`	Activates the Hot Standby Router Protocol (HSRP).
Step 9	**standby use-bia** **Example:** `Device(config-if)# standby use-bia`	Configures HSRP to use the burned-in address of the interface as its virtual MAC address, instead of the preassigned MAC address or the functional address.
Step 10	**standby** *group-number* **priority** *priority* **Example:** `Device(config-if)# standby 1 priority 120`	Configures the HSRP priority. • The priority range is from 1 to 255, where 1 denotes the lowest priority and 255 denotes the highest priority. The default priority value is 100. The device in the HSRP group with the highest priority value becomes the active device.
Step 11	**standby** *group-number* **preempt** [*delay*] **Example:** `Device(config-if)# standby 1 preempt`	Configures HSRP preemption and preemption delay. • If you configure this command, when a local device has an HSRP priority higher than the current active device, the local device assumes control as the active device. If preemption is not configured, the local device assumes control as the active device only if it receives information indicating no device is in the active state (acting as the designated device).
Step 12	**standby** *group-number* **track** *object-number* [**decrement** *priority-decrement*] **Example:** `Device(config-if)# standby 1 track 10 decrement 15`	Configure HSRP to track an object, and change the HSRP priority on the basis of the state of the object. • When a tracked object goes down, the HSRP priority decreases by 10. If an object is not tracked, state changes do not affect the priority.
Step 13	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 14	**ip nat pool** *pool-name start-ipend-ip* **netmask** *netmask* **Example:** `Device(config)# ip nat pool pool1 10.1.1.10 10.1.1.52 netmask 255.255.255.0`	Defines a pool of IP addresses for Network Address Translation (NAT) translations.
Step 15	**access-list** *standard-access-list* **permit** *ip-address mask* **Example:**	

	Command or Action	Purpose
	`Device(config)# acces-list 1 permit 190.0.0.0 0.255.255.255`	
Step 16	**ip nat inside source list** *list-name* **pool** *pool-name* [**overload**] **Example:** `Device(config)# ip nat inside source list list1 pool pool1 overload`	Enables NAT of the inside source address. • When overloading is configured, it enables the device to use one global address for many local addresses. The TCP or UDP port number of each inside host distinguishes between the multiple conversations using the same local IP address.
Step 17	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Configuration Examples for VRF-Aware Dynamic NAT Mapping with HSRP

Example: Enabling HSRP for VRF-Aware Dynamic NAT

Figure 5: HSRP NAT LAN-WAN Topology



The following example shows a LAN-WAN configuration for dynamic Network Address Translation (NAT) overload mapping with Hot Standby Router Protocol (HSRP). A virtual routing and forwarding (VRF) instance is enabled for this configuration. Devices that are configured with NAT do not have any route configurations related to HSRP Virtual IP Address (VIP). LAN users using static routes have to set the default route or next-hop to the HSRP VIP; for example configure the **ip route 0.0.0.0 0.0.0.0 192.0.2.1** command.

```
! Active device configuration:
Device# configure terminal
Device(config)# vrf definition vrf1
Device(config-vrf)# exit
Device(config)# track 10 interface fastethernet 1/1/1 line-protocol
Device(config-track)# exit
Device(config)# interface fastethernet 1/1/0
Device(config-if)# vrf forwarding vrf1
Device(config-if)# ip nat inside
Device(config-if)# ip address 192.0.2.2 255.255.255.240
Device(config-if)# standby 1 ip 192.0.2.1
Device(config-if)# standby use-bia
Device(config-if)# standby 1 priority 120
Device(config-if)# standby 1 preempt
Device(config-if)# standby 1 track 10 decrement 15
Device(config-if)# exit
Device(config)# interface fastethernet 1/1/1
Device(config-if)# ip address 198.51.100.1 255.255.255.240
Device(config-if)# ip nat outside
Device(config-if)# exit
Device(config)# ip nat pool pool1 10.1.1.1 10.1.1.255 netmask 255.255.255.0
Device(config)# access-list 1 permit 203.0.113.0 255.255.255.240
Device(config)# ip nat inside source list1 pool pool1 vrf vrf1 overload
Device(config)# end
```

```
! Standby device configuration:
Device# configure terminal
Device(config)# vrf definition vrf1
Device(config-vrf)# exit
Device#(config)# interface fastethernet 1/2/0
Device(config-if)# vrf forwarding vrf1
Device(config-if)# ip nat inside
Device(config-if)# ip address 192.0.2.3 255.255.255.240
Device(config-if)# standby 1 ip 192.0.2.1
Device(config-if)# standby use-bia
Device(config-if)# standby 1 priority 110
Device(config-if)# standby 1 preempt
Device(config-if)# exit
Device(config)# interface fastethernet 1/2/1
Device(config-if)# ip address 172.16.0.1 255.255.224.0
Device(config-if)# ip nat outside
Device(config-if)# exit
Device(config)# ip nat pool pool1 10.1.1.1 10.1.1.255 netmask 255.255.255.0
Device(config)# access-list 1 permit 2013.0.113.0 255.255.255.240
Device(config)# ip nat inside source list1 pool pool1 vrf vrf1 overload
Device(config)# end
```

Verifying HSRP for VRF-Aware Dynamic NAT

Before you begin

-

SUMMARY STEPS

1. enable

2. **show arp**
3. **show ip alias**
4. **show ip nat translations**
5. **show standby brief**

DETAILED STEPS

Step 1 enable

Example:

```
Device> enable
```

enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 show arp

Displays the entries in the Address Resolution Protocol (ARP) table.

Example:

```
Device# show arp

Protocol  Address          Age (min)  Hardware Addr   Type   Interface
Internet  192.0.0.1              -   0023.eb85.7650  ARPA   GigabitEthernet1/1/0
Internet  192.0.0.2              -   0023.eb85.7650  ARPA   GigabitEthernet1/1/0
```

Step 3 show ip alias

Displays the IP addresses that are mapped to TCP ports (aliases) and Serial Line Internet Protocol (SLIP) addresses, which are treated similar to aliases.

Example:

```
Device# show ip alias

Address Type             IP Address      Port
Interface                10.39.21.3
Dynamic                  192.0.0.1
Interface                192.0.0.2
```

Step 4 show ip nat translations

Displays active Network Address Translation (NAT) translations.

Example:

```
Device# show ip nat translations

Pro  Inside global            Inside local          Outside local         Outside global
udp  10.1.1.4:512             190.0.0.1:435         193.0.0.1:80          193.0.0.1:80
udp  10.1.1.4:515             190.0.0.5:435         193.0.0.1:80          193.0.0.1:80
udp  10.1.1.4:514             190.0.0.4:435         193.0.0.1:80          193.0.0.1:80
udp  10.1.1.4:518             190.0.0.3:435         193.0.0.1:80          193.0.0.1:80
```

Step 5 show standby brief

Displays Hot Standby Router Protocol (HSRP) information in a single line of output for each standby group.

Example:

```
Device# show standby brief

                     P indicates configured to preempt.
                     |
Interface   Grp  Pri P State   Active          Standby         Virtual IP
Ga1/1/0     1    120 P Active  local           192.0.0.3       192.0.0.1
```

Additional References VRF-Aware Dynamic NAT Mapping with HSRP

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
NAT commands	Cisco IOS IP Addressing Services Command Reference
Static NAT with HSRP	"Static NAT Mapping with HSRP" module of the *IP Addressing: NAT Configuration Guide*

Standards & RFCs

Standard/RFC	Title
RFC 826	*An Ethernet Address Resolution Protocol or Converting Network Protocol Addresses*
RFC 903	*A Reverse Address Resolution Protocol*
RFC 1027	*Using ARP to Implement Transparent Subnet Gateways*

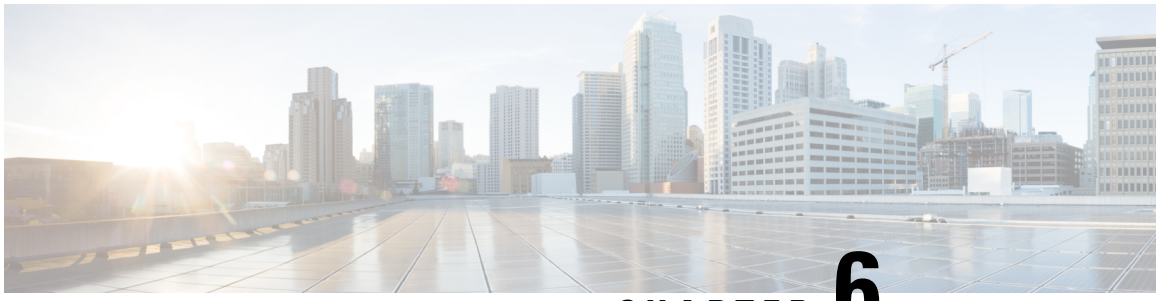
Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/support

Feature Information for VRF-Aware Dynamic NAT Mapping with HSRP

Table 6: Feature Information for VRF-Aware Dynamic NAT Mapping with HSRP

Feature Name	Releases	Feature Information
VRF-Aware Dynamic NAT Mapping with HSRP	Cisco IOS XE Release 3.15S	The VRF-Aware Dynamic NAT Mapping with HSRP feature supports stateless redundancy using HSRP with dynamic Network Address Translation (NAT), Port Address Translation (PAT), and interface overload configuration. Dynamic NAT, PAT and interface overload support HSRP with and without virtual routing and forwarding (VRF) instances. All these configurations are supported in the Carrier Grade NAT (CGN) mode. In Cisco IOS XE Release 3.15S, this feature was supported on Cisco ASR 1000 Series Aggregation Services Routers, Cisco Cloud Services Router 1000V Series, and Cisco ISR 4000 Series Integrated Services Routers. The following commands were updated for this release:**show arp**, **show ip alias**, **show ip nat translations**, and **show standby brief**.
Allow same ACL/router-map on multiple NAT statements	Cisco IOS XE Denali 16.3.1	The Allow use of same ACL/router-map on multiple NAT statements feature supports usage of same ACL for configuring both dynamic mapping and static mapping in NAT. Dynamic mapping is given the precedence over static mapping regardless of the configuration order. The precedence of dynamic mapping over static mapping using the sequence number of the class ensures class order consistency in NAT. This feature uses no new or modified commands.



CHAPTER 6

Configuring Stateful Interchassis Redundancy

The Stateful Interchassis Redundancy feature enables you to configure pairs of devices to act as backups for each other.

This module describes conceptual information about and tasks for configuring stateful interchassis redundancy.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Stateful Interchassis Redundancy

All application redundancy configurations, including Network Address Translation (NAT) rules that have redundancy group associations and mapping IDs, must be identical on both devices, or NAT sessions will not be synchronized between devices and NAT redundancy will not work.

Restrictions for Stateful Interchassis Redundancy

- By default, Network Address Translation (NAT) high availability (inter and intrabox) does not replicate HTTP sessions to the standby device. To replicate HTTP sessions on the standby device during a switchover, you must configure the **ip nat switchover replication http** command.
- During NAT payload translations with certain applications, there can be IP addresses in the payload that require NAT translation. The application-level gateway (ALG) for that specific application parses the packet for these IP addresses, NAT translates these addresses, and the ALG writes the translated addresses back into the packet.

  Fixup denotes the writing of the translated IP address back into the packet. The write back of data can change the length of a packet, which results in the adjustment of the packet's TCP sequence (SEQ) or acknowledgment (ACK) values by NAT for the life of the TCP connection. NAT writes the new TCP SEQ/ACK values into the packet during SEQ/ACK fixup.

  For example, during a TCP ALG session, SEQ/ACK values may require fixup with mainly ASCII applications such as Domain Name System (DNS), FTP/FTP64, H.323, Real Time Streaming Protocol (RTSP), and Session Initiation Protocol (SIP). This SEQ/ACK adjustment information gets associated with the NAT session and is synchronized to the standby device periodically.

  During a stateful switchover, if the SEQ/ACK information is not completely synchronized to the new active device it is likely that the TCP connection would be reset by endpoints of the application.
- Stateful interchassis redundancy cannot coexist with intrachassis redundancy, including software redundancy.
- In Service Software Upgrade (ISSU) is not supported.
- When changing the paired-address-pooling, bulk port-allocation, or NAT mode settings the following steps must be followed:
  1. Shutdown the redundancy group and NAT interfaces on the standby device using the **shutdown** command. Clear NAT sessions on the standby device after shutting down the redundancy group.
  2. Change the paired-address-pooling, bulk port-allocation, or NAT mode on the standby device first and then on the active device.
  3. Configure the **no shutdown** command for the redundancy group and NAT interfaces on the standby device.
- In a NAT Stateful Interchassis Redundancy configuration, it is mandatory that both peers use the same inside and outside NAT interfaces. If the interfaces are not same, it can lead to duplicate NAT entries.
- The following translations are not synchronized to the standby router :
  - Translations created based on an interface overload rule
  - ICMP requests

Note For a standalone NAT router, shut down the NAT interfaces before you make a configuration change.

Information About Stateful Interchassis Redundancy

Stateful Interchassis Redundancy Overview

You can configure the Stateful Interchassis Redundancy feature to determine the active device from a group of devices, based on a number of failover conditions. When a failover occurs, the standby device seamlessly takes over, starts performing traffic forwarding services, and maintains a dynamic routing table.

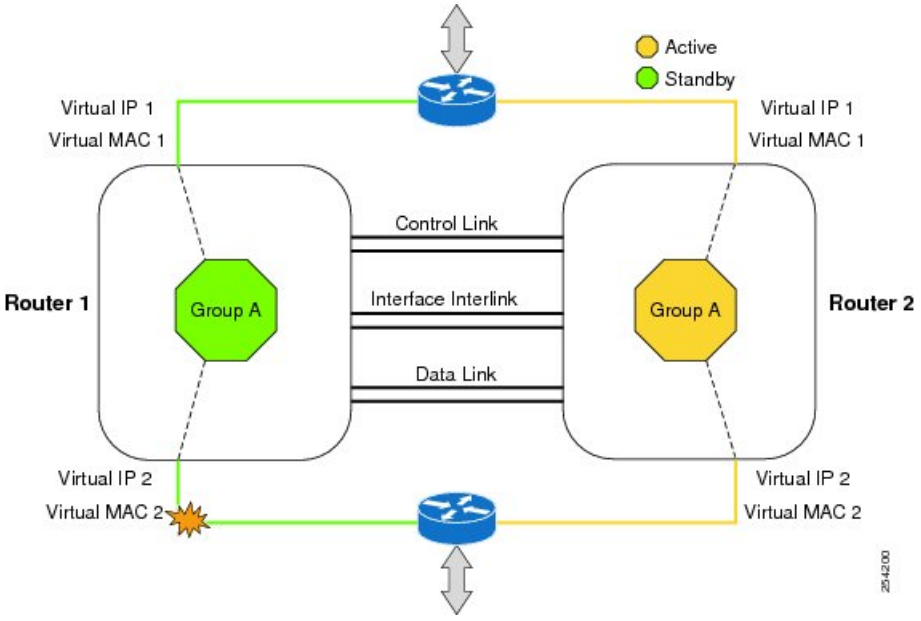
Stateful Interchassis Redundancy Operation

You can configure pairs of devices to act as hot standbys for each other. Redundancy is configured on an interface basis. Pairs of redundant interfaces are known as redundancy groups (RGs). Redundancy occurs at an application level and does not require a complete physical failure of the interface or device for a switchover of the application to occur. When a switchover occurs, the application activity continues to run seamlessly on the redundant interface.

The figure below depicts an active/standby load-sharing scenario. The figure shows how an RG is configured for a pair of devices that has one outgoing interface. Group A on Router 1 is the active RG and Group A on Router 2 is the standby RG.

Redundant devices are joined by a configurable control link and a data synchronization link. The control link is used to communicate the status of devices. The data synchronization link is used to transfer stateful information from Network Address Translation (NAT) and the firewall and synchronize the stateful database. The pairs of redundant interfaces are configured with the same unique ID number known as the redundant interface identifier (RII).

Figure 6: Redundancy Group Configuration—One Outgoing Interface



The status of redundancy group members is determined through the use of hello messages sent over the control link. The software considers either device not responding to a hello message within a configurable amount

of time to be a failure and initiates a switchover. For the software to detect a failure in milliseconds, control links run the failover protocol that is integrated with the Bidirectional Forwarding Detection (BFD) protocol. You can configure the following parameters for hello messages:

- Hello time—Interval at which hello messages are sent.
- Hold time—Amount of time before which the active or standby device is declared to be down.

The hello time defaults to 3 seconds to align with the Hot Standby Router Protocol (HSRP), and the hold time defaults to 10 seconds. You can also configure these timers in milliseconds by using the **timers hellotime msec** command.

To determine the pairs of interfaces that are affected by the switchover, you must configure a unique ID for each pair of redundant interfaces. This ID is known as the RII that is associated with the interface.

A switchover to the standby device can occur when the priority setting that is configured on each device changes. The device with the highest priority value acts as the active device. If a fault occurs on either the active or standby device, the priority of the device is decremented by a configurable amount known as the weight. If the priority of the active device falls below the priority of the standby device, a switchover occurs and the standby device becomes the active device. This default behavior can be overridden by disabling the preemption attribute for the RG. You can also configure each interface to decrease the priority when the Layer 1 state of the interface goes down. The priority that is configured overrides the default priority of an RG.

Each failure event that causes a modification of an RG priority generates a syslog entry that contains a time stamp, the RG that was affected, the previous priority, the new priority, and a description of the failure event cause.

A switchover also can occur when the priority of a device or interface falls below a configurable threshold level.

A switchover to the standby device occurs under the following circumstances:

- Power loss or a reload occurs on the active device (including reloads).
- The run-time priority of the active device goes below that of the standby device (with preempt configured).
- The run-time priority of the active device goes below that of the configured threshold.
- The redundancy group on the active device is reloaded manually. Use the **redundancy application reload group** *rg-number* command for a manual reload.

Associations with Firewalls and NAT

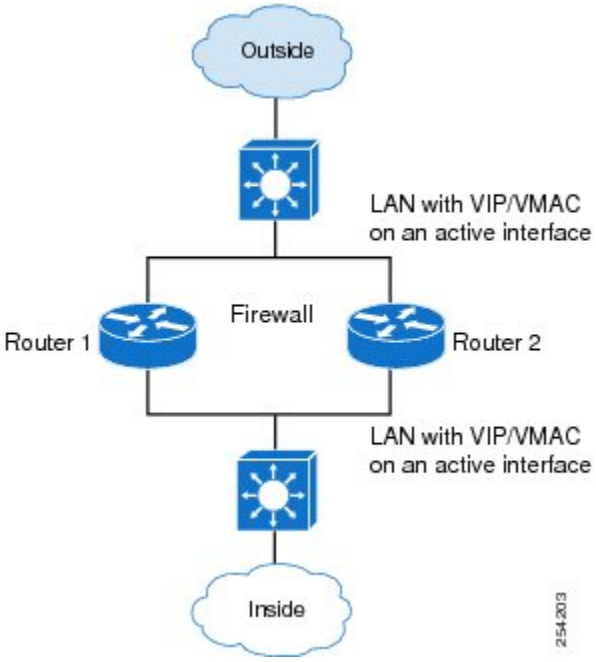
Firewalls use the association of the redundancy group with a traffic interface.

Network Address Translation (NAT) associates the redundancy group with a mapping ID.

LAN-LAN Topology

The figure below shows the LAN-LAN topology. In a LAN-LAN topology, all participating devices are connected to each other through LAN interfaces on both the inside and the outside. In this scenario, traffic is often directed to the correct firewall if static routing is configured on the upstream or downstream devices to an appropriate virtual IP address. Cisco ASR 1000 Aggregation Services Routers participate in dynamic routing with upstream or downstream devices. The dynamic routing configuration supported on LAN-facing interfaces must not introduce a dependency on the routing protocol convergence; otherwise, fast failover requirements will not be met.

Figure 7: LAN-LAN Topology



How to Configure Stateful Interchassis Redundancy

Configuring the Control Interface Protocol

The configuration for the control interface protocol consists of the following elements:

- Authentication information
- Group name
- Hello time
- Hold time
- Protocol instance
- Use of the bidirectional forwarding direction (BFD) protocol

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **redundancy**
4. **mode none**
5. **application redundancy**
6. **protocol** *number*

7. **name** *instance-name*
8. **timers hellotime** [**msec**] *number* **holdtime** [**msec**] *number*
9. **authentication** {**text** *string* | **md5 key-string** [**0** | **7**] *key* | **md5 key-chain** *key-chain-name*}
10. **bfd**
11. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**redundancy** **Example:** `Device(config)# redundancy`	Enters redundancy configuration mode.
Step 4	**mode none** **Example:** `Device(config-red)# mode none`	Sets the redundancy mode to none, which is required for this feature.
Step 5	**application redundancy** **Example:** `Device(config-red)# application redundancy`	Enters redundancy application configuration mode.
Step 6	**protocol** *number* **Example:** `Device(config-red-app)# protocol 4`	Specifies the protocol instance that will be attached to a control interface, and enters redundancy application protocol configuration mode.
Step 7	**name** *instance-name* **Example:** `Device(config-red-app-prot)# name rg1`	(Optional) Specifies an optional alias for the protocol instance.
Step 8	**timers hellotime** [**msec**] *number* **holdtime** [**msec**] *number* **Example:** `Device(config-red-app-prot)# timers hellotime 3 holdtime 10`	Specifies the interval between hello messages sent and the time before a device is declared to be down. • The default time for hello time is 3 seconds and for hold time is 10 seconds.
Step 9	**authentication** {**text** *string* | **md5 key-string** [**0** | **7**] *key* | **md5 key-chain** *key-chain-name*} **Example:**	Specifies authentication information.

	Command or Action	Purpose
	`Device(config-red-app-prot)# authentication text password`	
Step 10	**bfd** **Example:** `Device(config-red-app-prot)# bfd`	(Optional) Enables the integration of the failover protocol running on the control interface with the BFD protocol to achieve failure detection in milliseconds. • BFD is enabled by default.
Step 11	**end** **Example:** `Device(config-red-app-prot)# end`	Exits redundancy application protocol configuration mode and enters privileged EXEC mode.

Configuring a Redundancy Group

Redundancy groups consist of the following configuration elements:

- The amount by which the priority will be decremented for each object.
- Faults (objects) that will decrement the priority.
- Failover priority.
- Failover threshold.
- Group instance.
- Group name.
- Initialization delay timer.
- The interface that is associated with the redundancy group (RG).
- The interface that is used as the control interface.
- The interface that is used as the data interface.
- The redundancy interface identifier (RII) number of the RG interface.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **redundancy**
4. **application redundancy**
5. **group** {**1** | **2**}
6. **name** *group-name*
7. **preempt**
8. **priority** *number* **failover-threshold** *number*
9. **track** *object-number* [**decrement** *number* | **shutdown**]
10. **timers delay** *seconds* [**reload** *seconds*]

11. **control** *interface-name* **protocol** *instance*
12. **data** *interface-name*
13. To create another redundancy group, repeat Steps 3 through 12.
14. **end**
15. **configure terminal**
16. **interface** *type number*
17. **redundancy group** *number* **ip** *address* **exclusive** [**decrement** *number*]
18. **redundancy rii** *number*
19. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**redundancy** **Example:** `Device(config)# redundancy`	Enters redundancy configuration mode.
Step 4	**application redundancy** **Example:** `Device(config-red)# application redundancy`	Enters redundancy application configuration mode.
Step 5	**group** {**1** | **2**} **Example:** `Device(config-red-app)# group 1`	Specifies the redundancy group instance and enters redundancy application group configuration mode.
Step 6	**name** *group-name* **Example:** `Device(config-red-app-grp)# name rg1`	(Optional) Specifies an optional alias for the protocol instance.
Step 7	**preempt** **Example:** `Device(config-red-app-grp)# preempt`	Enables preemption on the group and enables the standby device to preempt the active device regardless of which device has higher priority.
Step 8	**priority** *number* **failover-threshold** *number* **Example:** `Device(config-red-app-grp)# priority 120 failover-threshold 80`	Specifies the initial priority and failover threshold for the redundancy group.

	Command or Action	Purpose
Step 9	**track** *object-number* [**decrement** *number* | **shutdown**] **Example:** `Device(config-red-app-grp)# track 44 decrement 20`	Specifies the amount by which the priority of a redundancy group will be decremented if an event occurs. • You can track multiple objects that influence the priority of the redundancy group.
Step 10	**timers delay** *seconds* [**reload** *seconds*] **Example:** `Device(config-red-app-grp)# timers delay 10 reload 20`	Specifies the amount of time by which the redundancy group will delay role negotiations that start after a fault occurs or after the system is reloaded.
Step 11	**control** *interface-name* **protocol** *instance* **Example:** `Device(config-red-app-grp)# control GigabitEthernet0/1/0 protocol 1`	Specifies the control interface that is used by the redundancy group. • This interface is also associated with an instance of the control interface protocol.
Step 12	**data** *interface-name* **Example:** `Device(config-red-app-grp)# data GigabitEthernet0/1/2`	Specifies the data interface that is used by the redundancy group.
Step 13	To create another redundancy group, repeat Steps 3 through 12.	—
Step 14	**end** **Example:** `Device(config-red-app-grp)# end`	Exits redundancy application group configuration mode and enters privileged EXEC mode.
Step 15	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 16	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 0/0/1`	Selects an interface to associate with the redundancy group and enters interface configuration mode.
Step 17	**redundancy group** *number* **ip** *address* **exclusive** [**decrement** *number*] **Example:** `Device(config-if)# redundancy group 1 ip 10.10.1.1 exclusive decrement 20`	Associates the interface with the redundancy group identified by the *number* argument.
Step 18	**redundancy rii** *number* **Example:** `Device(config-if)# redundancy rii 40`	Specifies a number for the RII associated with this interface. • This number must match the RII of the other interface in the redundancy group.

	Command or Action	Purpose
Step 19	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and enters privileged EXEC mode.

Configuring a Redundant Traffic Interface

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ip address** *ip-address mask*
5. **ip nat outside**
6. **ip virtual-reassembly**
7. **negotiation auto**
8. **redundancy rii** *number*
9. **redundancy group** *number* **ip** *address* **exclusive** [**decrement** *number*]
10. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 0/1/5`	Configures an interface and enters interface configuration mode.
Step 4	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 10.1.1.2 255.0.0.0`	Sets a primary or secondary IP address for an interface.
Step 5	**ip nat outside** **Example:** `Device(config-if)# ip nat outside`	Configures the outside interface for IP address translation.

	Command or Action	Purpose
Step 6	**ip virtual-reassembly** **Example:** `Device(config-if)# ip virtual-reassembly`	Enables Virtual Fragmentation Reassembly (VFR) on an interface.
Step 7	**negotiation auto** **Example:** `Device(config-if)# negotiation auto`	Enables the autonegotiation protocol to configure the speed, duplex, and automatic flow control of the Gigabit Ethernet interface.
Step 8	**redundancy rii** *number* **Example:** `Device(config-if)# redundancy rii 200`	Specifies a number for the redundancy interface identifier (RII) that is associated with this interface. • This number must match the RII of the other interface in the redundancy group.
Step 9	**redundancy group** *number* **ip** *address* **exclusive** [**decrement** *number*] **Example:** `Device(config-if)# redundancy group 1 ip 10.1.1.200 exclusive decrement 10`	Associates the interface with the redundancy group identified by the *number* argument.
Step 10	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and enters privileged EXEC mode.

Configuring NAT with Stateful Interchassis Redundancy

You must use a mapping ID to associate Network Address Translation (NAT) with a redundancy group.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip nat pool** *name start-ip end-ip* {**netmask** *netmask* | **prefix-length** *prefix-length*}
4. **ip nat inside source list** {{*access-list-number* | *access-list-name*} | **route-map** *name*} **pool** *name* [**redundancy** *redundancy-id* [**mapping-id** *map-id* | **overload** | **reversible** | **vrf** *name*]]
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip nat pool** *name start-ip end-ip* {**netmask** *netmask* | **prefix-length** *prefix-length*} **Example:** `Device(config)# ip nat pool VPN-18 10.10.0.0 10.10.255.255 netmask 255.255.0.0`	Defines a pool of IP addresses for NAT.
Step 4	**ip nat inside source list** {{*access-list-number* | *access-list-name*} | **route-map** *name*} **pool** *name* [**redundancy** *redundancy-id* [**mapping-id** *map-id* | **overload** | **reversible** | **vrf** *name*]] **Example:** `Device(config)# ip nat inside source list acl-18 pool VPN-18 redundancy 2 mapping-id 152`	Enables NAT of the inside source address. • You must use a mapping ID to associate NAT with the redundancy group.
Step 5	**end** **Example:** `Device(config)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Managing and Monitoring Stateful Interchassis Redundancy

All configuration commands in this task are optional. You can use the **show** commands in any order.

SUMMARY STEPS

1. **enable**
2. **redundancy application reload group** *number* [**peer** | **self**]
3. **show redundancy application group** [*group-id* | **all**]
4. **show redundancy application transport** {**clients** | **group** [*group-id*]}
5. **show redundancy application protocol** {*protocol-id* | **group** [*group-id*]}
6. **show redundancy application faults group** [*group-id*]
7. **show redundancy application if-mgr group** [*group-id*]
8. **show redundancy application control-interface group** [*group-id*]
9. **show redundancy application data-interface group** [*group-id*]
10. **show monitor event-trace rg_infra all**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
	`Device> enable`	
Step 2	**redundancy application reload group** *number* [**peer** | **self**] **Example:** `Device# redundancy application reload group 2 self`	Forces the active redundancy group (RG) to reload and the standby RG to become the active RG. • Use the **redundancy application reload** command to verify if the redundancy configuration is working. You must enter this command on the active RG.
Step 3	**show redundancy application group** [*group-id* | **all**] **Example:** `Device# show redundancy application group 2`	Displays summary information for the specified group or for all groups.
Step 4	**show redundancy application transport** {**clients** | **group** [*group-id*]} **Example:** `Device# show redundancy application transport group 2`	Displays transport information for the specified group or for all groups.
Step 5	**show redundancy application protocol** {*protocol-id* | **group** [*group-id*]} **Example:** `Device# show redundancy application protocol 2`	Displays protocol information for the specified group or for all groups.
Step 6	**show redundancy application faults group** [*group-id*] **Example:** `Device# show redundancy application faults group 2`	Displays information about faults for the specified group or for all groups.
Step 7	**show redundancy application if-mgr group** [*group-id*] **Example:** `Device# show redundancy application if-mgr group 2`	Displays information about the interface manager (if-mgr) for the specified group or for all groups.
Step 8	**show redundancy application control-interface group** [*group-id*] **Example:** `Device# show redundancy application control-interface group IF-2`	Displays interface information associated with redundancy groups for the specified control interface.
Step 9	**show redundancy application data-interface group** [*group-id*] **Example:** `Device# show redundancy application data-interface group IF-2`	Displays interface information associated with redundancy groups for the specified data interface.
Step 10	**show monitor event-trace rg_infra all** **Example:**	Displays event trace information associated with all redundancy groups.

	Command or Action	Purpose
	Device# show monitor event-trace rg_infra all	

Configuration Examples for Stateful Interchassis Redundancy

Example: Configuring the Control Interface Protocol

```
Device# configure terminal
Device(config)# redundancy
Device(config-red)# mode none
Device(config-red)# application redundancy
Device(config-red-app)# protocol 4
Device(config-red-app-prot)# name rg1
Device(config-red-app-prot)# timers hellotime 3 holdtime 10
Device(config-red-app-prot)# authentication text password
Device(config-red-app-prot)# bfd
```

Example: Configuring a Redundancy Group

```
Device# configure terminal
Device(config)# redundancy
Device(config-red)# application redundancy
Device(config-red-app)# group 1
Device(config-red-app-grp)# name rg1
Device(config-red-app-grp)# preempt
Device(config-red-app-grp)# priority 120 failover-threshold 80
Device(config-red-app-grp)# track 44 decrement 20
Device(config-red-app-grp)# timers delay 10 reload 20
Device(config-red-app-grp)# control GigabitEthernet0/1/0 protocol 1
Device(config-red-app-grp)# data GigabitEthernet0/1/2
Device(config-red-app-grp)# end
Device# configure terminal
Device(config)# interface GigabitEthernet 0/0/1
Device(config-if)# redundancy group 1 ip 10.10.1.1 exclusive decrement 20
Device(config-if)# redundancy rii 40
```

Example: Configuring a Redundant Traffic Interface

```
Device# configure terminal
Device(config)# interface GigabitEthernet 0/1/5
Device(config-if)# ip address 10.1.1.2 255.0.0.0
Device(config-if)# ip nat outside
Device(config-if)# ip virtual-reassembly
Device(config-if)# negotiation auto
Device(config-if)# redundancy rii 200
Device(config-if)# redundancy group 1 ip 10.1.1.200 exclusive decrement 10
```

Example: Configuring NAT with Stateful Interchassis Redundancy

```
Device# configure terminal
Device(config)# ip nat pool VPN-18 10.10.0.0 10.10.255.255 netmask 255.255.0.0
Device(config)# ip nat inside source list acl-18 pool VPN-18 redundancy 2 mapping-id 152
```

Additional References for Stateful Interchassis Redundancy

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
IP addressing commands: complete command syntax, command mode, command history, defaults, usage guidelines, and examples	Cisco IOS IP Addressing Services Command Reference
Fundamental principles of IP addressing and IP routing	*IP Routing Primer*

Standards and RFCs

Standards/RFCs	Title
RFC 791	Internet Protocol
RFC 1338	Classless Inter-Domain Routing (CIDR): an Address Assignment and Aggregation Strategy
RFC 1466	Guidelines for Management of IP Address Space
RFC 1716	Towards Requirements for IP Routers
RFC 1918	Address Allocation for Private Internets
RFC 3330	Special-Use IP Addresses

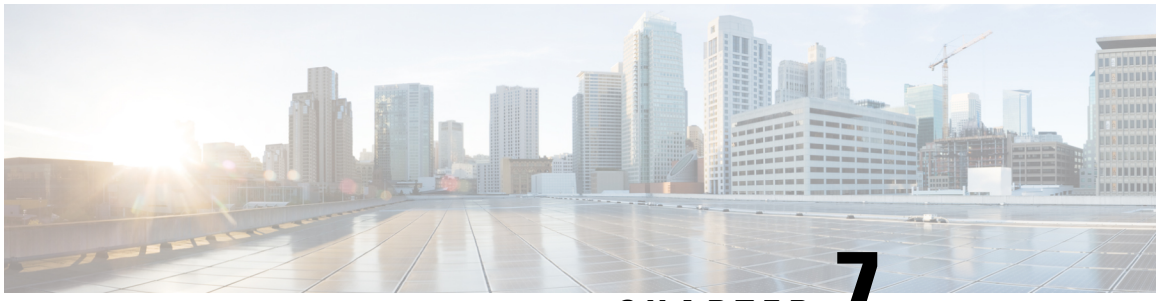
Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Stateful Interchassis Redundancy

Table 7: Feature Information for Stateful Interchassis Redundancy

Feature Name	Releases	Feature Information
Stateful Interchassis Redundancy	Cisco IOS XE Release 3.1S Cisco IOS XE Release 3.14S	The Stateful Interchassis Redundancy feature enables you to configure pairs of devices to act as backups for each other. In Cisco IOS XE Release 3.14S, this feature was supported on Cisco CSR1000v Series Routers.



CHAPTER 7

Interchassis Asymmetric Routing Support for Zone-Based Firewall and NAT

The Interchassis Asymmetric Routing Support for Zone-Based Firewall and NAT feature supports the forwarding of packets from a standby redundancy group to the active redundancy group for packet handling. If this feature is not enabled, the return TCP packets forwarded to the router that did not receive the initial synchronization (SYN) message are dropped because they do not belong to any known existing session.

This module provides an overview of asymmetric routing and describes how to configure asymmetric routing

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for Interchassis Asymmetric Routing Support for Zone-Based Firewall and NAT

The following restrictions apply to the Interchassis Asymmetric Routing Support feature:

- LANs that use virtual IP addresses and virtual MAC (VMAC) addresses do not support asymmetric routing.
- In Service Software Upgrade (ISSU) is not supported.

The following features are not supported by the VRF-Aware Asymmetric Routing Support feature:

- Cisco Trustsec
- Edge switching services
- Header compression
- IPsec
- Policy Based Routing (PBR)
- Port bundle
- Lawful intercept
- Layer 2 Tunneling Protocol (L2TP)
- Locator/ID Separation Protocol (LISP) inner packet inspection
- Secure Shell (SSL) VPN
- Session Border Controller (SBC)
- If you enable NAT on the primary and backup WAN link, switchover between the primary and backup interface is not supported. NAT backup interface overload is not supported on the following platforms:
  - ASR1000 Series Aggregation Services Routers
  - ISR4000 Series Integrated Services Routers
  - ISR1000 Series Integrated Services Routers
  - CSR1000 Series Cloud Services Routers

Information About Interchassis Asymmetric Routing Support for Zone-Based Firewall and NAT

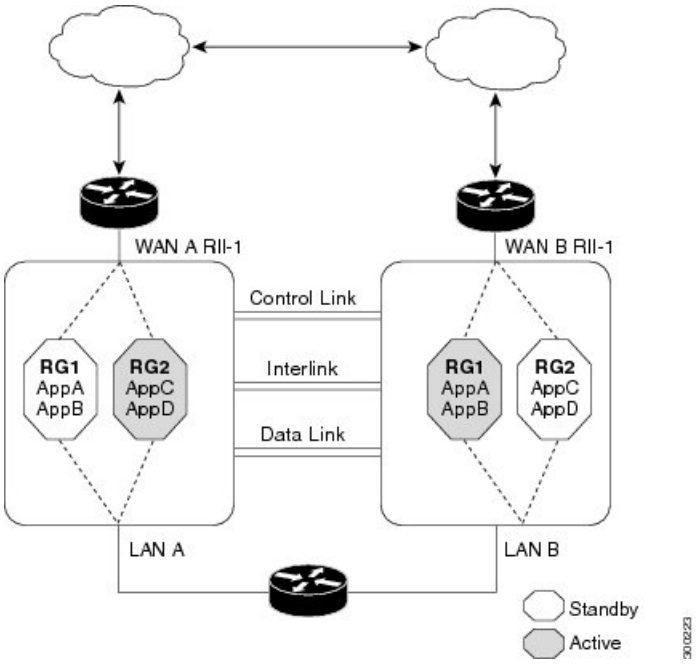
Asymmetric Routing Overview

Asymmetric routing occurs when packets from TCP or UDP connections flow in different directions through different routes. In asymmetric routing, packets that belong to a single TCP or UDP connection are forwarded through one interface in a redundancy group (RG), but returned through another interface in the same RG. In asymmetric routing, the packet flow remains in the same RG. When you configure asymmetric routing, packets received on the standby RG are redirected to the active RG for processing. If asymmetric routing is not configured, the packets received on the standby RG may be dropped.

Asymmetric routing determines the RG for a particular traffic flow. The state of the RG is critical in determining the handling of packets. If an RG is active, normal packet processing is performed. In case the RG is in a standby state and you have configured asymmetric routing and the **asymmetric-routing always-divert enable** command, packets are diverted to the active RG. Use the **asymmetric-routing always-divert enable** command to always divert packets received from the standby RG to the active RG.

The figure below shows an asymmetric routing scenario with a separate asymmetric-routing interlink interface to divert packets to the active RG.

Figure 8: Asymmetric Routing Scenario



The following rules apply to asymmetric routing:

- 1:1 mapping exists between the redundancy interface identifier (RII) and the interface.
- 1:*n* mapping exists between the interface and an RG. (An asymmetric routing interface can receive traffic from and send traffic to multiple RGs. For a non asymmetric-routing interface (normal LAN interface), a 1:1 mapping exists between the interface and the RG.)
- 1:*n* mapping exists between an RG and applications that use it. (Multiple applications can use the same RG).
- 1:1 mapping exists between an RG and the traffic flow. The traffic flow must map only to a single RG. If a traffic flow maps to multiple RGs, an error occurs.
- 1:1 or 1:*n* mapping can exist between an RG and an asymmetric-routing interlink as long as the interlink has sufficient bandwidth to support all the RG interlink traffic.

Asymmetric routing consists of an interlink interface that handles all traffic that is to be diverted. The bandwidth of the asymmetric-routing interlink interface must be large enough to handle all expected traffic that is to be diverted. An IPv4 address must be configured on the asymmetric-routing interlink interface, and the IP address of the asymmetric routing interface must be reachable from this interface.

Note We recommend that the asymmetric-routing interlink interface be used for interlink traffic only and not be shared with high availability control or data interfaces because the amount of traffic on the asymmetric-routing interlink interface could be quite high.

Asymmetric Routing Support in Firewalls

For intrabox asymmetric routing support, the firewall does a stateful Layer 3 and Layer 4 inspection of Internet Control Message Protocol (ICMP), TCP, and UDP packets. The firewall does a stateful inspection of TCP packets by verifying the window size and order of packets. The firewall also requires the state information from both directions of the traffic for stateful inspection. The firewall does a limited inspection of ICMP information flows. It verifies the sequence number associated with the ICMP echo request and response. The firewall does not synchronize any packet flows to the standby redundancy group (RG) until a session is established for that packet. An established session is a three-way handshake for TCP, the second packet for UDP, and informational messages for ICMP. All ICMP flows are sent to the active RG.

The firewall does a stateless verification of policies for packets that do not belong to the ICMP, TCP, and UDP protocols.

The firewall depends on bidirectional traffic to determine when a packet flow should be aged out and diverts all inspected packet flows to the active RG. Packet flows that have a pass policy and that include the same zone with no policy or a drop policy are not diverted.

Note The firewall does not support the **asymmetric-routing always-divert enable** command that diverts packets received on the standby RG to the active RG. By default, the firewall forces all packet flows to be diverted to the active RG.

Asymmetric Routing in NAT

By default, when asymmetric routing is configured, Network Address Translation (NAT) processes non-ALG packets on the standby RG, instead of forwarding them to the active. The NAT-only configuration (that is when the firewall is not configured) can use both the active and standby RGs for processing packets. If you have a NAT-only configuration and you have configured asymmetric routing, the default asymmetric routing rule is that NAT will selectively process packets on the standby RG. You can configure the **asymmetric-routing always-divert enable** command to divert packets received on the standby RG to the active RG. Alternatively, if you have configured the firewall along with NAT, the default asymmetric routing rule is to always divert the packets to the active RG.

When NAT receives a packet on the standby RG and if you have not configured the diverting of packets, NAT does a lookup to see if a session exists for that packet. If a session exists and there is no ALG associated for that session, NAT processes the packet on the standby RG. The processing of packets on the standby RG when a session exists significantly increases the bandwidth of the NAT traffic.

ALGs are used by NAT to identify and translate payload and to create child flows. ALGs require a two-way traffic to function correctly. NAT must divert all traffic to the active RG for any packet flow that is associated with an ALG. This is accomplished by checking if ALG data that is associated with the session is found on the standby RG. If ALG data exits, the packet is diverted for asymmetric routing.

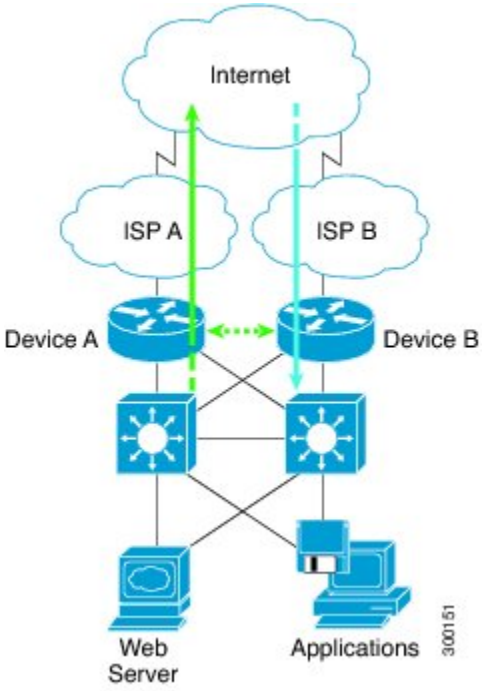
VRF-Aware Software Infrastructure (VASI) support was added in Cisco IOS XE Release 3.16S. Multiprotocol Label Switching (MPLS) asymmetric routing is also supported.

In Cisco IOS XE Release 3.16S, NAT supports asymmetric routing with ALGs, Carrier Grade NAT (CGN), and virtual routing and forwarding (VRF) instances. No configuration changes are required to enable asymmetric routing with ALGs, CGN, or VRF. For more information, see the section, "Example: Configuring Asymmetric Routing with VRF".

Asymmetric Routing in a WAN-LAN Topology

Asymmetric routing supports only a WAN-LAN topology. In a WAN-LAN topology, devices are connected through LAN interfaces on the inside and WAN interfaces on the outside. There is no control on the routing of return traffic received through WAN links. Asymmetric routing controls the routing of return traffic received through WAN links in a WAN-LAN topology. The figure below shows a WAN-LAN topology.

Figure 9: Asymmetric Routing in a WAN-LAN Topology



VRF-Aware Asymmetric Routing in Zone-Based Firewalls

In Cisco IOS XE Release 3.14S, zone-based firewalls support the VRF-Aware Interchassis Asymmetric Routing feature. The feature supports Multiprotocol Label Switching (MPLS).

During asymmetric routing diversion, the VPN routing and forwarding (VRF) name hash value is sent with diverted packets. The VRF name hash value is converted to the local VRF ID and table ID at the active device after the diversion.

When diverted packets reach the active device on which Network Address Translation (NAT) and the zone-based firewall are configured, the firewall retrieves the VRF ID from NAT or NAT64 and saves the VRF ID in the firewall session key.

The following section describes the asymmetric routing packet flow when only the zone-based firewall is configured on a device:

- When MPLS is configured on a device, the VRF ID handling for diverted packets is the same as the handling of non-asymmetric routing diverted packets. An MPLS packet is diverted to the active device, even though the MPLS label is removed at the standby device. The zone-based firewall inspects the packet at the egress interface, and the egress VRF ID is set to zero, if MPLS is detected at this interface. The firewall sets the ingress VRF ID to zero if MPLS is configured at the ingress interface.
- When a Multiprotocol Label Switching (MPLS) packet is diverted to the active device from the standby device, the MPLS label is removed before the asymmetric routing diversion happens.
- When MPLS is not configured on a device, an IP packet is diverted to the active device and the VRF ID is set. The firewall gets the local VRF ID, when it inspects the packet at the egress interface.

VRF mapping between active and standby devices require no configuration changes.

VRF-Aware Asymmetric Routing in NAT

In Cisco IOS XE Release 3.14S, Network Address Translation supports VRF-aware interchassis asymmetric routing. VRF-aware interchassis asymmetric routing uses message digest (MD) 5 hash of the VPN routing and forwarding (VRF) name to identify the VRF and datapath in the active and standby devices to retrieve the local VRF ID from the VRF name hash and viceversa.

For VRF-aware interchassis asymmetric routing, the VRFs on active and standby devices must have the same VRF name. However, the VRF ID need not be identical on both devices because the VRF ID is mapped based on the VRF name on the standby and active devices during asymmetric routing diversion or box-to-box high availability synchronization.

In case of MD5 hash collision for VRF names, the firewall and NAT sessions that belong to the VRF are not synced to the standby device.

VRF mapping between active and standby devices require no configuration changes.

How to Configure Interchassis Asymmetric Routing Support for Zone-Based Firewall and NAT

Configuring a Redundancy Application Group and a Redundancy Group Protocol

Redundancy groups consist of the following configuration elements:

- The amount by which the priority will be decremented for each object.
- Faults (objects) that decrement the priority
- Failover priority
- Failover threshold
- Group instance
- Group name

- Initialization delay timer

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **redundancy**
4. **application redundancy**
5. **group** *id*
6. **name** *group-name*
7. **priority** *value* [**failover threshold** *value*]
8. **preempt**
9. **track** *object-number* **decrement** *number*
10. **exit**
11. **protocol** *id*
12. **timers hellotime** {*seconds* | **msec** *msec*} **holdtime** {*seconds* | **msec** *msec*}
13. **authentication** {**text** *string* | **md5 key-string** [**0** | **7**] *key* [**timeout** *seconds*] | **key-chain** *key-chain-name*}
14. **bfd**
15. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**redundancy** **Example:** `Device(config)# redundancy`	Enters redundancy configuration mode.
Step 4	**application redundancy** **Example:** `Device(config-red)# application redundancy`	Configures application redundancy and enters redundancy application configuration mode.
Step 5	**group** *id* **Example:** `Device(config-red-app)# group 1`	Configures a redundancy group and enters redundancy application group configuration mode.

	Command or Action	Purpose
Step 6	**name** *group-name* **Example:** `Device(config-red-app-grp)# name group1`	Specifies an optional alias for the protocol instance.
Step 7	**priority** *value* [**failover threshold** *value*] **Example:** `Device(config-red-app-grp)# priority 100 failover threshold 50`	Specifies the initial priority and failover threshold for a redundancy group.
Step 8	**preempt** **Example:** `Device(config-red-app-grp)# preempt`	Enables preemption on the redundancy group and enables the standby device to preempt the active device. • The standby device preempts only when its priority is higher than that of the active device.
Step 9	**track** *object-number* **decrement** *number* **Example:** `Device(config-red-app-grp)# track 50 decrement 50`	Specifies the priority value of a redundancy group that will be decremented if an event occurs on the tracked object.
Step 10	**exit** **Example:** `Device(config-red-app-grp)# exit`	Exits redundancy application group configuration mode and enters redundancy application configuration mode.
Step 11	**protocol** *id* **Example:** `Device(config-red-app)# protocol 1`	Specifies the protocol instance that will be attached to a control interface and enters redundancy application protocol configuration mode.
Step 12	**timers hellotime** {*seconds* | **msec** *msec*} **holdtime** {*seconds* | **msec** *msec*} **Example:** `Device(config-red-app-prtcl)# timers hellotime 3 holdtime 10`	Specifies the interval between hello messages sent and the time period before which a device is declared to be down. • Holdtime should be at least three times the hellotime.
Step 13	**authentication** {**text** *string* | **md5 key-string** [**0** | **7**] *key* [**timeout** *seconds*] | **key-chain** *key-chain-name*} **Example:** `Device(config-red-app-prtcl)# authentication md5 key-string 0 n1 timeout 100`	Specifies authentication information.
Step 14	**bfd** **Example:** `Device(config-red-app-prtcl)# bfd`	Enables the integration of the failover protocol running on the control interface with the Bidirectional Forwarding Detection (BFD) protocol to achieve failure detection in milliseconds. • BFD is enabled by default.

	Command or Action	Purpose
Step 15	**end** **Example:** `Device(config-red-app-prtcl)# end`	Exits redundancy application protocol configuration mode and enters privileged EXEC mode.

Configuring Data, Control, and Asymmetric Routing Interfaces

In this task, you configure the following redundancy group (RG) elements:

- The interface that is used as the control interface.
- The interface that is used as the data interface.
- The interface that is used for asymmetric routing. This is an optional task. Perform this task only if you are configuring asymmetric routing for Network Address Translation (NAT).

Note Asymmetric routing, data, and control must be configured on separate interfaces for zone-based firewall. However, for Network Address Translation (NAT), asymmetric routing, data, and control can be configured on the same interface.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **redundancy**
4. **application redundancy**
5. **group** *id*
6. **data** *interface-type interface-number*
7. **control** *interface-type interface-number* **protocol** *id*
8. **timers delay** *seconds* [**reload** *seconds*]
9. **asymmetric-routing interface** *type number*
10. **asymmetric-routing always-divert enable**
11. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**redundancy** **Example:** `Device(config)# redundancy`	Enters redundancy configuration mode.
Step 4	**application redundancy** **Example:** `Device(config-red)# application redundancy`	Configures application redundancy and enters redundancy application configuration mode.
Step 5	**group** *id* **Example:** `Device(config-red-app)# group 1`	Configures a redundancy group (RG) and enters redundancy application group configuration mode.
Step 6	**data** *interface-type interface-number* **Example:** `Device(config-red-app-grp)# data GigabitEthernet 0/0/1`	Specifies the data interface that is used by the RG.
Step 7	**control** *interface-type interface-number* **protocol** *id* **Example:** `Device(config-red-app-grp)# control GigabitEthernet 1/0/0 protocol 1`	Specifies the control interface that is used by the RG. • The control interface is also associated with an instance of the control interface protocol.
Step 8	**timers delay** *seconds* [**reload** *seconds*] **Example:** `Device(config-red-app-grp)# timers delay 100 reload 400`	Specifies the time required for an RG to delay role negotiations that start after a fault occurs or the system is reloaded.
Step 9	**asymmetric-routing interface** *type number* **Example:** `Device(config-red-app-grp)# asymmetric-routing interface GigabitEthernet 0/1/1`	Specifies the asymmetric routing interface that is used by the RG.
Step 10	**asymmetric-routing always-divert enable** **Example:** `Device(config-red-app-grp)# asymmetric-routing always-divert enable`	Always diverts packets received from the standby RG to the active RG.
Step 11	**end** **Example:** `Device(config-red-app-grp)# end`	Exits redundancy application group configuration mode and enters privileged EXEC mode.

Configuring a Redundant Interface Identifier and Asymmetric Routing on an Interface

Note

- You must not configure a redundant interface identifier (RII) on an interface that is configured either as a data interface or as a control interface.
- You must configure the RII and asymmetric routing on both active and standby devices.
- You cannot enable asymmetric routing on the interface that has a virtual IP address configured.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **redundancy rii** *id*
5. **redundancy group** *id* [**decrement** *number*]
6. **redundancy asymmetric-routing enable**
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface GigabitEthernet 0/1/3`	Selects an interface to be associated with the redundancy group (RG) and enters interface configuration mode.
Step 4	**redundancy rii** *id* **Example:** `Device(config-if)# redundancy rii 600`	Configures the redundancy interface identifier (RII).
Step 5	**redundancy group** *id* [**decrement** *number*] **Example:** `Device(config-if)# redundancy group 1 decrement 20`	Enables the RG redundancy traffic interface configuration and specifies the amount to be decremented from the priority when the interface goes down. **Note** You need not configure an RG on the traffic interface on which asymmetric routing is enabled.

	Command or Action	Purpose
Step 6	**redundancy asymmetric-routing enable** **Example:** `Device(config-if)# redundancy asymmetric-routing enable`	Establishes an asymmetric flow diversion tunnel for each RG.
Step 7	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and enters privileged EXEC mode.

Configuring Dynamic Inside Source Translation with Asymmetric Routing

The following configuration is a sample dynamic inside source translation with asymmetric routing. You can configure asymmetric routing with the following types of NAT configurations—dynamic outside source, static inside and outside source, and Port Address Translation (PAT) inside and outside source translations. For more information on different types of NAT configurations, see the "Configuring NAT for IP Address Conservation" chapter.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ip address** *ip-address mask*
5. **ip nat outside**
6. **exit**
7. **redundancy**
8. **application redundancy**
9. **group** *id*
10. **asymmetric-routing always-divert enable**
11. **end**
12. **configure terminal**
13. **ip nat pool** *name start-ip end-ip* {*mask* | **prefix-length** *prefix-length*}
14. **exit**
15. **ip nat inside source list** *acl-number* **pool** *name* **redundancy** *redundancy-id* **mapping-id** *map-id*
16. **access-list** *standard-acl-number* **permit** *source-address wildcard-bits*
17. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 0/1/3`	Configures an interface and enters interface configuration mode.
Step 4	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 10.1.1.1 255.255.255.0`	Sets a primary IP address for an interface.
Step 5	**ip nat outside** **Example:** `Device(config-if)# ip nat outside`	Marks the interface as connected to the outside.
Step 6	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and enters global configuration mode.
Step 7	**redundancy** **Example:** `Device(config)# redundancy`	Configures redundancy and enters redundancy configuration mode.
Step 8	**application redundancy** **Example:** `Device(config-red)# application redundancy`	Configures application redundancy and enters redundancy application configuration mode.
Step 9	**group** *id* **Example:** `Device(config-red-app)# group 1`	Configures a redundancy group and enters redundancy application group configuration mode.
Step 10	**asymmetric-routing always-divert enable** **Example:** `Device(config-red-app-grp)# asymmetric-routing always-divert enable`	Diverts the traffic to the active device.
Step 11	**end** **Example:** `Device(config-red-app-grp)# end`	Exits redundancy application group configuration mode and enters privileged EXEC mode.
Step 12	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 13	**ip nat pool** *name start-ip end-ip* {*mask* | **prefix-length** *prefix-length*} **Example:** `Device(config)# ip nat pool pool1 prefix-length 24`	Defines a pool of global addresses. • Enters IP NAT pool configuration mode.
Step 14	**exit** **Example:** `Device(config-ipnat-pool)# exit`	Exits IP NAT pool configuration mode and enters global configuration mode.
Step 15	**ip nat inside source list** *acl-number* **pool** *name* **redundancy** *redundancy-id* **mapping-id** *map-id* **Example:** `Device(config)# ip nat inside source list pool pool1 redundancy 1 mapping-id 100`	Enables NAT of the inside source address and associates NAT with a redundancy group by using the mapping ID.
Step 16	**access-list** *standard-acl-number* **permit** *source-address wildcard-bits* **Example:** `Device(config)# access-list 10 permit 10.1.1.1 255.255.255.0`	Defines a standard access list for the inside addresses that are to be translated.
Step 17	**end** **Example:** `Device(config)# end`	Exits global configuration mode and enters privileged EXEC mode.

Configuration Examples for Interchassis Asymmetric Routing Support for Zone-Based Firewall and NAT

Example: Configuring a Redundancy Application Group and a Redundancy Group Protocol

```
Device# configure terminal
Device(config)# redundancy
Device(config-red)# application redundancy
Device(config-red-app)# group 1
Device(config-red-app-grp)# name group1
Device(config-red-app-grp)# priority 100 failover threshold 50
Device(config-red-app-grp)# preempt
Device(config-red-app-grp)# track 50 decrement 50
Device(config-red-app-grp)# exit
Device(config-red-app)# protocol 1
Device(config-red-app-prtcl)# timers hellotime 3 holdtime 10
Device(config-red-app-prtcl)# authentication md5 key-string 0 n1 timeout 100
```

```
Device(config-red-app-prtcl)# bfd
Device(config-red-app-prtcl)# end
```

Example: Configuring Data, Control, and Asymmetric Routing Interfaces

```
Device# configure terminal
Device(config)# redundancy
Device(config-red)# application redundancy
Device(config-red-app)# group 1
Device(config-red-app-grp)# data GigabitEthernet 0/0/1
Device(config-red-app-grp)# control GigabitEthernet 1/0/0 protocol 1
Device(config-red-app-grp)# timers delay 100 reload 400
Device(config-red-app-grp)# asymmetric-routing interface GigabitEthernet 0/1/1
Device(config-red-app-grp)# asymmetric-routing always-divert enable
Device(config-red-app-grp)# end
```

Example: Configuring a Redundant Interface Identifier and Asymmetric Routing on an Interface

```
Device# configure terminal
Device(config)# interface GigabitEthernet 0/1/3
Device(config-if)# redundancy rii 600
Device(config-if)# redundancy group 1 decrement 20
Device(config-if)# redundancy asymmetric-routing enable
Device(config-if)# end
```

Example: Configuring Dynamic Inside Source Translation with Asymmetric Routing

```
Device(config)# interface gigabitethernet 0/1/3
Device(config-if)# ip address 10.1.1.1 255.255.255.0
Device(config-if)# ip nat outside
Device(config-if)# exit
Device(config)# redundancy
Device(config-red)# application redundancy
Device(config-red-app)# group 1
Device(config-red-app-grp)# asymmetric-routing always-divert enable
Device(config-red-app-grp)# end
Device# configure terminal
Device(config)# ip nat pool pool1 prefix-length 24
Device(config-ipnat-pool)# exit
Device(config)# ip nat inside source list pool pool1 redundancy 1 mapping-id 100
Device(config)# access-list 10 permit 10.1.1.1 255.255.255.0
```

Example: Configuring VRF-Aware NAT for WAN-WAN Topology with Symmetric Routing Box-to-Box Redundancy

The following is a sample WAN-to-WAN symmetric routing configuration:

```
vrf definition Mgmt-intf
 address-family ipv4
  exit-address-family
!
address-family ipv6
 exit-address-family
!
!
vrf definition VRFA
 rd 100:1
 route-target export 100:1
 route-target import 100:1
 address-family ipv4
  exit-address-family
!
!
no logging console
no aaa new-model
!
multilink bundle-name authenticated
!
redundancy
 mode sso
 application redundancy
  group 1
   preempt
   priority 120
   control GigabitEthernet 0/0/1 protocol 1
   data GigabitEthernet 0/0/2
!
!
!
!
ip tftp source-interface GigabitEthernet0
ip tftp blocksize 8192
!
track 1 interface GigabitEthernet 0/0/4 line-protocol
!
interface Loopback 0
 ip address 209.165.201.1 255.255.255.224
!
interface GigabitEthernet 0/0/0
 vrf forwarding VRFA
 ip address 192.168.0.1 255.255.255.248
 ip nat inside
 negotiation auto
 bfd interval 50 min_rx 50 multiplier 3
 redundancy rii 2
!
interface GigabitEthernet 0/0/1
 ip address 209.165.202.129 255.255.255.224
 negotiation auto
!
interface GigabitEthernet 0/0/2
 ip address 192.0.2.1 255.255.255.224
 negotiation auto
!
interface GigabitEthernet 0/0/3
 ip address 198.51.100.1 255.255.255.240
 negotiation auto
!
interface GigabitEthernet 0/0/4
```

```
 ip address 203.0.113.1 255.255.255.240
 negotiation auto
!
interface GigabitEthernet 0
 vrf forwarding Mgmt-intf
 ip address 172.16.0.1 255.255.0.0
 negotiation auto
!
interface vasileft 1
 vrf forwarding VRFA
 ip address 10.4.4.1 255.255.0.0
 ip nat outside
 no keepalive
!
interface vasiright 1
ip address 10.4.4.2 255.255.0.0
no keepalive
!
router mobile
!
router bgp 577
 bgp router-id 1.1.1.1
 bgp log-neighbor-changes
 neighbor 203.0.113.1 remote-as 223
 neighbor 203.0.113.1 description PEERING to PTNR neighbor 10.4.4.1 remote-as 577
 neighbor 10.4.4.1 description PEEERING to VASI VRFA interface
!
address-family ipv4
 network 203.0.113.1 mask 255.255.255.240
 network 10.4.0.0 mask 255.255.0.0
 network 209.165.200.224 mask 255.255.255.224
 neighbor 203.0.113.1 activate
 neighbor 10.4.4.1 activate
 neighbor 10.4.4.1 next-hop-self
 exit-address-family
!
address-family ipv4 vrf VRFA
 bgp router-id 4.4.4.4
 network 192.168.0.0 mask 255.255.255.248
 network 10.4.0.0 mask 255.255.0.0
 redistribute connected
 redistribute static
 neighbor 192.168.0.2 remote-as 65004
 neighbor 192.168.0.2 fall-over bfd
 neighbor 192.168.0.2 activate
 neighbor 10.4.4.2 remote-as 577
 neighbor 10.4.4.2 description PEERING to VASI Global intf
 neighbor 10.4.4.2 activate
 exit-address-family
!
ip nat switchover replication http
ip nat pool att_pool 209.165.200.225 209.165.200.225 prefix-length 16
ip nat inside source list 4 pool att_pool redundancy 1 mapping-id 100 vrf VRFA overload
ip forward-protocol nd
!
no ip http server
no ip http secure-server
ip route 203.0.113.1 255.255.255.224 10.4.4.1
ip route 192.168.0.0 255.255.0.0 10.4.4.1
ip route 209.165.200.224 255.255.255.224 10.4.4.1
ip route vrf Mgmt-intf 209.165.200.1 255.255.255.224 172.16.0.0
!
ip prefix-list VRF_Pool seq 5 permit 209.165.200.0/27
ip prefix-list p1-adv-1 seq 5 permit 209.165.200.0/27
```

```
ip prefix-list p1-exist-1 seq 5 permit 203.0.113.193/27
logging esm config
access-list 4 permit 203.0.113.193 255.255.255.224
!
control-plane
line console 0
 stopbits 1
!
line vty 0 3
 login
!
line vty 4
 password lab
 login
!
end
```

Example: Configuring Asymmetric Routing with VRF

The following example shows how to configure asymmetric routing with virtual routing and forwarding (VRF) instances:

```
Device(config)# redundancy
Device(config-red)# mode sso
Device(config-red)# application redundancy
Device(config-red-app)# group 1
Device(config-red-app-grp)# name RG1
Device(config-red-app-grp)# preempt
Device(config-red-app-grp)# priority 100 failover threshold 40
Device(config-red-app-grp)# control GigabitEthernet 1/0/3 protocol 1
Device(config-red-app-grp)# data GigabitEthernet 1/0/3
Device(config-red-app-grp)# asymmetric-routing interface GigabitEthernet 1/0/4
Device(config-red-app-grp)# asymmetric-routing always-divert enable
Device(config-red-app-grp)# exit
Device(config-red-app)# exit
Device(config-red)# exit
!
Device(config)# interface TenGigabitEthernet 2/0/0
Device(config-if)# ip vrf forwarding vrf001
Device(config-if)# ip address 10.0.0.1 255.255.255.0
Device(config-if)# ip nat inside
Device(config-if)# exit
!
Device(config)# interface TenGigabitEthernet 3/0/0
Device(config-if)# ip vrf forwarding vrf001
Device(config-if)# ip address 192.0.2.1 255.255.255.0
Device(config-if)# ip nat outside
Device(config-if)# exit
!
Device(config-if)# ip nat pool pool-vrf001 209.165.201.1 209.165.201.30 prefix-length 24
Device(config-if)# ip nat inside source list 1 pool pool-vrf001 redundancy 1 mapping-id 1
vrf vrf001 match-in-vrf overload
Device(config-if)# end
```

Additional References for Interchassis Asymmetric Routing Support for Zone-Based Firewall and NAT

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
Security commands	• Cisco IOS Security Command Reference Commands A to C • Cisco IOS Security Command Reference Commands D to L • Cisco IOS Security Command Reference Commands M to R • Cisco IOS Security Command Reference Commands S to Z
Firewall inter-chassis redundancy	"Configuring Firewall Stateful Inter-Chassis Redundancy" module
NAT inter-chassis redundancy	"Configuring Stateful Inter-Chassis Redundancy" module

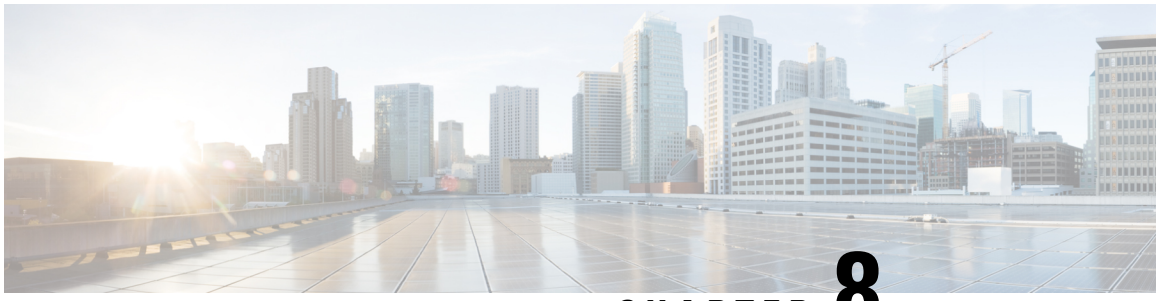
Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Interchassis Asymmetric Routing Support for Zone-Based Firewall and NAT

Table 8: Feature Information for Interchassis Asymmetric Routing Support for Zone-Based Firewall and NAT

Feature Name	Releases	Feature Information
Asymmetric Routing Enhancements for NAT44	Cisco IOS XE Release 3.16S	The Asymmetric Routing Enhancements for NAT44 feature supports asymmetric routing with CGN, ALGs, VRF, VASI and MPLS. No commands were introduced or modified.
Interchassis Asymmetric Routing Support for Zone-Based Firewall and NAT	Cisco IOS XE Release 3.5S	The Interchassis Asymmetric Routing Support for Zone-Based Firewall and NAT feature supports the forwarding of packets from a standby redundancy group to the active redundancy group for packet handling. The following commands were introduced or modified: **asymmetric-routing**, **redundancy asymmetric-routing enable**.
VRF-Aware Interchassis Asymmetric Routing Support for Zone-Based Firewalls	Cisco IOS XE Release 3.14S	Zone-based firewalls support the VRF-Aware Interchassis Asymmetric Routing feature. This feature supports MPLS. There are no configuration changes for this feature. No commands were introduced or modified.
VRF-Aware Interchassis Asymmetric Routing Support for NAT	Cisco IOS XE Release 3.14S	NAT supports the VRF-Aware Interchassis Asymmetric Routing feature. This feature supports MPLS. There are no configuration changes for this feature. No commands were introduced or modified.



CHAPTER 8

VRF-Aware NAT for WAN-WAN Topology with Symmetric Routing Box-to-Box Redundancy

In Cisco IOS XE Release 3.14S, Network Address Translation (NAT) supports the VRF-Aware NAT for WAN-WAN Topology with Symmetric Routing Box-to-Box Redundancy feature. VRF-Aware NAT for WAN-to-LAN topology is already supported in NAT.

This module describes this feature.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for VRF-Aware NAT for WAN-WAN Topology with Symmetric Routing Box-to-Box Redundancy

The following features are not supported:

- Asymmetric routing
- Cisco TrustSec
- Edge switching services
- Header Compression
- IPsec
- Lawful intercept (Intercept twice, once at active and once at standby)
- Layer 2 Tunneling Protocol (L2TP)
- Locator-ID Separation Protocol (LISP) inner packet inspection
- Port Bundle
- Stile and Ceasr
- Secure Sockets Layer (SSL) VPN
- Session Border Controller (SBC)
- If you enable NAT on the primary and backup WAN link, switchover between the primary and backup interface is not supported. NAT backup interface overlord is not supported on the following platforms:
  - ASR1000 Series Aggregation Services Routers
  - ISR4000 Series Integrated Services Routers
  - ISR1000 Series Integrated Services Routers
  - CSR1000 Series Cloud Services Routers

Information About VRF-Aware NAT for WAN-WAN Topology with Symmetric Routing Box-to-Box Redundancy

VRF-Aware Box-to-Box High Availability Support

In Cisco IOS XE Release 3.14S, Network Address Translation (NAT) supports VRF-aware box-to-box high availability in a WAN-to-WAN topology.

To support VRF-aware box-to-box high availability, NAT ties the NAT mapping with a mandatorily configured mapping ID when a redundancy group (RG) is configured. The standby device retrieves the correct locally significant VRF ID from the mapping ID after synchronization. The VRF ID is set before NAT processes or translates a packet on the active device.

The VRF-aware box-to-box high availability configuration must be the same on both active and standby devices. The VRF configuration must use the same VRF name at active and standby devices. NAT provides a hashed VRF name value in the high availability message, and sends it to active and standby devices, so that

the corresponding local VRF ID is converted at the peer device by using the VRF name hash value-to-VRF ID mapping.

Note In some cases you might experience FTP disconnection after failover in a NAT B2B scenario. To resolve this issue, quit the existing FTP connection and start a new FTP connection.

Stateful Interchassis Redundancy Overview

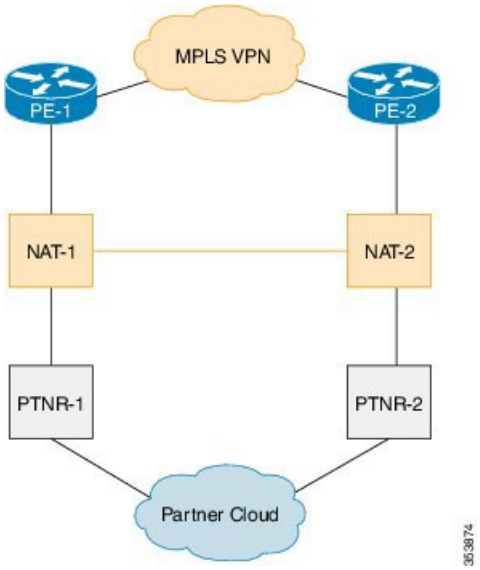
You can configure the Stateful Interchassis Redundancy feature to determine the active device from a group of devices, based on a number of failover conditions. When a failover occurs, the standby device seamlessly takes over, starts performing traffic forwarding services, and maintains a dynamic routing table.

Stateful Interchassis Redundancy Operation in NAT

You can configure pairs of devices to act as hot standbys for each other. Redundancy is configured on an interface basis. Pairs of redundant interfaces are known as redundancy groups (RGs). Redundancy occurs at an application level and does not require a complete physical failure of the interface or device for a switchover of the application to occur. When a switchover occurs, the application activity continues to run seamlessly on the redundant interface.

The figure below depicts an active/standby load-sharing scenario. The figure shows how an RG is configured for a pair of devices that has one outgoing interface. Group A on Router 1 is the active RG and Group A on Router 2 is the standby RG.

Figure 10: Stateful Interchassis Redundancy Operation in a WAN-WAN Topology



Redundant devices are joined by a configurable control link and a data synchronization link. The control link is used to communicate the status of devices. The data synchronization link is used to transfer stateful information from Network Address Translation (NAT) and the firewall and synchronize the stateful database. The pairs of redundant interfaces are configured with the same unique ID number known as the redundant interface identifier (RII).

The status of redundancy group members is determined through the use of hello messages sent over the control link. The software considers either device not responding to a hello message within a configurable amount of time to be a failure and initiates a switchover. For the software to detect a failure in milliseconds, control links run the failover protocol that is integrated with the Bidirectional Forwarding Detection (BFD) protocol. You can configure the following parameters for hello messages:

- Hello time—Interval at which hello messages are sent.
- Hold time—Amount of time before which the active or standby device is declared to be down.

The hello time defaults to 3 seconds to align with the Hot Standby Router Protocol (HSRP), and the hold time defaults to 10 seconds. You can also configure these timers in milliseconds by using the **timers hellotime msec** command.

To determine the pairs of interfaces that are affected by the switchover, you must configure a unique ID for each pair of redundant interfaces. This ID is known as the RII that is associated with the interface.

A switchover to the standby device can occur when the priority setting that is configured on each device changes. The device with the highest priority value acts as the active device. If a fault occurs on either the active or standby device, the priority of the device is decremented by a configurable amount known as the weight. If the priority of the active device falls below the priority of the standby device, a switchover occurs and the standby device becomes the active device. This default behavior can be overridden by disabling the preemption attribute for the RG. You can also configure each interface to decrease the priority when the Layer 1 state of the interface goes down. The priority that is configured overrides the default priority of an RG.

Each failure event that causes a modification of an RG priority generates a syslog entry that contains a time stamp, the RG that was affected, the previous priority, the new priority, and a description of the failure event cause.

A switchover also can occur when the priority of a device or interface falls below a configurable threshold level.

A switchover to the standby device occurs under the following circumstances:

- Power loss or a reload occurs on the active device (including reloads).
- The run-time priority of the active device goes below that of the standby device (with preempt configured).
- The run-time priority of the active device goes below that of the configured threshold.
- The redundancy group on the active device is reloaded manually. Use the **redundancy application reload group** *rg-number* command for a manual reload.

How to Configure VRF-Aware NAT for WAN-WAN Topology with Symmetric Routing Box-to-Box Redundancy

The configuration for VRF-aware box-to-box redundancy is same as the configuration for stateful interchassis redundancy. For more information, see the "Configuring Stateful Interchassis Redundancy" module in the *IP Addressing: NAT Configuration Guide*.

Configuration Examples for VRF-Aware NAT for WAN-WAN Topology with Symmetric Routing Box-to-Box Redundancy

Example: Configuring VRF-Aware NAT for WAN-WAN Topology with Symmetric Routing Box-to-Box Redundancy

The following is a sample WAN-to-WAN symmetric routing configuration:

```
vrf definition Mgmt-intf
 address-family ipv4
  exit-address-family
!
address-family ipv6
 exit-address-family
!
!
vrf definition VRFA
 rd 100:1
 route-target export 100:1
 route-target import 100:1
 address-family ipv4
  exit-address-family
!
!
no logging console
no aaa new-model
!
multilink bundle-name authenticated
!
redundancy
 mode sso
 application redundancy
  group 1
   preempt
   priority 120
   control GigabitEthernet 0/0/1 protocol 1
   data GigabitEthernet 0/0/2
!
!
!
!
ip tftp source-interface GigabitEthernet0
ip tftp blocksize 8192
!
track 1 interface GigabitEthernet 0/0/4 line-protocol
!
interface Loopback 0
 ip address 209.165.201.1 255.255.255.224
!
interface GigabitEthernet 0/0/0
 vrf forwarding VRFA
 ip address 192.168.0.1 255.255.255.248
 ip nat inside
 negotiation auto
```

```
 bfd interval 50 min_rx 50 multiplier 3
 redundancy rii 2
!
interface GigabitEthernet 0/0/1
 ip address 209.165.202.129 255.255.255.224
 negotiation auto
!
interface GigabitEthernet 0/0/2
 ip address 192.0.2.1 255.255.255.224
 negotiation auto
!
interface GigabitEthernet 0/0/3
 ip address 198.51.100.1 255.255.255.240
 negotiation auto
!
interface GigabitEthernet 0/0/4
 ip address 203.0.113.1 255.255.255.240
 negotiation auto
!
interface GigabitEthernet 0
 vrf forwarding Mgmt-intf
 ip address 172.16.0.1 255.255.0.0
 negotiation auto
!
interface vasileft 1
 vrf forwarding VRFA
 ip address 10.4.4.1 255.255.0.0
 ip nat outside
 no keepalive
!
interface vasiright 1
ip address 10.4.4.2 255.255.0.0
no keepalive
!
router mobile
!
router bgp 577
 bgp router-id 1.1.1.1
 bgp log-neighbor-changes
 neighbor 203.0.113.1 remote-as 223
 neighbor 203.0.113.1 description PEERING to PTNR neighbor 10.4.4.1 remote-as 577
 neighbor 10.4.4.1 description PEEERING to VASI VRFA interface
!
address-family ipv4
 network 203.0.113.1 mask 255.255.255.240
 network 10.4.0.0 mask 255.255.0.0
 network 209.165.200.224 mask 255.255.255.224
 neighbor 203.0.113.1 activate
 neighbor 10.4.4.1 activate
 neighbor 10.4.4.1 next-hop-self
 exit-address-family
!
address-family ipv4 vrf VRFA
 bgp router-id 4.4.4.4
 network 192.168.0.0 mask 255.255.255.248
 network 10.4.0.0 mask 255.255.0.0
 redistribute connected
 redistribute static
 neighbor 192.168.0.2 remote-as 65004
 neighbor 192.168.0.2 fall-over bfd
 neighbor 192.168.0.2 activate
 neighbor 10.4.4.2 remote-as 577
 neighbor 10.4.4.2 description PEERING to VASI Global intf
 neighbor 10.4.4.2 activate
```

```
 exit-address-family
!
ip nat switchover replication http
ip nat pool att_pool 209.165.200.225 209.165.200.225 prefix-length 16
ip nat inside source list 4 pool att_pool redundancy 1 mapping-id 100 vrf VRFA overload
ip forward-protocol nd
!
no ip http server
no ip http secure-server
ip route 203.0.113.1 255.255.255.224 10.4.4.1
ip route 192.168.0.0 255.255.0.0 10.4.4.1
ip route 209.165.200.224 255.255.255.224 10.4.4.1
ip route vrf Mgmt-intf 209.165.200.1 255.255.255.224 172.16.0.0
!
ip prefix-list VRF_Pool seq 5 permit 209.165.200.0/27
ip prefix-list p1-adv-1 seq 5 permit 209.165.200.0/27
ip prefix-list p1-exist-1 seq 5 permit 203.0.113.193/27
logging esm config
access-list 4 permit 203.0.113.193 255.255.255.224
!
control-plane
line console 0
 stopbits 1
!
line vty 0 3
 login
!
line vty 4
 password lab
 login
!
end
```

Additional References for VRF-Aware NAT for WAN-WAN Topology with Symmetric Routing Box-to-Box Redundancy

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
NAT commands	Cisco IOS IP Addressing Services Command Reference
NAT stateful interchassis redundancy	Configuring Stateful Interchassis Redundancy

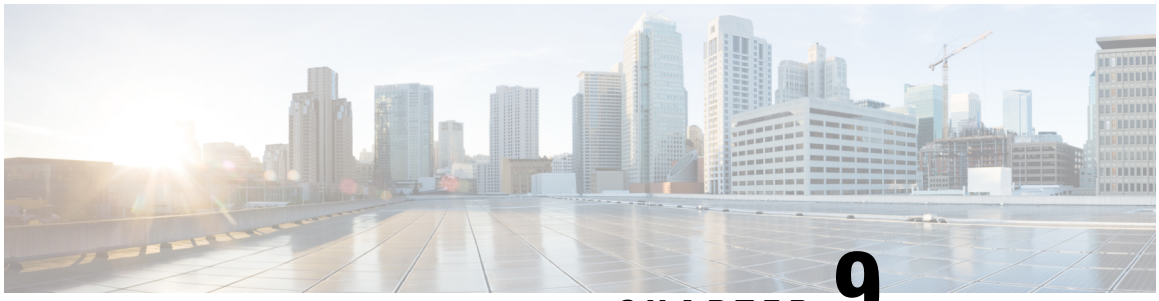
Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/support

Feature Information for VRF-Aware NAT for WAN-WAN Topology with Symmetric Routing Box-to-Box Redundancy

Table 9: Feature Information for VRF-Aware NAT for WAN-WAN Topology with Symmetric Routing Box-to-Box Redundancy

Feature Name	Releases	Feature Information
VRF-Aware NAT for WAN-WAN Topology with Symmetric Routing Box-to-Box Redundancy	Cisco IOS XE Release 3.14S	In Cisco IOS XE Release 3.14S, Network Address Translation (NAT) supports the VRF-Aware NAT for WAN-WAN Topology with Symmetric Routing Box-to-Box Redundancy feature. This feature contains the following two features: VRF-aware stateful interchassis redundancy and VRF-aware interchassis symmetric routing. No commands were introduced or modified by this feature.



CHAPTER 9

Integrating NAT with MPLS VPNs

Network Address Translation (NAT) Integration with MPLS VPNs feature allows multiple Multiprotocol Label Switching (MPLS) Virtual Private Networks (VPNs) to be configured on a single device to work together. NAT can differentiate which MPLS VPN it receives IP traffic from even if the MPLS VPNs are all using the same IP addressing scheme. This enhancement enables multiple MPLS VPN customers to share services while ensuring that each MPLS VPN is completely separate from the other.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Integrating NAT with MPLS VPNs

- Before performing the tasks in this module, you should be familiar with the concepts described in the "Configuring NAT for IP Address Conservation" module.
- All access lists required for use with the tasks in this module should be configured prior to beginning the configuration task. For information about how to configure an access list, see the *IP Access List Sequence Numbering* document at the following URL:

http://www.cisco.com/univercd/cc/td/doc/product/software/ios122s/122snwft/release/122s14/fsaclseq.htm

Note If you specify an access list to use with a NAT command, NAT does not support the commonly used **permit ip any any** command in the access list.

Restrictions for Integrating NAT with MPLS VPNs

Inside VPN to VPN with NAT is not supported.

Information About Integrating NAT with MPLS VPNs

Benefits of NAT Integration with MPLS VPNs

MPLS service providers would like to provide value-added services such as Internet connectivity, domain name servers (DNS), and voice over IP (VoIP) service to their customers. The providers require that their customers; IP addresses be different when reaching the services. Because MPLS VPN allows customers to use overlapped IP addresses in their networks, NAT must be implemented to make the services possible.

Implementation Options for Integrating Nat with MPLS VPNs

There are two approaches to implementing NAT in the MPLS VPN network. NAT can be implemented on the customer edge (CE) router, which is already supported by NAT, or it can be implemented on a provider edge (PE) router. The NAT Integration with MPLS VPNs feature enables the implementation of NAT on a PE router in an MPLS cloud.

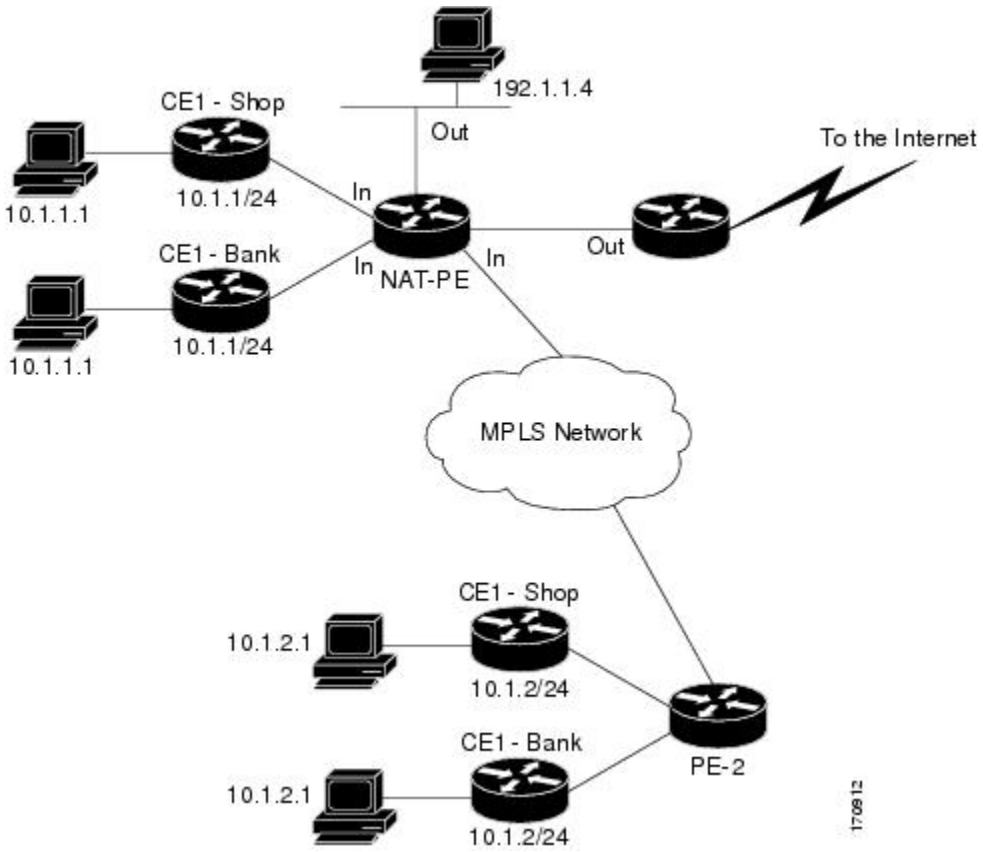
Scenarios for Implementing NAT on the PE Router

NAT could be implemented on the PE router in the following scenarios:

- Service point--Shared access can be from a generic interface or from a VPN interface.
- NAT point--NAT can be configured on the PE router that is directly connected to the shared access gateway, or on the PE router that is not directly connected to the shared access gateway.
- NAT interface--The shared access gateway interface most often is configured as the outside interface of NAT. The inside interface of NAT can be either the PE-CE interface of a VPN, the interface to the MPLS backbone, or both. The shared access gateway interface can also be configured as the inside interface.
- Routing type--Common service can be Internet connectivity or a common server. For Internet connectivity, a default route should be propagated to all the VPN customers that use the service. For common server access, a static or dynamically learned route should be propagated to the VPN customers.
- NAT configuration--NAT can have different configurations: static, dynamic, pool/interface overloading, and route-map.

The figure below shows a typical NAT integration with MPLS VPNs. The PE router connected to the internet and centralized mail service is employed to do the address translation.

Figure 11: Typical NAT Integration with MPLS VPNs



How to Integrate NAT with MPLS VPNs

Perform one or more of the following tasks depending on the type of translation you wish to configure for your network:

Configuring Inside Dynamic NAT with MPLS VPNs

Perform this task to configure your NAT PE router for dynamic translations to integrate with MPLS VPNs.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip nat pool** *name start-ip end-ip* **netmask** *netmask*
4. **ip nat** [**inside** | **outside**] **source** [**list** {*access-list-number* | *access-list-name*} | **route-map** *name*] [**interface** *type number* | **pool** *pool-name*] **vrf** *vrf-name*[**overload**]
5. Repeat Step 4 for each VPN being configured

6. **ip route vrf** *vrf-name prefix mask interface-type interface-number next-hop-address*
7. Repeat Step 6 for each VPN being configured.
8. **exit**
9. **show ip nat translations vrf** *vrf-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables higher privilege levels, such as privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ip nat pool** *name start-ip end-ip* **netmask** *netmask* **Example:** `Router(config)# ip nat pool inside 2.2.2.10 2.2.2.10 netmask 255.255.255.0`	Defines a pool of IP addresses for NAT.
Step 4	**ip nat** [**inside** | **outside**] **source** [**list** {*access-list-number* | *access-list-name*} | **route-map** *name*] [**interface** *type number* | **pool** *pool-name*] **vrf** *vrf-name*[**overload**] **Example:** `Router(config)# ip nat inside source list 1 pool mypool vrf shop overload`	Allows NAT to be configured on a particular VPN.
Step 5	Repeat Step 4 for each VPN being configured	--
Step 6	**ip route vrf** *vrf-name prefix mask interface-type interface-number next-hop-address* **Example:** `Router(config)# ip route vrf shop 0.0.0.0 0.0.0.0 ethernet 0 168.58.88.2`	Allows NAT to be configured on a particular VPN.
Step 7	Repeat Step 6 for each VPN being configured.	--
Step 8	**exit** **Example:** `Router(config)# exit`	Returns to privileged EXEC mode.

	Command or Action	Purpose
Step 9	**show ip nat translations vrf** *vrf-name* **Example:** `Router# show ip nat translations vrf shop`	(Optional) Displays the settings used by virtual routing/forwarding (VRF) table translations.

Configuring Inside Static NAT with MPLS VPNs

Perform this task to configure your NAT PE router for static translations to integrate with MPLS VPNs.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip nat inside source** {**static** {**esp** *local-ip* **interface** *type number* | *local-ip global-ip*}} [**extendable** | **mapping-id** *map-id*| **no-alias** | **no-payload** | **redundancy** *group-name* | **route-map** | **vrf** *name*]
4. Repeat Step 3 for each VPN being configured.
5. **ip route vrf** *vrf-name* **prefix** *prefix mask next-hop-address* **global**
6. Repeat Step 5 for each VPN being configured.
7. **exit**
8. **show ip nat translations vrf** *vrf-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables higher privilege levels, such as privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

Step 3	**ip nat inside source** {**static** {**esp** *local-ip* **interface** *type number*	*local-ip global-ip*}} [**extendable**	**mapping-id** *map-id*| **no-alias**	**no-payload**	**redundancy** *group-name*	**route-map**	**vrf** *name*]
							Example:
							`Router(config)#`
							`ip nat inside source static 192.168.121.113 2.2.2.1`
							`vrf shop`
								Enables inside static translation on the VRF.

Step 4	Repeat Step 3 for each VPN being configured.	--

	Command or Action	Purpose
Step 5	**ip route vrf** *vrf-name* **prefix** *prefix mask next-hop-address* **global** **Example:** `Router(config)# ip route vrf shop 0.0.0.0 0.0.0.0 168.58.88.2 global`	Allows the route to be shared by several customers.
Step 6	Repeat Step 5 for each VPN being configured.	--
Step 7	**exit** **Example:** `Router(config)# exit`	Returns to privileged EXEC mode.
Step 8	**show ip nat translations vrf** *vrf-name* **Example:** `Router# show ip nat translations vrf shop`	(Optional) Displays the settings used by VRF translations.

Configuring Outside Dynamic NAT with MPLS VPNs

Perform this step to configure your NAT PE router for dynamic outside translations to integrate with MPLS VPNs.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip nat pool outside** *global-ip local-ip* **netmask** *netmask*
4. **ip nat inside source static** *local-ip global-ip* **vrf** *vrf-name*
5. Repeat Step 4 for each VRF being configured.
6. **ip nat outside source static** *global-ip local-ip* **vrf** *vrf-name*
7. **exit**
8. **show ip nat translations vrf** *vrf-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables higher privilege levels, such as privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Router# configure terminal`	
Step 3	**ip nat pool outside** *global-ip local-ip* **netmask** *netmask* **Example:** `Router(config)# ip nat pool outside 4.4.4.1 4.4.4.254 netmask 255.255.255.00`	Allows the configured VRF to be associated with the NAT translation rule.
Step 4	**ip nat inside source static** *local-ip global-ip* **vrf** *vrf-name* **Example:** `Router(config)# ip nat inside source static 192.168.121.113 2.2.2.1 vrf shop`	Allows the route to be shared by several customers.
Step 5	Repeat Step 4 for each VRF being configured.	Allows the route to be shared by several customers.
Step 6	**ip nat outside source static** *global-ip local-ip* **vrf** *vrf-name* **Example:** `Router(config)# ip nat outside source static 168.58.88.2 4.4.4.1 vrf shop`	Enables NAT translation of the outside source address.
Step 7	**exit** **Example:** `Router(config)# exit`	Returns to privileged EXEC mode.
Step 8	**show ip nat translations vrf** *vrf-name* **Example:** `Router# show ip nat translations vrf shop`	(Optional) Displays the settings used by VRF translations.

Configuring Outside Static NAT with MPLS VPNs

Perform this task to configure your NAT PE router for static outside translations to integrate with MPLS VPNs.

SUMMARY STEPS

1. **enable**
2. **configure** {**terminal** | **memory** | **network**}
3. **ip nat pool inside** *global-ip local-ip* **netmask** *netmask*
4. Repeat Step 3 for each pool being configured.

5. **ip nat inside source list** *access-list-number* **pool** *pool-name* **vrf** *vrf-name*
6. Repeat Step 5 for each pool being configured.
7. **ip nat outside source static** *global-ip local-ip* vrf *vrf-name*
8. Repeat Step 7 for all VPNs being configured.
9. **exit**
10. **show ip nat translations vrf** *vrf-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables higher privilege levels, such as privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure** {**terminal** | **memory** | **network**} **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ip nat pool inside** *global-ip local-ip* **netmask** *netmask* **Example:** `Router(config)# ip nat pool inside1 2.2.1.1 2.2.1.254 netmask 255.255.255.0`	Allows the configured VRF to be associated with the NAT translation rule.
Step 4	Repeat Step 3 for each pool being configured.	--
Step 5	**ip nat inside source list** *access-list-number* **pool** *pool-name* **vrf** *vrf-name* **Example:** `Router(config)# ip nat inside source list 1 pool inside2 vrf shop`	Allows the route to be shared by several customers.
Step 6	Repeat Step 5 for each pool being configured.	Defines the access list.
Step 7	**ip nat outside source static** *global-ip local-ip* vrf *vrf-name* **Example:** `Router(config)# ip nat outside source static 168.58.88.2 4.4.4.1 vrf shop`	Allows the route to be shared by several customers.
Step 8	Repeat Step 7 for all VPNs being configured.	--
Step 9	**exit** **Example:**	Returns to privileged EXEC mode.

	Command or Action	Purpose
	`Router(config)# exit`	
Step 10	**show ip nat translations vrf** *vrf-name* **Example:** `Router# show ip nat translations vrf shop`	(Optional) Displays the settings used by VRF translations.

Configuration Examples for Integrating NAT with MPLS VPNs

Configuring Inside Dynamic NAT with MPLS VPNs Example

The following example shows configuring inside Dynamic NAT with MPLS VPNs.

```
!
ip nat pool inside 2.2.2.10 2.2.2.10 netmask 255.255.255.0
ip nat inside source list 1 pool inside vrf bank overload
ip nat inside source list 1 pool inside vrf park overload
ip nat inside source list 1 pool inside vrf shop overload
!
ip route vrf shop 0.0.0.0 0.0.0.0 Ethernet1/3 168.58.88.2
ip route vrf bank 0.0.0.0 0.0.0.0 Ethernet1/3 168.58.88.2
ip route vrf park 0.0.0.0 0.0.0.0 Ethernet1/3 168.58.88.2
!
access-list 1 permit 192.168.0.0 0.0.255.255
```

Configuring Inside Static NAT with MPLS VPNs Example

The following example shows configuring inside static NAT with MPLS VPNs.

```
!
ip nat inside source static 192.168.121.113 2.2.2.1 vrf shop
ip nat inside source static 192.168.122.49 2.2.2.2 vrf shop
ip nat inside source static 192.168.121.113 2.2.2.3 vrf bank
ip nat inside source static 192.168.22.49 2.2.2.4 vrf bank
ip nat inside source static 192.168.121.113 2.2.2.5 vrf park
ip nat inside source static 192.168.22.49 2.2.2.6 vrf park
ip nat inside source static 192.168.11.1 2.2.2.11 vrf shop
ip nat inside source static 192.168.11.3 2.2.2.12 vrf shop
ip nat inside source static 140.48.5.20 2.2.2.13 vrf shop
!
ip route 2.2.2.1 255.255.255.255 Ethernet1/0 192.168.121.113
ip route 2.2.2.2 255.255.255.255 Ethernet1/0 192.168.121.113
ip route 2.2.2.3 255.255.255.255 Serial2/1.1 192.168.121.113
ip route 2.2.2.4 255.255.255.255 Serial2/1.1 192.168.121.113
ip route 2.2.2.5 255.255.255.255 FastEthernet0/0 192.168.121.113
ip route 2.2.2.6 255.255.255.255 FastEthernet0/0 192.168.121.113
ip route 2.2.2.11 255.255.255.255 Ethernet1/0 192.168.121.113
ip route 2.2.2.12 255.255.255.255 Ethernet1/0 192.168.121.113
ip route 2.2.2.13 255.255.255.255 Ethernet1/0 192.168.121.113
```

Configuring Outside Dynamic NAT with MPLS VPNs Example

The following example shows configuring outside dynamic NAT with MPLS VPNs.

```
!
ip nat pool outside 4.4.4.1 4.4.4.254 netmask 255.255.255.0
ip nat inside source static 192.168.121.113 2.2.2.1 vrf shop
ip nat inside source static 192.168.122.49 2.2.2.2 vrf shop
ip nat inside source static 192.168.121.113 2.2.2.3 vrf bank
ip nat inside source static 192.168.22.49 2.2.2.4 vrf bank
ip nat inside source static 192.168.121.113 2.2.2.5 vrf park
ip nat inside source static 192.168.22.49 2.2.2.6 vrf park
ip nat outside source list 1 pool outside
!
```

Configuring Outside Static NAT with MPLS VPNs Example

The following example shows configuring outside static NAT with MPLS VPNs.

```
!
ip default-gateway 10.1.15.1
ip nat pool inside1 2.2.1.1 2.2.1.254 netmask 255.255.255.0
ip nat pool inside2 2.2.2.1 2.2.2.254 netmask 255.255.255.0
ip nat pool inside3 2.2.3.1 2.2.3.254 netmask 255.255.255.0
ip nat inside source list 1 pool inside2 vrf bank
ip nat inside source list 1 pool inside3 vrf park
ip nat inside source list 1 pool inside1 vrf shop
ip nat outside source static 168.58.88.2 4.4.4.1 vrf bank
ip nat outside source static 18.68.58.1 4.4.4.2 vrf park
ip nat outside source static 168.58.88.1 4.4.4.3 vrf shop
ip classless
ip route 192.170.10.0 255.255.255.0 Ethernet1/0 192.168.121.113
ip route 192.170.11.0 255.255.255.0 Serial2/1.1 192.168.121.113
ip route 192.170.12.0 255.255.255.0 FastEthernet0/0 192.168.121.113
ip route vrf shop 0.0.0.0 0.0.0.0 168.58.88.2 global
ip route vrf bank 0.0.0.0 0.0.0.0 168.58.88.2 global
ip route vrf park 0.0.0.0 0.0.0.0 168.58.88.2 global
no ip http server
!
access-list 1 permit 192.168.0.0 0.0.255.255
```

Where to Go Next

- To learn about Network Address Translation and configure NAT for IP address conservation, see the “Configuring NAT for IP Address Conservation” module.
- To verify, monitor, and maintain NAT, see the “Monitoring and Maintaining NAT” module.
- To use NAT with application level gateways, see the “Using Application Level Gateways with NAT” module.
- To configure NAT for high availability, see the “Configuring NAT for High Availability” module.

Additional References for Integrating NAT with MPLS VPNs

Related Documents

Related Topic	Document Title
IOS Commands	Cisco IOS Master Command List
NAT commands	Cisco IOS IP Addressing Services Command Reference

Standards and RFCs

Standard & RFC	Title
RFC 2547	*BGP/MPLS VPNs*

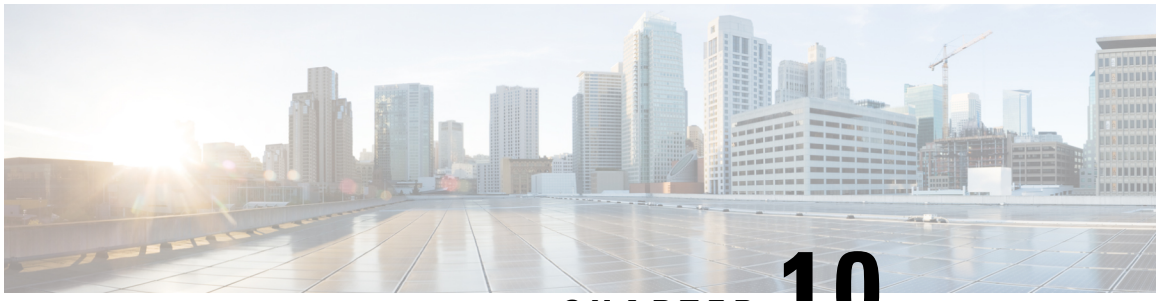
Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Integrating NAT with MPLS VPNs

Table 10: Feature Information for Integrating NAT with MPLS VPNs

Feature Name	Releases	Feature Configuration Information
Integrating NAT with MPLS VPNs	12.1(13)T 15.1(1)SY	The Integrating NAT with MPLS VPNs feature allows multiple Multiprotocol Label Switching (MPLS) VPNs to be configured on a single device to work together.



CHAPTER 10

Monitoring and Maintaining NAT

The Monitoring and Maintaining NAT feature enables the monitoring of Network Address Translation (NAT) by using translation information and statistics displays. It enables the logging of NAT translation to log and track system error messages and exceptions. The Monitoring and Maintaining NAT feature helps maintain NAT by clearing NAT translations before the timeout is expired.

This modules the Monitoring and Maintaining NAT feature.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Monitoring and Maintaining NAT

Before performing the tasks in this module, you must be familiar with the concepts described in the “Configuring NAT for IP Address Conservation” module and have NAT configured in your network.

Restrictions for Monitoring and Maintaining NAT

Syslog for Network Address Translation (NAT) is not supported.

Information About Monitoring and Maintaining NAT

NAT Display Contents

There are two basic types of IP Network Address Translation (NAT) translation information:

Translation Entries

Translation entry information includes the following:

- The protocol of the port identifying the address.
- The legitimate IP address that represents one or more inside local IP addresses to the outside world.
- The IP address assigned to a host on the inside network; probably not a legitimate address assigned by the NIC or service provider.
- The IP address of an outside host as it appears to the inside network; probably not a legitimate address assigned by the NIC or service provider.
- The IP address assigned to a host on the outside network by its owner.
- The time since the entry was created (in hours:minutes:seconds).
- The time since the entry was last used (in hours:minutes:seconds).
- Flags indicating the type of translation. Possible flags are:
  - extended—Extended translation.
  - static—Static translation.
  - destination—Rotary translation.
  - outside—Outside translation.
  - timing out—Translation will no longer be used, due to a TCP finish (FIN) or reset (RST) flag.

Statistical Information

Statistical information includes the following:

- The total number of translations active in the system. This number is incremented each time a translation is created and is decremented each time a translation is cleared or times out.
- A list of interfaces marked as outside with the **ip nat outside** command.
- A list of interfaces marked as inside with the **ip nat inside** command.
- The number of times the software does a translations table lookup and finds an entry.
- The number of times the software does a translations table lookup, fails to find an entry, and must try to create one.
- A cumulative count of translations that have expired since the router was booted.
- Information about dynamic mappings.
- Information about an inside source translation.

- The access list number being used for the translation.
- The name of the pool.
- The number of translations using this pool.
- The IP network mask being used in the pool.
- The starting IP address in the pool range.
- The ending IP address in the pool range.
- The type of pool. Possible types are generic or rotary.
- The number of addresses in the pool available for translation.
- The number of addresses being used.
- The number of failed allocations from the pool.

NAT does not support access control lists (ACLs) with the log option. The same functionality can be achieved by using one of the following options:

- By having a physical interface or virtual LAN (VLAN) with the logging option
- By using NetFlow

NAT-Forced Clear of Dynamic NAT Half-Entries

The NAT-Forced Clear of Dynamic NAT Half-Entries feature filters the display of the translation table by specifying an inside or outside address. This feature introduces the **clear ip nat translation forced** command that forcefully clears active dynamic Network Address Translation (NAT) half-entries that have child translations.

How to Monitor and Maintain NAT

Displaying NAT Translation Information

SUMMARY STEPS

1. **enable**
2. **show ip nat translations** [**verbose**]
3. **show ip nat statistics**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**show ip nat translations** [**verbose**] **Example:** `Device# show ip nat translations`	(Optional) Displays active NAT translations.
Step 3	**show ip nat statistics** **Example:** `Device# show ip nat statistics`	(Optional) Displays active NAT translation statistics.

Example:

The following is sample output from the **show ip nat translations** command:

```
Device# show ip nat translations

Pro Inside global         Inside local       Outside local        Outside global
tcp 192.168.1.1:514      192.168.2.3:53     192.168.2.22:256     192.168.2.22:256
tcp 192.168.1.1:513      192.168.2.2:53     192.168.2.22:256     192.168.2.22:256
tcp 192.168.1.1:512      192.168.2.4:53     192.168.2.22:256     192.168.2.22:256
Total number of translations: 3
```

The following is sample output from the **show ip nat translations verbose** command:

```
Device# show ip nat translations verbose

Pro Inside global        Inside local       Outside local      Outside global
tcp 192.168.1.1:514      192.168.2.3:53     192.168.2.22:256     192.168.2.22:256
         create 04/09/11 10:51:48, use 04/09/11 10:52:31, timeout: 00:01:00
         Map-Id(In):1, Mac-Address: 0000.0000.0000 Input-IDB: GigabitEthernet0/3/1
          entry-id: 0x8ef80350, use_count:1
tcp 192.168.1.1:513      192.168.2.2:53     192.168.2.22:256     192.168.2.22:256
         create 04/09/11 10:51:48, use 04/09/11 10:52:31, timeout: 00:01:00
         Map-Id(In):1, Mac-Address: 0000.0000.0000 Input-IDB: GigabitEthernet0/3/1
          entry-id: 0x8ef801b0, use_count:1
tcp 192.168.1.1:512      192.168.2.4:53     192.168.2.22:256     192.168.2.22:256
         create 04/09/11 10:51:48, use 04/09/11 10:52:31, timeout: 00:01:00
         Map-Id(In):1, Mac-Address: 0000.0000.0000 Input-IDB: GigabitEthernet0/3/1
          entry-id: 0x8ef80280, use_count:1
Total number of translations: 3
```

The following is sample output from the **show ip nat statistics** command:

```
Device# show ip nat statistics

Total active translations: 3 (0 static, 3 dynamic; 3 extended)
Outside interfaces:
GigabitEthernet0/3/0
Inside interfaces:
GigabitEthernet0/3/1
Hits: 3228980 Misses: 3
CEF Translated packets: 0, CEF Punted packets: 0
Expired translations: 0
Dynamic mappings:
-- Inside Source
[Id: 1] access-list 1 pool pool1 refcount 3
  pool pool1: netmask 255.255.255.0
  start 198.168.1.1 end 198.168.254.254
  type generic, total addresses 254, allocated 0 (0%), misses 0
  longest chain in pool: pool1's addr-hash: 0, average len 0,chains 0/256
```

```
Pool stats drop: 0 Mapping stats drop: 0
Port block alloc fail: 0
IP alias add fail: 0
Limit entry add fail: 0
```

Clearing NAT Entries Before the Timeout

By default, dynamic address translations will time out from the NAT translation table at some point. Perform this task to clear the entries before the timeout.

SUMMARY STEPS

1. **enable**
2. **clear ip nat translation inside** *global-ip local-ip* **outside** *local-ip global-ip*
3. **clear ip nat translation outside** *global-ip local-ip*
4. **clear ip nat translation** *protocol* **inside** *global-ip global-port local-ip local-port* **outside** *local-ip local-port global-ip global-port*
5. **clear ip nat translation** {* | [**forced**] | [**inside** *global-ip local-ip*] [**outside** *local-ip global-ip*]}
6. **clear ip nat translation inside** *global-ip local-ip* [**forced**]
7. **clear ip nat translation outside** *local-ip global-ip* [**forced**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear ip nat translation inside** *global-ip local-ip* **outside** *local-ip global-ip* **Example:** `Device# clear ip nat translation inside 192.168.2.209 192.168.2.95 outside 192.168.2.100 192.168.2.101`	(Optional) Clears a single dynamic half-entry containing an inside translation or both an inside and outside translation created in a dynamic configuration. • A dynamic half-entry is cleared only if it does not have any child translations.
Step 3	**clear ip nat translation outside** *global-ip local-ip* **Example:** `Device# clear ip nat translation outside 192.168.2.100 192.168.2.80`	(Optional) Clears a single dynamic half-entry containing an outside translation created in a dynamic configuration. • A dynamic half-entry is cleared only if it does not have any child translations.
Step 4	**clear ip nat translation** *protocol* **inside** *global-ip global-port local-ip local-port* **outside** *local-ip local-port global-ip global-port* **Example:** `Device # clear ip nat translation udp inside 192.168.2.209 1220 192.168.2.195 1220 outside 192.168.2.13 53 192.168.2.132 53`	(Optional) Clears a UDP translation entry.

	Command or Action	Purpose
Step 5	**clear ip nat translation** {* | [**forced**] | [**inside** *global-ip local-ip*] [**outside** *local-ip global-ip*]} **Example:** `Device# clear ip nat translation *`	(Optional) Clears either all dynamic translations (with the * or **forced** keyword), a single dynamic half-entry containing an inside translation, or a single dynamic half-entry containing an outside translation. • A single dynamic half-entry is cleared only if it does not have any child translations.
Step 6	**clear ip nat translation inside** *global-ip local-ip* [**forced**] **Example:** `Device# clear ip nat translation inside 192.168.2.209 192.168.2.195 forced`	(Optional) Forces the clearing of a single dynamic half-entry and its child translations containing an inside translation created in a dynamic configuration, with or without its corresponding outside translation. • A dynamic half-entry is always cleared, regardless of whether it has any child translations.
Step 7	**clear ip nat translation outside** *local-ip global-ip* [**forced**] **Example:** `Device# clear ip nat translation outside 192.168.2.100 192.168.2.80 forced`	(Optional) Forces the clearing of a single dynamic half-entry and its child translations containing an outside translation created in a dynamic configuration. • A dynamic half-entry is always cleared, regardless of whether it has any child translations.

Examples for Monitoring and Maintaining NAT

Example: Clearing UDP NAT Translations

The following example shows the Network Address Translation (NAT) entries before and after the UDP entry is cleared:

```
Device# show ip nat translation
Pro Inside global        Inside local         Outside local      Outside global
udp 192.168.2.20:1220    192.168.2.95:1220    192.168.2.22:53    192.168.2.20:53
tcp 192.168.2.20:11012   192.168.2.209:11012  171.69.1.220:23    192.168.2.20:23
tcp 192.168.2.20:1067    192.168.2.20:1067    192.168.2.20:23    192.168.2.20:23

Device# clear ip nat translation udp inside 192.168.2.20:1067 192.168.2.20:1067 outside
192.168.2.20:23 192.168.2.20:23
Device# show ip nat translation

Pro Inside global      Inside local        Outside local      Outside global
udp 192.168.2.20:1220  192.168.2.95:1220   192.168.2.22:53    192.168.2.20:53
tcp 192.168.2.20:11012 192.168.2.209:11012 171.69.1.220:23    192.168.2.20:23
```

Additional References for Monitoring and Maintaining NAT

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
NAT commands: complete command syntax, command mode, command history, defaults, usage guidelines, and examples	Cisco IOS IP Addressing Services Command Reference
NAT for IP address conservation	"Configuring NAT for IP Address Conservation" module

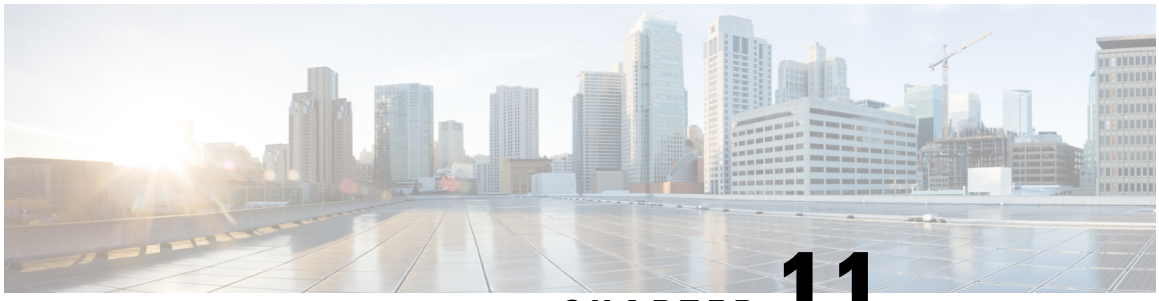
Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Monitoring and Maintaining NAT

Table 11: Feature Information for Monitoring and Maintaining NAT

Feature Name	Releases	Feature Information
NAT—Forced Clear of Dynamic NAT Half-Entries	Cisco IOS XE Release 2.4	The NAT-Forced Clear of Dynamic NAT Half-Entries feature filters the display of the translation table by specifying an inside or outside address. The following commands were introduced or modified: **clear ip nat translations forced**, **show ip nat translations**.



CHAPTER 11

Enabling NAT High-Speed Logging per VRF

The Enabling NAT High-Speed Logging Per VRF feature provides the ability to enable and disable Network Address Translation (NAT) high-speed logging (HAL) for virtual routing and forwarding (VRF) instances.

This module provides information about how to enable HSL for VRFs.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About Enabling NAT High-Speed Logging per VRF

High-Speed Logging for NAT

Network Address Translation (NAT) supports high-speed logging (HSL) for upto 4 destinations. When HSL is configured, NAT provides a log of the packets flowing through the routing devices (similar to the Version 9 NetFlow-like records) to an external collector. Records are sent for each binding (binding is the address binding between the local address and the global address to which the local address is translated) and when sessions are created and destroyed. Session records contain the full 5-tuple of information (the source IP address, destination IP address, source port, destination port, and protocol). A tuple is an ordered list of elements. NAT also sends an HSL message when a NAT pool runs out of addresses (also called *pool exhaustion*). Because the pool exhaustion messages are rate limited, each packet that hits the pool exhaustion condition does not trigger an HSL message.

The table below describes the templates for HSL bind and session create or destroy.

Table 12: Template for HSL Bind and Session Create or Destroy

Field	Format	ID	Value
Source IP address	IPv4 address	8	varies
Translated source IP address	IPv4 address	225	varies
Destination IP address	IPv4 address	12	varies
Translated destination IP address	IPv4 address	226	varies
Original source port	16-bit port	7	varies
Translated source port	16-bit port	227	varies
Original destination port	16-bit port	11	varies
Translated destination port	16-bit port	228	varies
Virtual routing and forwarding (VRF) ID	32-bit ID	234	varies
Protocol	8-bit value	4	varies
Event	8-bit value	230	0-Invalid 1-Adds event 2-Deletes event
Unix timestamp in milliseconds	64-bit value	323	varies **Note** Based on your release version, this field will be available.

The table below describes the HSL pool exhaustion templates.

Table 13: Template for HSL Pool Exhaustion

Field	Format	ID	Values
NAT pool ID	32-bit value	283	varies
NAT event	8-bit value	230	3-Pool exhaust

How to Configure Enabling NAT High-Speed Logging per VRF

Enabling High-Speed Logging of NAT Translations

You can enable or disable high-speed logging (HSL) of all Network Address Translation (NAT) translations or only translations for specific VPNs.

You must first use the **ip nat log translations flow-export v9 udp destination** command to enable HSL for all VPN and non-VPN translations. . VPN translations are also known as Virtual Routing and Forwarding (VRF) translations.

After you enable HSL for all NAT translations, you can then use the **ip nat log translations flow-export v9** *vrf-name* command to enable or disable translations for specific VPNs. When you use this command, HSL is disabled for all VPNs, except for the ones the command is explicitly enabled.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip nat log translations flow-export v9 udp destination source** *interface type interface-number*
4. **ip nat log translations flow-export v9** {*vrf-name* | **global-on**}
5. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip nat log translations flow-export v9 udp destination source** *interface type interface-number* **Example:** This example shows how to enable high-speed logging using an IPv4 address `Device(config)# ip nat log translations flow-export v9 udp destination 10.10.0.1 1020 source GigabitEthernet 0/0/0`	
Step 4	**ip nat log translations flow-export v9** {*vrf-name*	**global-on**} **Example:**	Enables or disables the high-speed logging of specific NAT VPN translations.

	Command or Action	Purpose
	`Device(config)# ip nat log translations flow-export v9 VPN-18`	
Step 5	**exit** **Example:** `Device(config)# exit`	(Optional) Exits global configuration mode and enters privileged EXEC mode.

Configuration Examples for Enabling NAT High-Speed Logging per VRF

Example: Enabling High-Speed Logging of NAT Translations

```
Device# configure terminal
Device(config)# ip nat log translations flow-export v9 udp destination 10.10.0.1 1020 source
 GigabitEthernet 0/0/0
Device(config)# ip nat log translations flow-export v9 VPN-18
Device(config)# exit
```

Additional References for Enabling NAT High-Speed Logging per VRF

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
NAT commands	Cisco IOS IP Addressing Services Command Reference

Standards and RFCs

Standard/RFC	Title

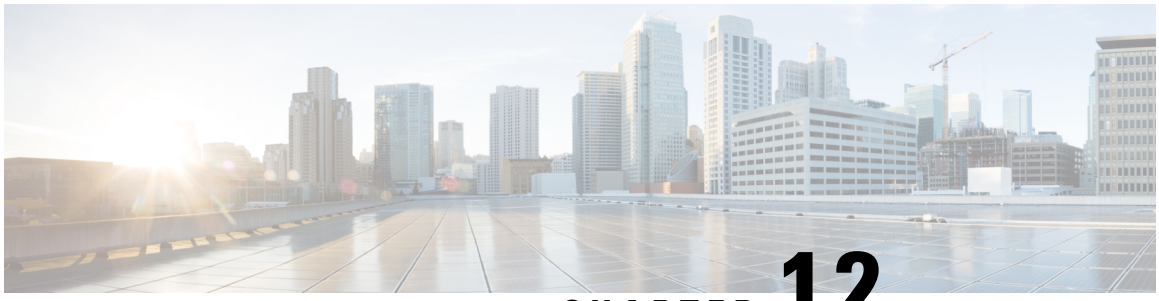
Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Enabling NAT High-Speed Logging per VRF

Table 14: Feature Information for Enabling NAT HIgh-Speed Logging per VRF

Feature Name	Releases	Feature Information
Enabling NAT High-Speed Logging per VRF	Cisco IOS XE Release 3.1S	The Enabling NAT High-Speed Logging per VRF feature provides the ability to enable and disable Network Address Translation (NAT) high-speed logging (HAL) for virtual routing and forwarding (VRF) instances. The following commands were introduced or modified: **ip nat log translations flow-export**.



CHAPTER 12

Stateless Network Address Translation 64

The Stateless Network Address Translation 64 (NAT64) feature provides a translation mechanism that translates an IPv6 packet into an IPv4 packet and vice versa. The translation involves parsing the entire IPv6 header, including the extension headers, and extracting the relevant information and translating it into an IPv4 header.This processing happens on a per-packet basis on the interfaces that are configured for Stateless NAT64 translation.

The Stateless NAT64 translator enables native IPv6 or IPv4 communication and facilitates coexistence of IPv4 and IPv6 networks.

The Stateless NAT64 translator does not maintain any state information in the datapath.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for Stateless Network Address Translation 64

The following restrictions apply to the Stateless NAT64 feature:

- Only valid IPv4-translatable addresses can be used for stateless translation.
- Multicast is not supported.

- Applications without a corresponding application layer gateway (ALG) may not work properly with the Stateless NAT64 translator.
- The translation of IPv4 options, IPv6 routing headers, hop-by-hop extension headers, destination option headers, and source routing headers are not supported.
- Fragmented IPv4 UDP packets that do not contain a UDP checksum are not translated.
- IPv6 packets with zero UDP checksum are not translated.
- Both NAT44 (static, dynamic, and Port Address Translation [PAT]) configurations and Stateless NAT64 configuration are not supported on the same interface.

Information About Stateless Network Address Translation 64

Fragmentation of IP Datagrams in IPv6 and IPv4 Networks

In IPv4 networks, any intermediate router can do the fragmentation of an IP datagram. However, in IPv6 networks, fragmentation can be done only by the originating IPv6 host. Because fragmentation in IPv6 networks is done by the IPv6 hosts, the path maximum transmission unit (PMTU) discovery should also be done by the IPv6 hosts. However, a PMTU discovery is not possible across an IPv4 network where the routers are allowed to fragment the packets. In IPv4 networks, a Stateless NAT64 translator is used to fragment the IPv6 datagram and set the Don't Fragment (DF) bits in the IPv4 header. Similarly, the translator can add the fragment header to the IPv6 packet if an IPv4 fragment is received.

Translation of ICMP for Stateless NAT64 Translation

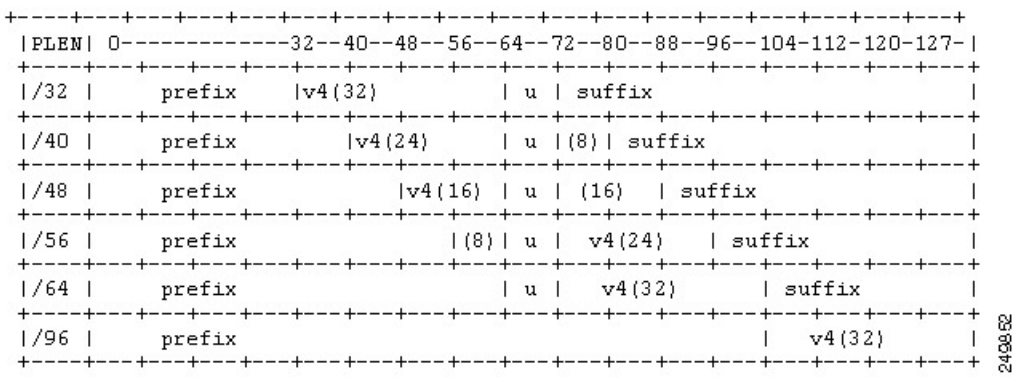
The IETF draft on the IP/ICMP translation algorithm describes the ICMP types or codes that should be translated between IPv4 and IPv6. ICMP errors embed the actual IP header and the transport header. Because the ICMP errors are embedded in the IP header, the IP header is not translated properly. For ICMP error packets, Stateless NAT64 translation should be applied twice: once for the outer header, and once again for the embedded header.

IPv4-Translatable IPv6 Address

IPv4-translatable IPv6 addresses are IPv6 addresses assigned to the IPv6 nodes for use with stateless translation. IPv4-translatable addresses consist of a variable-length prefix, an embedded IPv4 address, fixed universal bits (u-bits), and in some cases a suffix. IPv4-embedded IPv6 addresses are IPv6 addresses in which 32 bits contain an IPv4 address. This format is the same for both IPv4-converted and IPv4-translatable IPv6 addresses.

The figure below shows an IPv4-translatable IPv6 address format with several different prefixes and embedded IPv4 address positions.

Figure 12: IPv4-Translatable IPv6 Address Format



Prefixes Format

A set of bits at the start of an IPv6 address is called the format prefix. Prefix length is a decimal value that specifies how many of the leftmost contiguous bits of an address comprise the prefix.

An embedded IPv4 address is used to construct IPv4 addresses from the IPv6 packet. The Stateless NAT64 translator has to derive the IPv4 addresses that are embedded in the IPv6-translatable address by using the prefix length. The translator has to construct an IPv6-translatable address based on the prefix and prefix length and embed the IPv4 address based on the algorithm.

The prefix lengths of 32, 40, 48, 56, 64, or 96 are supported for Stateless NAT64 translation. The Well Known Prefix (WKP) is not supported. When traffic flows from the IPv4-to-IPv6 direction, either a WKP or a configured prefix can be added only in stateful translation.

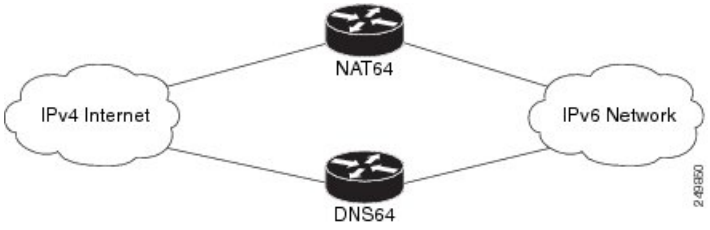
Supported Stateless NAT64 Scenarios

The following scenarios are supported by the Cisco IOS Stateless NAT64 feature and are described in this section:

- Scenario 1--an IPv6 network to the IPv4 Internet
- Scenario 2--the IPv4 Internet to an IPv6 network
- Scenario 5--an IPv6 network to an IPv4 network
- Scenario 6--an IPv4 network to an IPv6 network

The figure below shows stateless translation for scenarios 1 and 2. An IPv6-only network communicates with the IPv4 Internet.

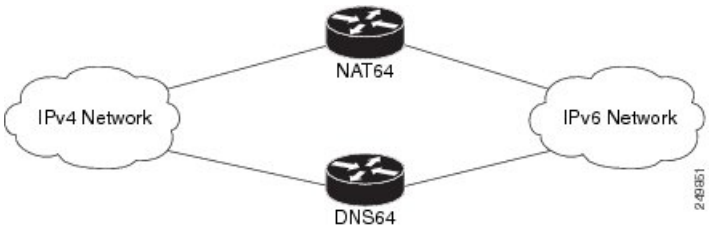
Figure 13: Stateless Translation for Scenarios 1 and 2



Scenario 1 is an IPv6 initiated connection and scenario 2 is an IPv4 initiated connection. Stateless NAT64 translates these two scenarios only if the IPv6 addresses are IPv4 translatable. In these two scenarios, the Stateless NAT64 feature does not help with IPv4 address depletion, because each IPv6 host that communicates with the IPv4 Internet is a globally routable IPv4 address. This consumption is similar to the IPv4 consumption rate as a dual-stack. The savings, however, is that the internal network is 100 percent IPv6, which eases management (Access Control Lists, routing tables), and IPv4 exists only at the edge where the Stateless translators live.

The figure below shows stateless translation for scenarios 5 and 6. The IPv4 network and IPv6 network are within the same organization.

Figure 14: Stateless Translation for Scenarios 5 and 6



The IPv4 addresses used are either public IPv4 addresses or RFC 1918 addresses. The IPv6 addresses used are either public IPv6 addresses or Unique Local Addresses (ULAs).

Both these scenarios consist of an IPv6 network that communicates with an IPv4 network. Scenario 5 is an IPv6 initiated connection and scenario 6 is an IPv4 initiated connection. The IPv4 and IPv6 addresses may not be public addresses. These scenarios are similar to the scenarios 1 and 2. The Stateless NAT64 feature supports these scenarios if the IPv6 addresses are IPv4 translatable.

Multiple Prefixes Support for Stateless NAT64 Translation

Network topologies that use the same IPv6 prefix for source and destination addresses may not handle routing correctly and may be difficult to troubleshoot. The Stateless NAT64 feature addresses these challenges in Cisco IOS XE Release 3.3S and later releases through the support of multiple prefixes for stateless translation. The entire IPv4 Internet is represented as using a different prefix from the one used for the IPv6 network.

How to Configure Stateless Network Address Translation 64

Configuring a Routing Network for Stateless NAT64 Communication

Perform this task to configure and verify a routing network for Stateless NAT64 communication. You can configure stateless NAT64 along with your NAT configuration: static, dynamic, or overload.

Before you begin

- An IPv6 address assigned to any host in the network should have a valid IPv4-translatable address and vice versa.
- You should enable the **ipv6 unicast-routing** command for this configuration to work.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 unicast-routing**
4. **interface** *type number*
5. **description** *string*
6. **ipv6 enable**
7. **ipv6 address** {*ipv6-address*/*prefix-length* | *prefix-name sub-bits*/*prefix-length*}
8. **nat64 enable**
9. **exit**
10. **interface** *type number*
11. **description** *string*
12. **ip address** *ip-address mask*
13. **nat64 enable**
14. **exit**
15. **nat64 prefix stateless** *ipv6-prefix*/*length*
16. **nat64 route** *ipv4-prefix*/*mask interface-type interface-number*
17. **ipv6 route** *ipv4-prefix*/*length interface-type interface-number*
18. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 unicast-routing** **Example:** `Device(config)# ipv6 unicast-routing`	Enables the forwarding of IPv6 unicast datagrams.
Step 4	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 0/0/0`	Configures an interface type and enters interface configuration mode.
Step 5	**description** *string* **Example:**	Adds a description to an interface configuration.

	Command or Action	Purpose
	`Device(config-if)# description interface facing ipv6`	
Step 6	**ipv6 enable** **Example:** `Device(config-if)# ipv6 enable`	Enables IPv6 processing on an interface.
Step 7	**ipv6 address** {*ipv6-address*/*prefix-length* | *prefix-name sub-bits*/*prefix-length*} **Example:** `Device(config-if)# ipv6 address 2001:DB8::1/128`	Configures an IPv6 address based on an IPv6 general prefix and enables IPv6 processing on an interface.
Step 8	**nat64 enable** **Example:** `Device(config-if)# nat64 enable`	Enables Stateless NAT64 translation on an IPv6 interface.
Step 9	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 10	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 1/2/0`	Configures an interface type and enters interface configuration mode.
Step 11	**description** *string* **Example:** `Device(config-if)# description interface facing ipv4`	Adds a description to an interface configuration.
Step 12	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 198.51.100.1 255.255.255.0`	Configures an IPv4 address for an interface.
Step 13	**nat64 enable** **Example:** `Device(config-if)# nat64 enable`	Enables Stateless NAT64 translation on an IPv4 interface.
Step 14	**exit** **Example:**	Exits interface configuration mode and returns to global configuration mode.

	Command or Action	Purpose
	`Device(config-if)# exit`	
Step 15	**nat64 prefix stateless** *ipv6-prefix/length* **Example:** `Device(config)# nat64 prefix stateless 2001:0db8:0:1::/96`	Defines the Stateless NAT64 prefix to be added to the IPv4 hosts to translate the IPv4 address into an IPv6 address. • The command also identifies the prefix that must be used to create the IPv4-translatable addresses for the IPv6 hosts.
Step 16	**nat64 route** *ipv4-prefix/mask interface-type interface-number* **Example:** `Device(config)# nat64 route 203.0.113.0/24 gigabitethernet 0/0/0`	Routes the IPv4 traffic towards the correct IPv6 interface.
Step 17	**ipv6 route** *ipv4-prefix/length interface-type interface-number* **Example:** `Device(config)# ipv6 route 2001:DB8:0:1::CB00:7100/120 gigabitethernet 0/0/0`	Routes the translated packets to the IPv4 address. • You must configure the **ipv6 route** command if your network is not running IPv6 routing protocols.
Step 18	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Configuring Multiple Prefixes for Stateless NAT64 Translation

Perform this task to configure multiple prefixes for Stateless NAT64 translation.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 unicast-routing**
4. **interface** *type number*
5. **ipv6 address** {*ipv6-address/prefix-length* | *prefix-name sub-bits/prefix-length*}
6. **ipv6 enable**
7. **nat64 enable**
8. **nat64 prefix stateless v6v4** *ipv6-prefix/length*
9. **exit**
10. **interface** *type number*
11. **ip address** *ip-address mask*
12. **negotiation auto**
13. **nat64 enable**

14. **exit**
15. **nat64 prefix stateless v4v6** *ipv6-prefix*/*length*
16. **nat64 route** *ipv4-prefix*/*mask interface-type interface-number*
17. **ipv6 route** *ipv6-prefix*/*length interface-type interface-number*
18. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 unicast-routing** **Example:** `Router(config)# ipv6 unicast-routing`	Enables the forwarding of IPv6 unicast datagrams.
Step 4	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 0/0/0`	Configures an interface type and enters interface configuration mode.
Step 5	**ipv6 address** {*ipv6-address*/*prefix-length* | *prefix-name sub-bits*/*prefix-length*} **Example:** `Device(config-if)# ipv6 address 2001:DB8::1/128`	Configures an IPv6 address based on an IPv6 general prefix and enables IPv6 processing on an interface.
Step 6	**ipv6 enable** **Example:** `Device(config-if)# ipv6 enable`	Enables IPv6 processing on an interface.
Step 7	**nat64 enable** **Example:** `Device(config-if)# nat64 enable`	Enables Stateless NAT64 translation on an IPv6 interface.
Step 8	**nat64 prefix stateless v6v4** *ipv6-prefix*/*length* **Example:**	Maps an IPv6 address to an IPv4 host for Stateless NAT 64 translation.

	Command or Action	Purpose
	`Device(config-if)# nat64 prefix stateless v6v4 2001:0db8:0:1::/96`	• The NAT64 prefix in the command is the same as the prefix of the source packet that is coming from the IPv6-to-IPv4 direction.
Step 9	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 10	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 1/2/0`	Configures an interface type and enters interface configuration mode.
Step 11	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 203.0.113.1 255.255.255.0`	Configures an IPv4 address for an interface.
Step 12	**negotiation auto** **Example:** `Device(config-if)# negotiation auto`	Enables the autonegotiation protocol to configure the speed, duplex, and automatic flow control on an interface.
Step 13	**nat64 enable** **Example:** `Router(config-if)# nat64 enable`	Enables Stateless NAT64 translation on an IPv4 interface.
Step 14	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 15	**nat64 prefix stateless v4v6** *ipv6-prefix*/*length* **Example:** `Device(config)# nat64 prefix stateless v4v6 2001:DB8:2::/96`	Maps an IPv4 address to an IPv6 host for Stateless NAT 64 translation. • This command identifies the prefix that creates the IPv4-translatable addresses for the IPv6 hosts.
Step 16	**nat64 route** *ipv4-prefix*/*mask interface-type interface-number* **Example:** `Device(config)# nat64 route 203.0.113.0/24 gigabitethernet 0/0/0`	Routes the IPv4 traffic towards the correct IPv6 interface.

	Command or Action	Purpose
Step 17	**ipv6 route** *ipv6-prefix/length interface-type interface-number* **Example:** `Device(config)# ipv6 route 2001:DB8:0:1::CB00:7100/120 gigabitethernet 0/0/0`	Routes the translated packets to the IPv4 address. • You must configure the **ipv6 route** command if your network is not running IPv6 routing protocols.
Step 18	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Monitoring and Maintaining the Stateless NAT64 Routing Network

Perform this task to verify and monitor the Stateless NAT64 routing network. In the privileged EXEC mode, you can enter the commands in any order.

SUMMARY STEPS

1. **show nat64 statistics**
2. **show ipv6 route**
3. **show ip route**
4. **debug nat64** {**all** | **ha** {**all** | **info** | **trace** | **warn**} | **id-manager** | **info** | **issu** {**all** | **message** | **trace**} | **memory** | **statistics** | **trace** | **warn**}
5. **ping** [*protocol* [**tag**]] {*host-name* | *system-address*}

DETAILED STEPS

Step 1 show nat64 statistics

This command displays the global and interface-specific statistics of the packets that are translated and dropped.

Example:

```
Device# show nat64 statistics

NAT64 Statistics
Global Stats:
   Packets translated (IPv4 -> IPv6): 21
   Packets translated (IPv6 -> IPv4): 15
GigabitEthernet0/0/1 (IPv4 configured, IPv6 configured):
   Packets translated (IPv4 -> IPv6): 5
   Packets translated (IPv6 -> IPv4): 0
   Packets dropped: 0
GigabitEthernet1/2/0 (IPv4 configured, IPv6 configured):
   Packets translated (IPv4 -> IPv6): 0
   Packets translated (IPv6 -> IPv4): 5
   Packets dropped: 0
```

Step 2 **show ipv6 route**

This command displays the configured stateless prefix and the specific route for the IPv4 embedded IPv6 address pointing toward the IPv6 side.

Example:

```
Device# show ipv6 route

IPv6 Routing Table - default - 6 entries
Codes: C - Connected, L - Local, S - Static, U - Per-user Static route
B - BGP, R - RIP, I1 - ISIS L1, I2 - ISIS L2
IA - ISIS interarea, IS - ISIS summary, D - EIGRP, EX - EIGRP external
ND - Neighbor Discovery
O - OSPF Intra, OI - OSPF Inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
LC  2001::1/128 [0/0] via FastEthernet0/3/4, receive
S   2001::1B01:10A/128 [1/0] via FastEthernet0/3/4, directly connected
S   3001::/96 [1/0] via ::42, NVI0
S   3001::1E1E:2/128 [1/0] via FastEthernet0/3/0, directly connected
LC  3001::C0A8:64D5/128 [0/0] via FastEthernet0/3/0, receive
L   FF00::/8 [0/0] via Null0, receive
```

Step 3 **show ip route**

This command displays the IPv4 addresses in the Internet that have reached the IPv4 side.

Example:

```
Device# show ip route

Codes: R - RIP derived, O - OSPF derived,
       C - connected, S - static, B - BGP derived,
       * - candidate default route, IA - OSPF inter area route,
       i - IS-IS derived, ia - IS-IS, U - per-user static route,
       o - on-demand routing, M - mobile, P - periodic downloaded static route,
       D - EIGRP, EX - EIGRP external, E1 - OSPF external type 1 route,
       E2 - OSPF external type 2 route, N1 - OSPF NSSA external type 1 route,
       N2 - OSPF NSSA external type 2 route
Gateway of last resort is 10.119.254.240 to network 10.140.0.0
O E2 10.110.0.0 [160/5] via 10.119.254.6, 0:01:00, Ethernet2
E    10.67.10.0 [200/128] via 10.119.254.244, 0:02:22, Ethernet2
O E2 10.68.132.0 [160/5] via 10.119.254.6, 0:00:59, Ethernet2
O E2 10.130.0.0 [160/5] via 10.119.254.6, 0:00:59, Ethernet2
E    10.128.0.0 [200/128] via 10.119.254.244, 0:02:22, Ethernet2
E    10.129.0.0 [200/129] via 10.119.254.240, 0:02:22, Ethernet2
E    10.65.129.0 [200/128] via 10.119.254.244, 0:02:22, Ethernet2
E    10.10.0.0 [200/128] via 10.119.254.244, 0:02:22, Ethernet2
E    10.75.139.0 [200/129] via 10.119.254.240, 0:02:23, Ethernet2
E    10.16.208.0 [200/128] via 10.119.254.244, 0:02:22, Ethernet2
E    10.84.148.0 [200/129] via 10.119.254.240, 0:02:23, Ethernet2
E    10.31.223.0 [200/128] via 10.119.254.244, 0:02:22, Ethernet2
E    10.44.236.0 [200/129] via 10.119.254.240, 0:02:23, Ethernet2
E    10.141.0.0 [200/129] via 10.119.254.240, 0:02:22, Ethernet2
E    10.140.0.0 [200/129] via 10.119.254.240, 0:02:23, Ethernet2
IPv6 Routing Table - default - 6 entries
```

Step 4 **debug nat64** {**all** | **ha** {**all** | **info** | **trace** | **warn**} | **id-manager** | **info** | **issu** {**all** | **message** | **trace**} | **memory** | **statistics** | **trace** | **warn**}

This command enables Stateless NAT64 debugging.

Example:

```
Device# debug nat64 statistics
```

Step 5 **ping** [*protocol* [**tag**]] {*host-name* | *system-address*}

The following is a sample packet capture from the IPv6 side when you specify the **ping 198.168.0.2** command after you configure the **nat64 enable** command on both the IPv4 and IPv6 interfaces:

Example:

```
Device# ping 198.168.0.2

Time             Source            Destination         Protocol       Info
1 0.000000       2001::c6a7:2      2001::c6a8:2        ICMPv6         Echo request
Frame 1: 118 bytes on wire (944 bits), 118 bytes captured (944 bits)
  Arrival Time: Oct 8, 2010 11:54:06.408354000 India Standard Time
  Epoch Time: 1286519046.408354000 seconds
  [Time delta from previous captured frame: 0.000000000 seconds]
  [Time delta from previous displayed frame: 0.000000000 seconds]
  [Time since reference or first frame: 0.000000000 seconds]
  Frame Number: 1
  Frame Length: 118 bytes (944 bits)
  Capture Length: 118 bytes (944 bits)
  [Frame is marked: False]
  [Frame is ignored: False]
  [Protocol in frame: eth:1pv6:icmpv6: data]
Ethernet II, Src:Cisco_c3:64:94 (00:22:64:c3:64:94), Dst: Cisco_23:f2:30 (00:1f:6c:23:f2:30)
  Destination: Cisco_23:f2:30 (00:1f:6c:23:f2:30)
     Address: Cisco_23:f2:30 (00:1f:6c:23:f2:30)
     .... ...0 .... .... .... .... = IG bit: Individual address (unicast)
     .... ...0 .... .... .... .... = LG bit: Globally unique address (factory default)
  Source: Cisco_c3:64:94 (00:22:64:c3:64:94)
    Address: Cisco_c3:64:94 (00:22:64:c3:64:94)
     .... ...0 .... .... .... .... = IG bit: Individual address (unicast)
     .... ...0 .... .... .... .... = LG bit: Globally unique address (factory default)
  Type: IPv6 (0x86dd)
Internet Protocol Version 6, src: 2001::c6a7:2 (2001::c6a7:2), Dst: 2001::c6a8:2 (2001::c6a8:2)
   0110 .... = Version: 6
     [0110 .... = This field makes the filter "ip.version ==6" possible:: 6]
  .... 0000 0000 ... .... .... .... .... = Traffic class: 0x00000000
    .... 0000 00.. .... .... .... .... .... = Differentiated Services Field: Default (0x00000000)
    .... .... ..0. .... .... .... ... .... = ECN-Capable Transport (ECT): Not set
  .... .... .... 0000 0000 0000 0000 0000 = Flowlabel: 0x00000000
  Payload length: 64
  Next header: 64
  Hop limit: 64
  Source: 2001::c6a7:2 (2001::c6a7:2)
  [Source Teredo Server IPv4: 0.0.0.0 (0.0.0.0)]
  [Source Teredo Port: 6535]
  [Source Teredo Client IPv4: 198.51.100.1 (198.51.100.1)]
  Destination: 2001:c6a8:2 (2001::c6a8:2)
  [Destination Teredo Server IPv4: 0.0.0.0 {0.0.0.0)]
  [Destination Teredo Port: 65535]
  [Destination Teredo Client IPv4: 198.51.100.2 {198.51.100.2)]
Internet Control Message Protocol v6
  Type: 128 (Echo request)
  Code: 0 (Should always be zero)
  Checksum: 0xaed2 [correct]
  ID: 0x5018
  Sequence: 0x0000
  Data (56 bytes)
```

```
Data: 069ae4c0d3b060008090a0b0c0d0e0f1011121314151617...
[Length: 57]
```

Configuration Examples for Stateless Network Address Translation 64

Example Configuring a Routing Network for Stateless NAT64 Translation

The following example shows how to configure a routing network for Stateless NAT64 translation:

```
ipv6 unicast-routing
!
interface gigabitethernet 0/0/0
 description interface facing ipv6
 ipv6 enable
 ipv6 address 2001:DB8::1/128
 nat64 enable
!

interface gigabitethernet 1/2/0
 description interface facing ipv4
 ip address 198.51.100.1 255.255.255.0
 nat64 enable
!

nat64 prefix stateless 2001:0db8:0:1::/96
nat64 route 203.0.113.0/24 gigabitethernet 0/0/0
ipv6 route 2001:DB8:0:1::CB00:7100/120 gigabitethernet 0/0/0
```

Example: Configuring Multiple Prefixes for Stateless NAT64 Translation

```
ipv6 unicast-routing
!
interface gigabitethernet 0/0/0
 ipv6 address 2001:DB8::1/128
 ipv6 enable
 nat64 enable
 nat64 prefix stateless v6v4 2001:0db8:0:1::/96
!
interface gigabitethernet 1/2/0
 ip address 198.51.100.1 255.255.255.0
 negotiation auto
 nat64 enable
!
nat64 prefix stateless v4v6 2001:DB8:2::/96
nat64 route 203.0.113.0/24 gigabitethernet 0/0/0
ipv6 route 2001:DB8:0:1::CB00:7100/120 gigabitethernet 0/0/0
```

Additional References for Stateless Network Address Translation 64

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
NAT commands	Cisco IOS IP Addressing Services Command Reference

Standards and RFCs

Standard/RFC	Document Title
RFC 6052	IPv6 Addressing of IPv4/IPv6 Translators
RFC 6144	Framework for IPv4/IPv6 Translation
RFC 6145	IP/ICMP Translation Algorithm

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Stateless Network Address Translation 64

Table 15: Feature Information for Stateless Network Address Translation 64

Feature Name	Releases	Feature Information
Stateless Network Address Translation 64	Cisco IOS XE Release 3.2S Cisco IOS XE Release 3.10S	The Stateless Network Address Translation 64 feature provides a translation mechanism that translates an IPv6 packet into an IPv4 packet and vice versa. The translation involves parsing the entire IPv6 header, including the extension headers, and extracting the relevant information and translating it into an IPv4 header. Similarly, the IPv4 header is parsed in its entirety, including the IPv4 options, to construct an IPv6 header. This processing happens on a per-packet basis on the interfaces that are configured for Stateless NAT64 translation. The following commands were introduced or modified: **clear nat64 ha statistics**, **clear nat64 statistics**, **debug nat64**, **nat64 enable**, **nat64 prefix**, **nat64 route**, **show nat64 adjacency**, **show nat64 ha status**, **show nat64 prefix stateless**, **show nat64 routes**, and **show nat64 statistics**. In Cisco IOS XE Release 3.10S, support was added for the Cisco ISR 4400 Series Routers.

Glossary

ALG—application-layer gateway or application-level gateway.

FP—Forward Processor.

IPv4-converted address—IPv6 addresses used to represent the IPv4 hosts. These have an explicit mapping relationship to the IPv4 addresses. This relationship is self-described by mapping the IPv4 address in the IPv6 address. Both stateless and stateful translators use IPv4-converted IPv6 addresses to represent the IPv4 hosts.

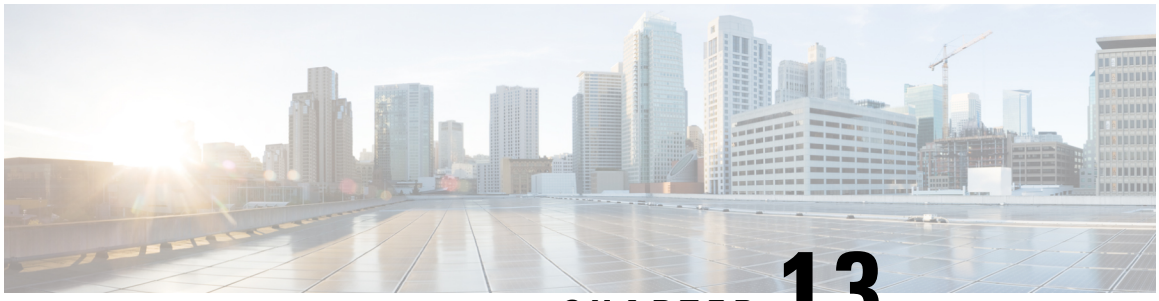
IPv6-converted address—IPv6 addresses that are assigned to the IPv6 hosts for the stateless translator. These IPv6-converted addresses have an explicit mapping relationship to the IPv4 addresses. This relationship is self-described by mapping the IPv4 address in the IPv6 address. The stateless translator uses the corresponding IPv4 addresses to represent the IPv6 hosts. The stateful translator does not use IPv6-converted addresses, because the IPv6 hosts are represented by the IPv4 address pool in the translator via dynamic states.

NAT—Network Address Translation.

RP—Route Processor.

stateful translation—In stateful translation a per-flow state is created when the first packet in a flow is received. A translation algorithm is said to be stateful if the transmission or reception of a packet creates or modifies a data structure in the relevant network element. Stateful translation allows the use of multiple translators interchangeably and also some level of scalability. Stateful translation is defined to enable the IPv6 clients and peers without mapped IPv4 addresses to connect to the IPv4-only servers and peers.

stateless translation—A translation algorithm that is not stateful is called stateless. A stateless translation requires configuring a static translation table, or may derive information algorithmically from the messages it is translating. Stateless translation requires less computational overhead than stateful translation. It also requires less memory to maintain the state, because the translation tables and the associated methods and processes exist in a stateful algorithm and do not exist in a stateless one. Stateless translation enables the IPv4-only clients and peers to initiate connections to the IPv6-only servers or peers that are equipped with IPv4-embedded IPv6 addresses. It also enables scalable coordination of IPv4-only stub networks or ISP IPv6-only networks. Because the source port in an IPv6-to-IPv4 translation may have to be changed to provide adequate flow identification, the source port in the IPv4-to-IPv6 direction need not be changed.



CHAPTER 13

Stateful Network Address Translation 64

The Stateful Network Address Translation 64 feature provides a translation mechanism that translates IPv6 packets into IPv4 packets and vice versa. The stateful NAT64 translator algorithmically translates the IPv4 addresses of IPv4 hosts to and from IPv6 addresses by using the configured stateful prefix. In a similar manner, the IPv6 addresses of IPv6 hosts are translated to and from IPv4 addresses through Network Address Translation (NAT). Stateful Network Address Translation 64 (NAT64) also translates protocols and IP addresses. The Stateful NAT64 translator enables native IPv6 or IPv4 communication and facilitates coexistence of IPv4 and IPv6 networks.

This document explains how Stateful NAT64 works and how to configure your network for Stateful NAT64 translation.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Configuring Stateful Network Address Translation 64

- For Domain Name System (DNS) traffic to work, you must have a separate working installation of DNS64.

Restrictions for Configuring Stateful Network Address Translation 64

- Applications without a corresponding application-level gateway (ALG) may not work properly with the Stateful NAT64 translator.
- IP Multicast is not supported.
- The translation of IPv4 options, IPv6 routing headers, hop-by-hop extension headers, destination option headers, and source routing headers is not supported.
- Virtual routing and forwarding (VRF)-aware NAT64 is not supported.
- When traffic flows from IPv6 to IPv4, the destination IP address that you have configured must match a stateful prefix to prevent hairpinning loops. However, the source IP address (source address of the IPv6 host) must not match the stateful prefix. If the source IP address matches the stateful prefix, packets are dropped.

  Hairpinning allows two endpoints inside Network Address Translation (NAT) to communicate with each other, even when the endpoints use only each other's external IP addresses and ports for communication.

- Only TCP and UDP Layer 4 protocols are supported for header translation.
- Routemaps are not supported.
- Application-level gateways (ALGs) FTP and ICMP are not supported.
- In the absence of a pre-existing state in NAT 64, stateful translation only supports IPv6-initiated sessions.
- If a static mapping host-binding entry exists for an IPv6 host, the IPv4 nodes can initiate communication. In dynamic mapping, IPv4 nodes can initiate communication only if a host-binding entry is created for the IPv6 host through a previously established connection to the same or a different IPv4 host.

  Dynamic mapping rules that use Port-Address Translation (PAT), host-binding entries cannot be created because IPv4-initiated communication not possible through PAT.

- Both NAT44 (static, dynamic and PAT) configuration and stateful NAT64 configuration are not supported on the same interface.

Information About Stateful Network Address Translation 64

Stateful Network Address Translation 64

The Stateful NAT64 feature provides a translation mechanism that translates IPv6 packets into IPv4 packets and vice versa.

Stateful NAT64 supports Internet Control Message Protocol (ICMP), TCP, and UDP traffic. Packets that are generated in an IPv6 network and are destined for an IPv4 network are routed within the IPv6 network towards the Stateful NAT64 translator. Stateful NAT64 translates the packets and forwards them as IPv4 packets through the IPv4 network. The process is reversed for traffic that is generated by hosts connected to the IPv4 network and destined for an IPv6 receiver.

The Stateful NAT64 translation is not symmetric, because the IPv6 address space is larger than the IPv4 address space and a one-to-one address mapping is not possible. Before it can perform an IPv6 to an IPv4 translation, Stateful NAT64 requires a state that binds the IPv6 address and the TCP/UDP port to the IPv4 address. The binding state is either statically configured or dynamically created when the first packet that flows from the IPv6 network to the IPv4 network is translated. After the binding state is created, packets flowing in both directions are translated. In dynamic binding, Stateful NAT64 supports communication initiated by the IPv6-only node toward an IPv4-only node. Static binding supports communication initiated by an IPv4-only node to an IPv6-only node and vice versa. Stateful NAT64 with NAT overload or Port Address Translation (PAT) provides a 1:*n* mapping between IPv4 and IPv6 addresses.

When an IPv6 node initiates traffic through Stateful NAT64, and the incoming packet does not have an existing state and the following events happen:

- The source IPv6 address (and the source port) is associated with an IPv4 configured pool address (and port, based on the configuration).
- The destination IPv6 address is translated mechanically based on the BEHAVE translation draft using either the configured NAT64 stateful prefix or the Well Known Prefix (WKP).
- The packet is translated from IPv6 to IPv4 and forwarded to the IPv4 network.

When an incoming packet is stateful (if a state exists for an incoming packet), NAT64 identifies the state and uses the state to translate the packet.

When Stateful NAT64 is configured on an interface, Virtual Fragmentation Reassembly (VFR) is configured automatically.

Prefixes Format for Stateful Network Address Translation 64

A set of bits at the start of an IPv6 address is called the format prefix. Prefix length is a decimal value that specifies how many of the leftmost contiguous bits of an address comprise the prefix.

When packets flow from the IPv6 to the IPv4 direction, the IPv4 host address is derived from the destination IP address of the IPv6 packet that uses the prefix length. When packets flow from the IPv4 to the IPv6 direction, the IPv4 host address is constructed using the stateful prefix.

According to the IETF address format BEHAVE draft, a u-bit (bit 70) defined in the IPv6 architecture should be set to zero. For more information on the u-bit usage, see RFC 2464. The reserved octet, also called u-octet, is reserved for compatibility with the host identifier format defined in the IPv6 addressing architecture. When constructing an IPv6 packet, the translator has to make sure that the u-bits are not tampered with and are set to the value suggested by RFC 2373. The suffix will be set to all zeros by the translator. IETF recommends that the 8 bits of the u-octet (bit range 64–71) be set to zero.

Well Known Prefix

The Well Known Prefix 64:FF9B::/96 is supported for Stateful NAT64. During a stateful translation, if no stateful prefix is configured (either on the interface or globally), the WKP prefix is used to translate the IPv4 host addresses.

Stateful IPv4-to-IPv6 Packet Flow

The packet flow of IPv4-initiated packets for Stateful NAT64 is as follows:

- The destination address is routed to a NAT Virtual Interface (NVI).

  A virtual interface is created when Stateful NAT64 is configured. For Stateful NAT64 translation to work, all packets must get routed to the NVI. When you configure an address pool, a route is automatically added to all IPv4 addresses in the pool. This route automatically points to the NVI.

- The IPv4-initiated packet hits static or dynamic binding.

  Dynamic address bindings are created by the Stateful NAT64 translator when you configure dynamic Stateful NAT64. A binding is dynamically created between an IPv6 and an IPv4 address pool. Dynamic binding is triggered by the IPv6-to-IPv4 traffic and the address is dynamically allocated. Based on your configuration, you can have static or dynamic binding.

- The IPv4-initiated packet is protocol-translated and the destination IP address of the packet is set to IPv6 based on static or dynamic binding. The Stateful NAT64 translator translates the source IP address to IPv6 by using the Stateful NAT64 prefix (if a stateful prefix is configured) or the Well Known Prefix (WKP) (if a stateful prefix is not configured).

- A session is created based on the translation information.

All subsequent IPv4-initiated packets are translated based on the previously created session.

Stateful IPv6-to-IPv4 Packet Flow

The stateful IPv6-initiated packet flow is as follows:

- The first IPv6 packet is routed to the NAT Virtual Interface (NVI) based on the automatic routing setup that is configured for the stateful prefix. Stateful NAT64 performs a series of lookups to determine whether the IPv6 packet matches any of the configured mappings based on an access control list (ACL) lookup. Based on the mapping, an IPv4 address (and port) is associated with the IPv6 destination address. The IPv6 packet is translated and the IPv4 packet is formed by using the following methods:
  - Extracting the destination IPv4 address by stripping the prefix from the IPv6 address. The source address is replaced by the allocated IPv4 address (and port).
  - The rest of the fields are translated from IPv6-to-IPv4 to form a valid IPv4 packet.

  Note This protocol translation is the same for stateless NAT64.

- A new NAT64 translation is created in the session database and in the bind database. The pool and port databases are updated depending on the configuration. The return traffic and the subsequent traffic of the IPv6 packet flow will use this session database entry for translation.

IP Packet Filtering

Stateful Network Address Translation 64 (NAT64) filters IPv6 and IPv4 packets. All IPv6 packets that are transmitted into the stateful translator are filtered because statefully translated IPv6 packets consume resources

in the translator. These packets consume processor resources for packet processing, memory resources (always session memory) for static configuration, IPv4 address resources for dynamic configuration, and IPv4 address and port resources for Port Address Translation (PAT).

Stateful NAT64 utilizes configured access control lists (ACLs) and prefix lists to filter IPv6-initiated traffic flows that are allowed to create the NAT64 state. Filtering of IPv6 packets is done in the IPv6-to-IPv4 direction because dynamic allocation of mapping between an IPv6 host and an IPv4 address can be done only in this direction.

Stateful NAT64 supports endpoint-dependent filtering for the IPv4-to-IPv6 packet flow with PAT configuration. In a Stateful NAT64 PAT configuration, the packet flow must have originated from the IPv6 realm and created the state information in NAT64 state tables. Packets from the IPv4 side that do not have a previously created state are dropped. Endpoint-independent filtering is supported with static Network Address Translation (NAT) and non-PAT configurations.

Differences Between Stateful NAT64 and Stateless NAT64

The table below displays the differences between Stateful NAT64 and Stateless NAT64.

Table 16: Differences Between Stateful NAT64 and Stateless NAT64

Supported Features	Stateful NAT64	Stateless NAT64
Address savings	*N*:1 mapping for PAT or overload configuration that saves IPv4 addresses.	One-to-one mapping—one IPv4 address is used for each IPv6 host).
Address space	IPv6 systems may use any type of IPv6 addresses.	IPv6 systems must have IPv4-translatable addresses (based on RFC 6052).
ALGs supported	FTP64	None
Protocols supported	ICMP, TCP, UDP	All
Standards	Draft-ieft-behave-v6v4-xlate-stateful-12	Draft-ietf-behave-v6v4-xlate-05
State creation	Each traffic flow creates a state in the NAT64 translator. The maximum number of states depends on the number of supported translations.	Traffic flow does not create any state in the NAT64 translator. Algorithmic operation is performed on the packet headers.

High-Speed Logging for NAT64

When HSL is configured, NAT64 provides a log of packets that flow through routing devices (similar to the Version 9 NetFlow-like records) to an external collector. Records are sent for each binding (binding is the address binding between the local address and the global address to which the local address is translated) and when sessions are created and destroyed. Session records contain the full 5-tuple of information (the source IP address, destination IP address, source port, destination port, and protocol). A tuple is an ordered list of elements. NAT64 also sends an HSL message when a NAT64 pool runs out of addresses (also called pool exhaustion). Because the pool exhaustion messages are rate limited, each packet that hits the pool exhaustion condition does not trigger an HSL message. Depending on your release, Stateful NAT64 supports high-speed logging (HSL) for upto 4 destinations.

Configure the **nat64 logging translations flow-export v9 udp destination** command to enable NAT64 HSL logging. The **vrf** keyword can be used to enable NAT64 HSL for a specific VRF

The table below describes the templates for HSL bind and session create or destroy. These fields (in the order they are displayed in the log) describe how the log collector must interpret the bytes in HSL records. The value for some of the fields varies based on whether the session is being created, destroyed, or modified.

Table 17: Templates for HSL Bind and Session Create or Destroy

Field	Format	ID	Value
Original IPv6 address	IPv6 address	27	Varies
Translated IPv4 address	IPv6 address	282	Varies
Translated IPv6 address	IPv4 address	225	Varies
Original IPv4 address	IPv4 address	12	Varies
Original IPv6 port	16-bit port	7	Varies
Translated IPv6 port	16-bit port	227	Varies
Translated IPv4 port	16-bit port	11	Varies
Original IPv4 port	16-bit port	228	Varies
Timestamp for an event	64 bits - milliseconds (This is a 64-bit field that holds the UNIX time, in milliseconds, when the event for the record occurred.)	323	Varies
VRF ID	32-bit ID	234	Zero
Protocol	8-bit value	4	Varies
Event	8-bit value	230	0–Invalid 1–Add event 2–Delete event

The table below describes the HSL pool exhaustion templates (in the order they are available in the template).

Table 18: Templates for HSL Pool Exhaustion

Field	Format	ID	Values
NAT pool ID	32-bit value	283	Varies
NAT event	8-bit value	230	3–Pool exhaust

How to Configure Enabling NAT64 High-Speed Logging per VRF

Enabling High-Speed Logging of NAT64 Translations

You can enable or disable high-speed logging (HSL) of all NAT64 translations or only translations for specific VPNs.

You must first use the **nat64 logging translations flow-export v9 udp destination** command to enable HSL for all VPN and non-VPN translations. The **vrf** keyword can be used to specify HSL destination address on a specific VRF. VPN translations are also known as Virtual Routing and Forwarding (VRF) translations.

After you enable HSL for all NAT translations, you can then use the **nat64 logging translations flow-export v9** *vrf-name* command to enable or disable translations for specific VPNs. When you use this command, HSL is disabled for all VPNs, except for the ones the command is explicitly enabled.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **nat64 logging translations flow-export v9 udp destination** *addr*|**ipv6-destination** *IPv6 address* **vrf***vrf name* **source** *interface type interface-number*
4. **nat64 logging translations flow-export v9** {*vrf-name* | **global-on**}
5. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**nat64 logging translations flow-export v9 udp destination** *addr*|**ipv6-destination** *IPv6 address* **vrf***vrf name* **source** *interface type interface-number* **Example:** This example shows how to enable high-speed logging using an IPv4 address `Device(config)# nat64 logging translations flow-export v9 udp destination 10.10.0.1 1020 source GigabitEthernet 0/0/0` **Example:** This example shows how to enable high-speed logging using an IPv6 address	Enables the high-speed logging of all VPN and non-VPN translations for up to four destinations. You can enable logging for a specific destination VRF using the **vrf** keyword. To specify an IPv6 address for the UDP destination, use the **ipv6-destination** keyword followed by the IPv6 address.

	Command or Action	Purpose

	`Device(config)# nat64 logging translations flow-export v9 udp ipv6-destination 2001::06 5050 source GigabitEthernet 0/0/0`
	Example:
	This example shows how to enable high-speed logging using an IPv6 address for a destination VRF
	`Device(config)# nat64 logging translations flow-export v9 udp ipv6-destination 2001::06 5050 vrf hslvrf source GigabitEthernet 0/0/0`	

Step 4	**nat64 logging translations flow-export v9** {*vrf-name* | **global-on**} **Example:** `Device(config)# nat64 logging translations flow-export v9 VPN-18`	Enables or disables the high-speed logging of specific NAT VPN translations.
Step 5	**exit** **Example:** `Device(config)# exit`	(Optional) Exits global configuration mode and enters privileged EXEC mode.

FTP64 Application-Level Gateway Support

The FTP64 (or service FTP) application-level gateway (ALG) helps stateful Network Address Translation 64 (NAT64) to operate on Layer 7 data. FTP64 ALG translates IP addresses and the TCP port information embedded in the payload of an FTP control session.

NAT translates any TCP/UDP traffic that does not carry source and destination IP addresses in the application data stream. Protocols that embed the IP address information within the payload (or in the application data stream) require the support of an ALG. ALGs handle application data stream (Layer 7) protocol-specific services, such as translating embedded IP addresses and port numbers in the packet payload and extracting new connection or session information from control channels.

FTP64 is automatically enabled when Stateful NAT64 is enabled. Use the **no nat64 service ftp** command to disable the NAT64 FTP service.

Note The FTP64 ALG is not supported in Stateless NAT64 translation.

Note The FTP64 ALG does not support IPv4-compatible IPv6 addresses.

Based on *IPv6-to-IPv4 translation FTP considerations draft-ietf-behave-ftp64-02* and RFC 2228, the FTP64 ALG must switch to transparent mode (a device in a transparent mode is invisible in the network; however, this device can act as a bridge and inspect or filter packets), when commands and responses flow between the FTP client and the FTP server. When a client issues the FTP AUTH command, the FTP64 ALG transparently forwards all data on the control channel in both (ingress and egress) directions, until the end of the control

channel session. Similarly, during an AUTH negotiation, the ALG must be in transparent mode, whether the negotiation is successful or not.

Based on RFC 6384, the behavior of the FTP64 ALG during a client-server communication is different. During an IPv6-to-IPv4 translation, the FTP64 ALG must transparently copy data transmitted over the control channel so that the transport layer security (TLS) session works correctly. However, the client commands and server responses are hidden from the FTP64 ALG. To ensure a consistent behavior, as soon as the initial FTP AUTH command is issued by a client, the FTP64 ALG must stop translating commands and responses and start transparently copying TCP data that is sent by the server to the client and vice versa. The FTP64 ALG must ignore the AUTH command and not go into transparent mode if the server response is in the 4*xx* or 5*xx* ranges, which comprise FTP error/warning messages.

Prior to CSCtu37975, when an IPv6 FTP client issues an FTP AUTH command, irrespective of whether the IPv4 FTP server accepts or rejects that authorization negotiation, the FTP64 ALG moves the AUTH session to transparent mode (or bypass mode). When a session is in transparent mode, NAT cannot perform translation on the packets within the session. With CSCtu37975, during a client-server communication, the FTP64 ALG's behavior is compliant with RFC 6384.

FTP64 NAT ALG Intrabox High Availability Support

Depending on your release, the FTP64 application-level gateway (ALG) adds high availability (HA) support for Stateful NAT64. The FTP64 NAT ALG Intrabox HA Support feature supports the stateful switchover between redundant Forward Processors (FPs) within a single chassis. The HA support provided by the FTP64 ALG is applicable to both intrabox HA and In-Service Software Upgrade (ISSU).

Use the **no nat64 service ftp** command to disable the NAT64 ALG service.

The FTP64 ALG synchronizes data when it receives the following messages:

- User authentication flag after 230 replies.
- ALG disable/enable flag after ALG ENABLE and ALG DISABLE messages are received.
- Fragment detection information after the first segmented packet is detected.
- Fragment detection information after the end of the segmentation is detected.

Note

- Stateful NAT64 supports only intrabox HA in some releases.
- FTP64 ALG statistics and FTP64 debug logs are not synchronized to the standby device by the FTP64 ALG.

Stateful NAT64—Intrachassis Redundancy

Depending on your release, support for the Stateful NAT64—Intrachassis Redundancy feature is available. When a second Forward Processor (FP) is available inside a single chassis, the Stateful NAT64—Intrachassis Redundancy feature enables you to configure the second FP as a standby entity. When you plug in the second FP, redundancy starts automatically with no explicit configuration. There is a short delay before the standby FP becomes the "hot standby" (which means that all sessions have been synchronized). The standby FP maintains a backup of the Stateful NAT64 session information, and when the active (first) FP fails, there is very little disruption of NAT64 sessions.

NAT64 redundancy information is sent to the standby FP in the following instances:

- When a session or a dynamic bind is created.
- When a session or a dynamic bind is deleted.
- During periodic updates. Based on the time elapsed, the active FP periodically updates the state information to the standby. Not all changes in the replicated objects are sent immediately to the standby at the time of change. The most critical updates are sent immediately, and other changes are communicated by periodic updates.

When a standby FP is inserted or when a standby FP recovers from a reload, the active FP performs a bulk synchronization to synchronize the standby FP with the active FP. NAT does an aggressive synchronization by which the active FP pushes all the state information forcefully to the standby FP.

In addition to NAT64 session information, application-specific information (application-level gateway [ALG] information) also has to be communicated to the standby FP. Each ALG has a per-session state that needs to be synchronized in the standby. The ALG triggers the sending of all ALG state information to the standby FP. NAT provides the mechanism for actually sending the ALG state and associates the state to a particular session.

HTTP sessions are not backed up on the standby FP. To replicate HTTP sessions on the standby FP during a switchover, you must configure the **nat64 switchover replicate http enable** command.

Note The Stateful NAT64—Intrachassis Redundancy feature does not support box-to-box (B2B) redundancy or asymmetric routing.

Asymmetric Routing Support for NAT64

In Cisco IOS XE Release and later releases, Network Address Translation 64 (NAT64) supports asymmetric routing and asymmetric routing with Multiprotocol Label Switching (MPLS). In NAT 64, MPLS is enabled on the IPv4 interface. Packets coming from the IPv6 interface are switched to the IPv4 interface. No configuration changes are required to enable asymmetric routing or asymmetric routing with MPLS.

For more information, see the section “Example: Configuring Asymmetric Routing Support for NAT64”.

How to Configure Stateful Network Address Translation 64

Based on your network configuration, you can configure static, dynamic, or dynamic Port Address Translation (PAT) Stateful NAT64.

Note You need to configure at least one of the configurations described in the following tasks for Stateful NAT64 to work.

Configuring Static Stateful Network Address Translation 64

You can configure a static IPv6 address to an IPv4 address and vice versa. Optionally, you can configure static Stateful NAT64 with or without ports. Perform this task to configure static Stateful NAT64.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 unicast-routing**
4. **interface** *type number*
5. **description** *string*
6. **ipv6 enable**
7. **ipv6 address** {*ipv6-address*/*prefix-length* | *prefix-name sub-bits*/*prefix-length*}
8. **nat64 enable**
9. **exit**
10. **interface** *type number*
11. **description** *string*
12. **ip address** *ip-address mask*
13. **nat64 enable**
14. **exit**
15. **nat64 prefix stateful** *ipv6-prefix*/*length*
16. **nat64 v6v4 static** *ipv6-address ipv4-address*
17. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 unicast-routing** **Example:** `Device(config)# ipv6 unicast-routing`	Enables the forwarding of IPv6 unicast datagrams.
Step 4	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 0/0/0`	Configures an interface type and enters interface configuration mode.
Step 5	**description** *string* **Example:**	Adds a description to an interface configuration.

	Command or Action	Purpose
	`Device(config-if)# description interface facing ipv6`	
Step 6	**ipv6 enable** **Example:** `Device(config-if)# ipv6 enable`	Enables IPv6 processing on an interface.
Step 7	**ipv6 address** {*ipv6-address*/*prefix-length* | *prefix-name sub-bits*/*prefix-length*} **Example:** `Device(config-if)# ipv6 address 2001:DB8:1::1/96`	Configures an IPv6 address based on an IPv6 general prefix and enables IPv6 processing on an interface.
Step 8	**nat64 enable** **Example:** `Device(config-if)# nat64 enable`	Enables NAT64 translation on an IPv6 interface.
Step 9	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and enters global configuration mode.
Step 10	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 1/2/0`	Configures an interface and enters interface configuration mode.
Step 11	**description** *string* **Example:** `Device(config-if)# description interface facing ipv4`	Adds a description to an interface configuration.
Step 12	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 209.165.201.1 255.255.255.0`	Configures an IPv4 address for an interface.
Step 13	**nat64 enable** **Example:** `Device(config-if)# nat64 enable`	Enables NAT64 translation on an IPv4 interface.
Step 14	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and enters global configuration mode.
Step 15	**nat64 prefix stateful** *ipv6-prefix*/*length* **Example:** `Device(config)# nat64 prefix stateful 2001:DB8:1::1/96`	Defines the Stateful NAT64 prefix to be added to IPv4 hosts to translate the IPv4 address into an IPv6 address. • The Stateful NAT64 prefix can be configured at the global configuration level or at the interface level.

	Command or Action	Purpose
Step 16	**nat64 v6v4 static** *ipv6-address ipv4-address* **Example:** `Device(config)# nat64 v6v4 static 2001:DB8:1::FFFE 209.165.201.1`	Enables NAT64 IPv6-to-IPv4 static address mapping.
Step 17	**end** **Example:** `Device(config)# end`	Exits global configuration mode and enters privileged EXEC mode.

Configuring Dynamic Stateful Network Address Translation 64

A dynamic Stateful NAT64 configuration provides a one-to-one mapping of IPv6 addresses to IPv4 addresses in the address pool. You can use the dynamic Stateful NAT64 configuration when the number of active IPv6 hosts is less than the number of IPv4 addresses in the pool. Perform this task to configure dynamic Stateful NAT64.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 unicast-routing**
4. **interface** *type number*
5. **description** *string*
6. **ipv6 enable**
7. **ipv6** {*ipv6-address*/*prefix-length* | *prefix-name sub-bits*/*prefix-length*}
8. **nat64 enable**
9. **exit**
10. **interface** *type number*
11. **description** *string*
12. **ip address** *ip-address mask*
13. **nat64 enable**
14. **exit**
15. **ipv6 access-list** *access-list-name*
16. **permit ipv6** *ipv6-address* **any**
17. **exit**
18. **nat64 prefix stateful** *ipv6-prefix*/*length*
19. **nat64 v4 pool** *pool-name start-ip-address end-ip-address*
20. **nat64 v6v4 list** *access-list-name* **pool** *pool-name*
21. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Device> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 unicast-routing** **Example:** `Device(config)# ipv6 unicast-routing`	Enables the forwarding of IPv6 unicast datagrams.
Step 4	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 0/0/0`	Configures an interface type and enters interface configuration mode.
Step 5	**description** *string* **Example:** `Device(config-if)# description interface facing ipv6`	Adds a description to an interface configuration.
Step 6	**ipv6 enable** **Example:** `Device(config-if)# ipv6 enable`	Enables IPv6 processing on an interface.
Step 7	**ipv6** {*ipv6-address*|*prefix-length* | *prefix-name sub-bits*|*prefix-length*} **Example:** `Device(config-if)# ipv6 2001:DB8:1::1/96`	Configures an IPv6 address based on an IPv6 general prefix and enables IPv6 processing on an interface.
Step 8	**nat64 enable** **Example:** `Device(config-if)# nat64 enable`	Enables Stateful NAT64 translation on an IPv6 interface.
Step 9	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and enters global configuration mode.
Step 10	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 1/2/0`	Configures an interface type and enters interface configuration mode
Step 11	**description** *string* **Example:** `Device(config-if)# description interface facing ipv4`	Adds a description to an interface configuration.

	Command or Action	Purpose
Step 12	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 209.165.201.24 255.255.255.0`	Configures an IPv4 address for an interface.
Step 13	**nat64 enable** **Example:** `Device(config-if)# nat64 enable`	Enables Stateful NAT64 translation on an IPv4 interface.
Step 14	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and enters global configuration mode.
Step 15	**ipv6 access-list** *access-list-name* **Example:** `Device(config)# ipv6 access-list nat64-acl`	Defines an IPv6 access list and enters IPv6 access list configuration mode.
Step 16	**permit ipv6** *ipv6-address* **any** **Example:** `Device(config-ipv6-acl)# permit ipv6 2001:DB8:2::/96 any`	Sets permit conditions for an IPv6 access list.
Step 17	**exit** **Example:** `Device(config-ipv6-acl# exit`	Exits IPv6 access list configuration mode and enters global configuration mode.
Step 18	**nat64 prefix stateful** *ipv6-prefix/length* **Example:** `Device(config)# nat64 prefix stateful 2001:DB8:1::1/96`	Enables NAT64 IPv6-to-IPv4 address mapping.
Step 19	**nat64 v4 pool** *pool-name start-ip-address end-ip-address* **Example:** `Device(config)# nat64 v4 pool pool1 209.165.201.1 209.165.201.254`	Defines the Stateful NAT64 IPv4 address pool.
Step 20	**nat64 v6v4 list** *access-list-name* **pool** *pool-name* **Example:** `Device(config)# nat64 v6v4 list nat64-acl pool pool1`	Dynamically translates an IPv6 source address to an IPv6 source address and an IPv6 destination address to an IPv4 destination address for NAT64.
Step 21	**end** **Example:** `Device(config)# end`	Exits global configuration mode and enters privileged EXEC mode.

Configuring Dynamic Port Address Translation Stateful NAT64

A Port Address Translation (PAT) or overload configuration is used to multiplex (mapping IPv6 addresses to a single IPv4 pool address) multiple IPv6 hosts to a pool of available IPv4 addresses on a first-come first-served basis. The dynamic PAT configuration conserves the IPv4 address space while providing connectivity to the IPv4 Internet. Configure the **nat64 v6v4 list** command with the **overload** keyword to configure PAT address translation. Perform this task to configure dynamic PAT Stateful NAT64.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 unicast-routing**
4. **interface** *type number*
5. **description** *string*
6. **ipv6 enable**
7. **ipv6** {*ipv6-address*/*prefix-length* | *prefix-name sub-bits*/*prefix-length*}
8. **nat64 enable**
9. **exit**
10. **interface** *type number*
11. **description** *string*
12. **ip address** *ip-address mask*
13. **nat64 enable**
14. **exit**
15. **ipv6 access-list** *access-list-name*
16. **permit ipv6** *ipv6-address* **any**
17. **exit**
18. **nat64 prefix stateful** *ipv6-prefix*/*length*
19. **nat64 v4 pool** *pool-name start-ip-address end-ip-address*
20. **nat64 v6v4 list** *access-list-name* **pool** *pool-name* **overload**
21. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 unicast-routing** **Example:** `Device(config)# ipv6 unicast-routing`	Enables the forwarding of IPv6 unicast datagrams.

	Command or Action	Purpose
Step 4	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 0/0/0`	Configures an interface type and enters interface configuration mode.
Step 5	**description** *string* **Example:** `Device(config-if)# description interface facing ipv6`	Adds a description to an interface configuration.
Step 6	**ipv6 enable** **Example:** `Device(config-if)# ipv6 enable`	Enables IPv6 processing on an interface.
Step 7	**ipv6** {*ipv6-address*/*prefix-length* | *prefix-name sub-bits*/*prefix-length*} **Example:** `Device(config-if)# ipv6 2001:DB8:1::1/96`	Configures an IPv6 address based on an IPv6 general prefix and enables IPv6 processing on an interface.
Step 8	**nat64 enable** **Example:** `Device(config-if)# nat64 enable`	Enables Stateful NAT64 translation on an IPv6 interface.
Step 9	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and enters global configuration mode.
Step 10	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 1/2/0`	Configures an interface type and enters interface configuration mode
Step 11	**description** *string* **Example:** `Device(config-if)# description interface facing ipv4`	Adds a description to an interface configuration.
Step 12	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 209.165.201.24 255.255.255.0`	Configures an IPv4 address for an interface.
Step 13	**nat64 enable** **Example:** `Device(config-if)# nat64 enable`	Enables Stateful NAT64 translation on an IPv6 interface.

	Command or Action	Purpose
Step 14	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and enters global configuration mode.
Step 15	**ipv6 access-list** *access-list-name* **Example:** `Device(config)# ipv6 access-list nat64-acl`	Defines an IPv6 access list and places the device in IPv6 access list configuration mode.
Step 16	**permit ipv6** *ipv6-address* **any** **Example:** `Device(config-ipv6-acl)# permit ipv6 2001:db8:2::/96 any`	Sets permit conditions for an IPv6 access list.
Step 17	**exit** **Example:** `Device(config-ipv6-acl)# exit`	Exits IPv6 access list configuration mode and enters global configuration mode.
Step 18	**nat64 prefix stateful** *ipv6-prefix/length* **Example:** `Device(config)# nat64 prefix stateful 2001:db8:1::1/96`	Enables NAT64 IPv6-to-IPv4 address mapping.
Step 19	**nat64 v4 pool** *pool-name start-ip-address end-ip-address* **Example:** `Device(config)# nat64 v4 pool pool1 209.165.201.1 209.165.201.254`	Defines the Stateful NAT64 IPv4 address pool.
Step 20	**nat64 v6v4 list** *access-list-name* **pool** *pool-name* **overload** **Example:** `Device(config)# nat64 v6v4 list nat64-acl pool pool1 overload`	Enables NAT64 PAT or overload address translation.
Step 21	**end** **Example:** `Device(config)# end`	Exits global configuration mode and enters privileged EXEC mode.

Monitoring and Maintaining a Stateful NAT64 Routing Network

Use the following commands in any order to display the status of your Stateful Network Address Translation 64 (NAT64) configuration.

SUMMARY STEPS

1. **show nat64 aliases** [*lower-address-range upper-address-range*]

2. **show nat64 logging**
3. **show nat64 prefix stateful** {**global** | {**interfaces** | **static-routes**} [**prefix** *ipv6-address*/*prefix-length*]}
4. **show nat64 timeouts**

DETAILED STEPS

Step 1 **show nat64 aliases** [*lower-address-range upper-address-range*]

This command displays the IP aliases created by NAT64.

Example:

```
Device# show nat64 aliases

Aliases configured: 1
Address   Table ID  Inserted   Flags   Send ARP   Reconcilable   Stale   Ref-Count
10.1.1.1  0         FALSE      0x0030  FALSE      TRUE           FALSE   1
```

Step 2 **show nat64 logging**

This command displays NAT64 logging.

Example:

```
Device# show nat64 logging

NAT64 Logging Type

Method       Protocol   Dst. Address   Dst. Port   Src. Port
translation
flow export  UDP        10.1.1.1       5000        60087
```

Step 3 **show nat64 prefix stateful** {**global** | {**interfaces** | **static-routes**} [**prefix** *ipv6-address*/*prefix-length*]}

This command displays information about NAT64 stateful prefixes.

Example:

```
Device# show nat64 prefix stateful interfaces

Stateful Prefixes

Interface               NAT64   Enabled   Global Prefix
GigabitEthernet0/1/0    TRUE    TRUE      2001:DB8:1:1/96
GigabitEthernet0/1/3    TRUE    FALSE     2001:DB8:2:2/96
```

Step 4 **show nat64 timeouts**

This command displays statistics for NAT64 translation session timeout.

Example:

```
Device# show nat64 timeouts

NAT64 Timeout

Seconds    CLI Cfg    Uses 'All'    all flows
86400      FALSE      FALSE         udp
300        FALSE      TRUE          tcp
7200       FALSE      TRUE          tcp-transient
```

```
240         FALSE       FALSE          icmp
60          FALSE       TRUE
```

Configuration Examples for Stateful Network Address Translation 64

Example: Configuring Static Stateful Network Address Translation 64

```
Device# configure terminal
Device(config)# ipv6 unicast-routing
Device(config)# interface gigabitethernet 0/0/0
Device(config-if)# description interface facing ipv6
Device(config-if)# ipv6 enable
Device(config-if)# ipv6 address 2001:DB8:1::1/96
Device(confif-if)# nat64 enable
Device(config-fi)# exit
Device(config)# interface gigabitethernet 1/2/0
Device(config-if)# description interface facing ipv4
Device(config-if)# ip address 209.165.201.1 255.255.255.0
Device(config-if)# nat64 enable
Device(config-if)# exit
Device(config)# nat64 prefix stateful 2001:DB8:1::1/96
Device(config)# nat64 v6v4 static 2001:DB8:1::FFFE 209.165.201.1
Device(config)# end
```

Example: Configuring Dynamic Stateful Network Address Translation 64

```
Device# configure terminal
Device(config)# ipv6 unicast-routing
Device(config)# interface gigabitethernet 0/0/0
Device(config-if)# description interface facing ipv6
Device(config-if)# ipv6 enable
Device(config-if)# ipv6 2001:DB8:1::1/96
Device(config-if)# nat64 enable
Device(config-if)# exit
Device(config)# interface gigabitethernet 1/2/0
Device(config-if)# description interface facing ipv4
Device(config-if)# ip address 209.165.201.24 255.255.255.0
Device(config-if)# nat64 enable
Device(config-if)# exit
Device(config)# ipv6 access-list nat64-acl
Device(config-ipv6-acl)# permit ipv6 2001:db8:2::/96 any
Device(config-ipv6-acl)# exit
Device(config)# nat64 prefix stateful 2001:db8:1::1/96
Device(config)# nat64 v4 pool pool1 209.165.201.1 209.165.201.254
Device(config)# nat64 v6v4 list nat64-acl pool pool1
Device(config)# end
```

Example: Configuring Dynamic Port Address Translation Stateful NAT64

```
enable
 configure terminal
  ipv6 unicast-routing
  interface gigabitethernet 0/0/0
   description interface facing ipv6
   ipv6 enable
   ipv6 2001:DB8:1::1/96
   nat64 enable
   exit
  interface gigabitethernet 1/2/0
   description interface facing ipv4
   ip address 209.165.201.24 255.255.255.0
   nat64 enable
   exit
  ipv6 access-list nat64-acl
   permit ipv6 2001:db8:2::/96 any
   exit
  nat64 prefix stateful 2001:db8:1::1/96
  nat64 v4 pool pool1 209.165.201.1 209.165.201.254
  nat64 v6v4 list nat64-acl pool pool1 overload
  end
```

Example: Configuring Asymmetric Routing Support for NAT64

The following example shows how to configure asymmetric routing for Network Address Translation 64 (NAT64):

```
!RouterA Configuration

Device(config)# ipv6 unicast-routing
Device(config)# nat64 prefix stateful 2001:db8:2::/96
Device(config)# nat64 v6v4 static 2001:db8:1::5 150.0.0.1 redundancy 1 mapping-id 150
Device(config)# nat64 v6v4 static 2001:db8:1::6 150.0.0.2 redundancy 1 mapping-id 151
Device(config)# nat64 switchover replicate http enable port 80
!
Device(config)# redundancy
Device(config-red)# application redundancy
Device(config-red-app)# group 1
Device(config-red-app-grp)# name RG1
Device(config-red-app-grp)# data gigabitethernet 1/1/0
Device(config-red-app-grp)# control gigabitethernet 1/1/1 protocol 1
Device(config-red-app-grp)# asymmetric-routing interface gigabitethernet 1/1/2
Device(config-red-app-grp)# priority 150 failover threshold 140
Device(config-red-app-grp)# asymmetric-routing always-divert enable
Device(config-red-app-grp)# exit
Device(config-red-app)# exit
Device(config-red)# exit
!
Device(config)# interface gigabitethernet 1/1/0
Device(config-if)# ip address 10.10.10.1 255.255.255.0
Device(config-if)# no shutdown
Device(config-if)# exit
!
Device(config)# interface gigabitethernet 1/1/1
Device(config-if)# ip address 172.16.0.1 255.240.0.0
```

```
Device(config-if)# no shutdown
Device(config-if)# exit
!
Device(config)# interface gigabitethernet 1/1/2
Device(config-if)# ip address 192.168.0.1 255.255.0.0
Device(config-if)# no shutdown
Device(config-if)# exit
!
Device(config)# interface gigabitethernet 1/1/3
Device(config-if)# ipv6 enable
Device(config-if)# no shutdown
Device(config-if)# nat64 enable
Device(config-if)# ipv6 address 2001:db8:1::2/96
Device(config-if)# redundancy rii 100
Device(config-if)# redundancy group 1 ipv6 2001:db8:1::1/96 exclusive decrement 15
Device(config-if)# exit
!
Device(config)# interface gigabitethernet 1/1/4
Device(config-if)# ip address 192.0.2.1 255.255.255.0
Device(config-if)# nat64 enable
Device(config-if)# no shutdown
Device(config-if)# redundancy rii 101
Device(config-if)# redundancy asymmetric-routing enable
Device(config-if)# exit
!
Device(config)# router ospf 90
Device(config-router)# network 192.0.2.0 255.255.255.0 area 0
Device(config-router)# end


! Router B Configuration

Device(config)# ipv6 unicast-routing
Device(config)# nat64 prefix stateful 2001:db8:2::/96
Device(config)# nat64 v6v4 static 2001:db8:1::5 150.0.0.1 redundancy 1 mapping-id 150
Device(config)# nat64 v6v4 static 2001:db8:1::6 150.0.0.2 redundancy 1 mapping-id 151
Device(config)# nat64 switchover replicate http enable port 80
!
Device(config)# redundancy
Device(config-red)# application redundancy
Device(config-red-app)# group 1
Device(config-red-app-grp)# name RG1
Device(config-red-app-grp)# data gigabitethernet 1/2/0
Device(config-red-app-grp)# control gigabitethernet 1/2/1 protocol 1
Device(config-red-app-grp)# asymmetric-routing interface gigabitethernet 1/2/2
Device(config-red-app-grp)# priority 140 failover threshold 135
Device(config-red-app-grp)# asymmetric-routing always-divert enable
Device(config-red-app-grp)# exit
Device(config-red-app)# exit
Device(config-red)# exit
!
Device(config)# interface gigabitethernet 1/2/0
Device(config-if)# ip address 10.10.10.2 255.255.255.0
Device(config-if)# no shutdown
Device(config-if)# exit
!
Device(config)# interface gigabitethernet 1/2/1
Device(config-if)# ip address 172.16.0.2 255.240.0.0
Device(config-if)# no shutdown
Device(config-if)# exit
!
Device(config)# interface gigabitethernet 1/2/2
```

```
Device(config-if)# ip address 192.168.0.2 255.255.0.0
Device(config-if)#no shutdown
Device(config-if)# exit
!
Device(config-if)# interface gigabitethernet 1/2/3
Device(config-if)# ipv6 enable
Device(config-if)# no shutdown
Device(config-if)# nat64 enable
Device(config-if)# ipv6 addr 2001:db8:1::3/96
Device(config-if)# redundancy rii 100
Device(config-if)# redundancy group 1 ipv6 2001:db8:1::1/96 exclusive decrement 15
Device(config-if)# exit
!
Device(config)# interface gigabitethernet 1/2/4
Device(config-if)# ip address 198.51.100.1 255.255.255.0
Device(config-if)# nat64 enable
Device(config-if)# no shutdown
Device(config-if)# redundancy rii 101
Device(config-if)# redundancy asymmetric-routing enable
Device(config-if)# exit
!
Device(config)# router ospf 90
Device(config-router)# network 198.51.100.0 255.255.255.0 area 0
Device(config-router)# end
```

Additional References for Stateful Network Address Translation 64

Related Documents

Related Topic	Document Title
Cisco IOS commands	Master Command List, All Releases
NAT commands	IP Addressing Services Command Reference

Standards and RFCs

Standard/RFC	Title
Framework for IPv4/IPv6 Translation	Framework for IPv4/IPv6 Translation draft-ietf-behave-v6v4-framework-06
FTP ALG for IPv6-to-IPv4 translation	An FTP ALG for IPv6-to-IPv4 translation draft-ietf-behave-ftp64-06
IP/ICMP Translation Algorithm	IP/ICMP Translation Algorithm draft-ietf-behave-v6v4-xlate-10
IPv6 Addressing of IPv4/IPv6 Translators	IPv6 Addressing of IPv4/IPv6 Translators draft-ietf-behave-address-format-07
RFC 2228	FTP Security Extensions

Standard/RFC	Title
RFC 2373	IP Version 6 Addressing Architecture
RFC 2464	Transmission of IPv6 Packets over Ethernet Networks
RFC 2765	Stateless IP/ICMP Translation Algorithm (SIIT)
RFC 2766	Network Address Translation - Protocol Translation (NAT-PT)
RFC 4787	Network Address Translation (NAT) Behavioral Requirements for Unicast UDP
RFC 4966	Reasons to Move the Network Address Translator - Protocol Translator (NAT-PT) to Historic Status
RFC 6384	An FTP Application Layer Gateway (ALG) for IPv6-to-IPv4 Translation
Stateful NAT64: Network Address and Protocol Translation from IPv6 Clients to IPv4 Servers	Stateful NAT64: Network Address and Protocol Translation from IPv6 Clients to IPv4 Servers draft-ietf-behave-v6v4-xlate-stateful-12

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Stateful Network Address Translation 64

Table 19: Feature Information for Stateful Network Address Translation 64

Feature Name	Releases	Feature Information
Asymmetric Routing Support for NAT64	Cisco IOS XE Release 3.16S	In Cisco IOS XE Release and later releases, Network Address Translation 64 (NAT64) supports asymmetric routing and asymmetric routing with Multiprotocol Label Switching (MPLS).

Feature Name	Releases	Feature Information
FTP64 NAT ALG Intrabox HA Support	Cisco IOS XE Release 3.5S	In Cisco IOS XE Release 3.5S, the FTP64 ALG adds HA support for Stateful NAT64. The FTP64 NAT ALG Intrabox HA Support feature supports the stateful switchover between redundant FPs within a single chassis. The HA support provided by the FTP64 ALG is applicable to both intrabox and interbox HA and In-Service Software Upgrade (ISSU).
Stateful NAT64 ALG—Stateful FTP64 ALG Support	Cisco IOS XE Release 3.4S	Cisco IOS XE Release 3.4S and later releases support FTP64 (or service FTP) ALGs. The FTP64 ALG helps Stateful NAT64 operate on Layer 7 data. An FTP ALG translates IP addresses and the TCP port information embedded in the payload of an FTP control session. The following commands were introduced or modified: **nat64 service ftp**.
Stateful NAT64—Intra-Chassis Redundancy	Cisco IOS XE Release 3.5S Cisco IOS XE Release 3.10S	Cisco IOS XE Release 3.5S and later releases support the Stateful NAT64—Intra-Chassis Redundancy feature. When a second Forward Processor (FP) is available inside a single chassis, the Stateful NAT64 Intra-Chassis Redundancy feature enables you to configure the second FP as a standby entity. The standby FP maintains a backup of the stateful NAT64 session information and when the active (first) FP fails, there is no disruption of NAT64 sessions. The following commands were introduced or modified: **nat64 switchover replicate http port**.

Feature Name	Releases	Feature Information
Stateful Network Address Translation 64	Cisco IOS XE Release 3.4S	The Stateful Network Address Translation 64 feature provides a translation mechanism that translates IPv6 packets into IPv4 packets and vice versa. The Stateful NAT64 translator, algorithmically translates the IPv4 addresses of IPv4 hosts to and from IPv6 addresses by using the configured stateful prefix. In a similar manner, the IPv6 addresses of IPv6 hosts are translated to and from IPv4 addresses through NAT. The following commands were introduced or modified: **clear nat64 statistics**, **debug nat64**, **nat64 logging**, **nat64 prefix stateful**, **nat64 translation**, **nat64 v4**, **nat64 v4v6**, **nat64 v6v4**, **show nat64 aliases**, **show nat64 limits**, **show nat64 logging**, **show nat64 mappings dynamic**, **show nat64 mappings static**, **show nat64 services**, **show nat64 pools**, **show nat64 prefix stateful**, **show nat64 statistics**, **show nat64 timeouts**, and **show nat64 translations**.

Glossary

ALG—application-layer gateway or application-level gateway.

FP—Forward Processor.

IPv4-converted address—IPv6 addresses used to represent the IPv4 hosts. These have an explicit mapping relationship to the IPv4 addresses. This relationship is self-described by mapping the IPv4 address in the IPv6 address. Both stateless and stateful translators use IPv4-converted IPv6 addresses to represent the IPv4 hosts.

IPv6-converted address—IPv6 addresses that are assigned to the IPv6 hosts for the stateless translator. These IPv6-converted addresses have an explicit mapping relationship to the IPv4 addresses. This relationship is self-described by mapping the IPv4 address in the IPv6 address. The stateless translator uses the corresponding IPv4 addresses to represent the IPv6 hosts. The stateful translator does not use IPv6-converted addresses, because the IPv6 hosts are represented by the IPv4 address pool in the translator via dynamic states.

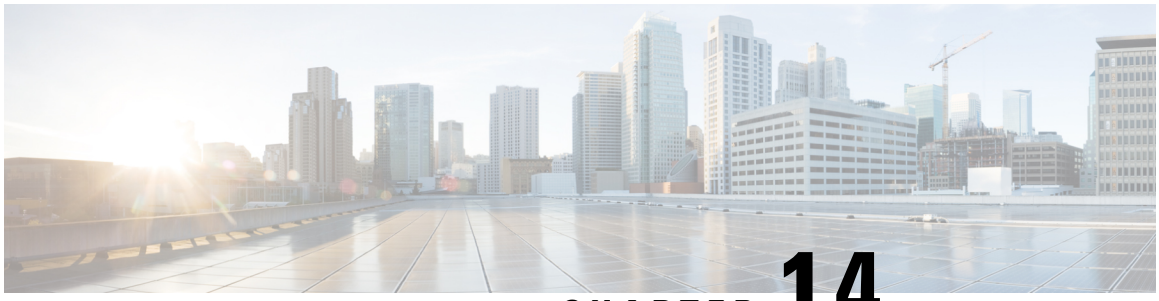
NAT—Network Address Translation.

RP—Route Processor.

stateful translation—In stateful translation a per-flow state is created when the first packet in a flow is received. A translation algorithm is said to be stateful if the transmission or reception of a packet creates or modifies a data structure in the relevant network element. Stateful translation allows the use of multiple translators interchangeably and also some level of scalability. Stateful translation is defined to enable the IPv6 clients and peers without mapped IPv4 addresses to connect to the IPv4-only servers and peers.

stateless translation—A translation algorithm that is not stateful is called stateless. A stateless translation requires configuring a static translation table, or may derive information algorithmically from the messages it is translating. Stateless translation requires less computational overhead than stateful translation. It also

requires less memory to maintain the state, because the translation tables and the associated methods and processes exist in a stateful algorithm and do not exist in a stateless one. Stateless translation enables the IPv4-only clients and peers to initiate connections to the IPv6-only servers or peers that are equipped with IPv4-embedded IPv6 addresses. It also enables scalable coordination of IPv4-only stub networks or ISP IPv6-only networks. Because the source port in an IPv6-to-IPv4 translation may have to be changed to provide adequate flow identification, the source port in the IPv4-to-IPv6 direction need not be changed.



CHAPTER 14

Stateful Network Address Translation 64 Interchassis Redundancy

The Stateful Network Address Translation 64 Interchassis Redundancy feature adds interchassis redundancy support to stateful Network Address Translation 64 (NAT64). The stateful interchassis redundancy enables you to configure pairs of devices to act as backups for each other.

This module describes how to configure stateful NAT64 interchassis redundancy.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for Stateful Network Address Translation 64 Interchassis Redundancy

- Asymmetric routing is not supported.
- Box-to-box (B2B) redundancy along with intrachassis redundancy is not supported.
- NAT interface overload configuration is not supported.

Information About Stateful Network Address Translation 64 Interchassis Redundancy

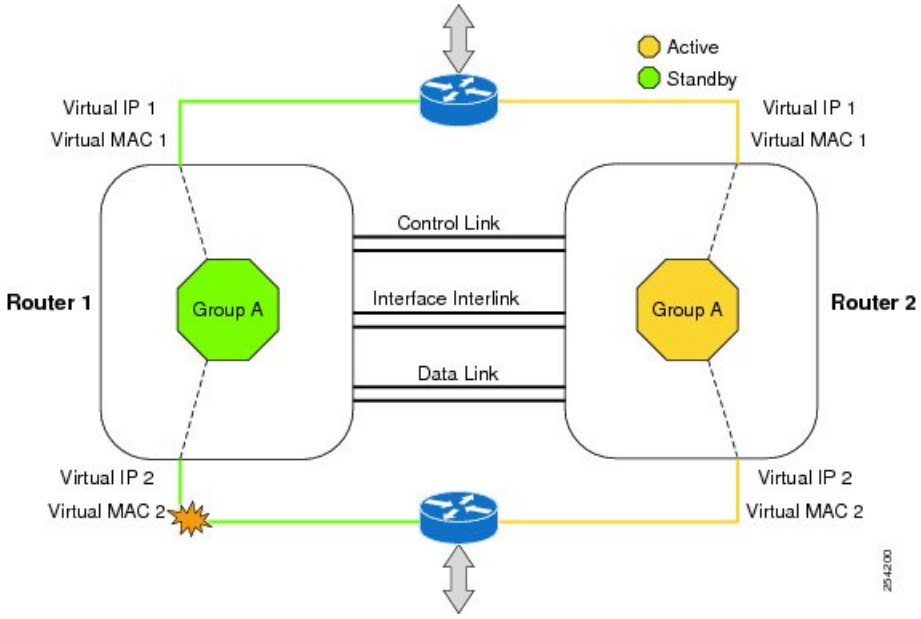
Stateful Interchassis Redundancy Operation

You can configure pairs of devices to act as hot standbys for each other. Redundancy is configured on an interface basis. Pairs of redundant interfaces are known as redundancy groups (RGs). Redundancy occurs at an application level and does not require a complete physical failure of the interface or device for a switchover of the application to occur. When a switchover occurs, the application activity continues to run seamlessly on the redundant interface.

The figure below depicts an active/standby load-sharing scenario. The figure shows how an RG is configured for a pair of devices that has one outgoing interface. Group A on Router 1 is the active RG and Group A on Router 2 is the standby RG.

Redundant devices are joined by a configurable control link and a data synchronization link. The control link is used to communicate the status of devices. The data synchronization link is used to transfer stateful information from Network Address Translation (NAT) and the firewall and synchronize the stateful database. The pairs of redundant interfaces are configured with the same unique ID number known as the redundant interface identifier (RII).

Figure 15: Redundancy Group Configuration—One Outgoing Interface



The status of redundancy group members is determined through the use of hello messages sent over the control link. The software considers either device not responding to a hello message within a configurable amount of time to be a failure and initiates a switchover. For the software to detect a failure in milliseconds, control links run the failover protocol that is integrated with the Bidirectional Forwarding Detection (BFD) protocol. You can configure the following parameters for hello messages:

- Hello time—Interval at which hello messages are sent.

- Hold time—Amount of time before which the active or standby device is declared to be down.

The hello time defaults to 3 seconds to align with the Hot Standby Router Protocol (HSRP), and the hold time defaults to 10 seconds. You can also configure these timers in milliseconds by using the **timers hellotime msec** command.

To determine the pairs of interfaces that are affected by the switchover, you must configure a unique ID for each pair of redundant interfaces. This ID is known as the RII that is associated with the interface.

A switchover to the standby device can occur when the priority setting that is configured on each device changes. The device with the highest priority value acts as the active device. If a fault occurs on either the active or standby device, the priority of the device is decremented by a configurable amount known as the weight. If the priority of the active device falls below the priority of the standby device, a switchover occurs and the standby device becomes the active device. This default behavior can be overridden by disabling the preemption attribute for the RG. You can also configure each interface to decrease the priority when the Layer 1 state of the interface goes down. The priority that is configured overrides the default priority of an RG.

Each failure event that causes a modification of an RG priority generates a syslog entry that contains a time stamp, the RG that was affected, the previous priority, the new priority, and a description of the failure event cause.

A switchover also can occur when the priority of a device or interface falls below a configurable threshold level.

A switchover to the standby device occurs under the following circumstances:

- Power loss or a reload occurs on the active device (including reloads).
- The run-time priority of the active device goes below that of the standby device (with preempt configured).
- The run-time priority of the active device goes below that of the configured threshold.
- The redundancy group on the active device is reloaded manually. Use the **redundancy application reload group** *rg-number* command for a manual reload.

Active/Active Failover

In an active/active failover configuration, both devices can process network traffic. Active/active failover generates virtual MAC (VMAC) addresses for interfaces in each redundancy group (RG).

One device in an active/active failover pair is designated as the primary (active) device, and the other is designated as the secondary (standby) device. Unlike with active/standby failover, this designation does not indicate which device becomes active when both devices start simultaneously. Instead, the primary/secondary designation determines the following:

- The device that provides the running configuration to the failover pair when they start simultaneously.
- The device on which the failover RG appears in the active state when devices start simultaneously. Each failover RG in the configuration is configured with a primary or secondary device preference. You can configure both failover RGs to be in the active state on a single device and the standby failover RGs to be on the other device. You can also configure one failover RG to be in the active state and the other RG to be in the standby state on a single device.

Active/Standby Failover

Active/standby failover enables you to use a standby device to take over the functionality of a failed device. A failed active device changes to the standby state, and the standby device changes to the active state. The

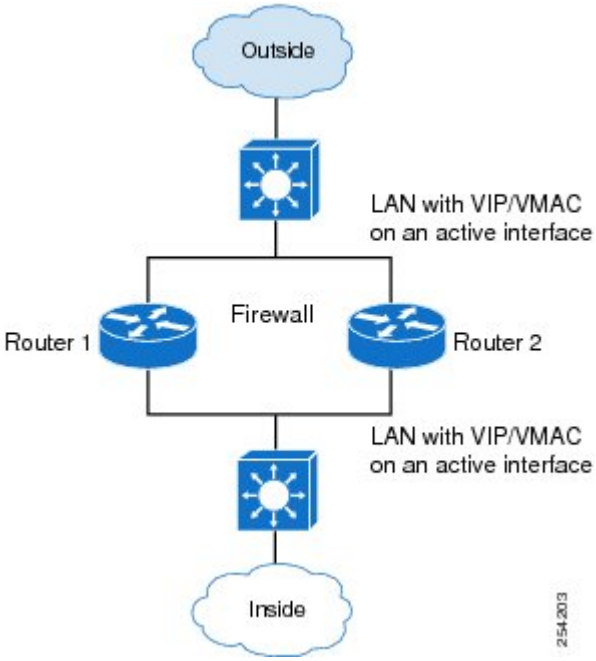
device that is now in the active state takes over IP addresses and MAC addresses of the failed device and starts processing traffic. The device that is now in the standby state takes over standby IP addresses and MAC addresses. Because network devices do not see any change in the MAC-to-IP address pairing, Address Resolution Protocol (ARP) entries do not change or time out anywhere on the network.

In an active/standby scenario, the main difference between two devices in a failover pair depends on which device is active and which device is a standby, namely which IP addresses to use and which device actively passes the traffic. The active device always becomes the active device if both devices start up at the same time (and are of equal operational health). MAC addresses of the active device are always paired with active IP addresses.

LAN-LAN Topology

In a LAN-LAN topology, all participating devices are connected to each other through LAN interfaces on both the inside and the outside. In this scenario, the traffic is often directed to the correct firewall if static routing is configured on the upstream or downstream devices to an appropriate virtual IP address. The dynamic routing configuration supported on LAN-facing interfaces must not introduce a dependency on routing protocol convergence; otherwise, fast failover requirements will not be met. The figure below shows a LAN-LAN topology.

Figure 16: LAN-LAN Scenario



Redundancy Groups for Stateful NAT64

To support stateful Network Address Translation 64 (NAT64) box-to-box (B2B) redundancy, all stateful NAT64 mappings must be associated with a redundancy group (RG). You can associate multiple stateful NAT64 mappings with one RG. Any session or bind that is created from a stateful NAT64 mapping is associated with the RG to which the stateful NAT64 is mapped. In B2B redundancy, stateful NAT64 checks the state of the created, changed, or destroyed session or bind in the RG to determine whether the stateful NAT64 high availability (HA) message should be sent to the standby device.

NAT binding is a one-to-one association between a local IP address and a global IP address. Sessions are identified by the 5-tuple (the source IP address, the destination IP address, the protocol, the source port, and the destination port) information. Sessions are normally created and destroyed at a much faster rate than bindings.

Translation Filtering

RFC 4787 provides translation filtering behaviors for Network Address Translation (NAT). The following options are used by NAT to filter packets that originate from specific external endpoints:

- Endpoint-independent filtering—Filters out packets that are not destined to an internal IP address and port regardless of the external IP address and port source.
- Address-dependent filtering—Filters out packets that are not destined to an internal IP address. NAT also filters out packets that are destined for an internal endpoint.
- Address- and port-dependent filtering—Filters out packets that are not destined to an internal IP address. NAT also filters out packets that are destined for an internal endpoint if packets were not sent to the endpoint previously.

FTP64 Application-Level Gateway Support

The FTP64 (or service FTP) application-level gateway (ALG) helps stateful Network Address Translation 64 (NAT64) to operate on Layer 7 data. FTP64 ALG translates IP addresses and the TCP port information embedded in the payload of an FTP control session.

NAT translates any TCP/UDP traffic that does not carry source and destination IP addresses in the application data stream. Protocols that embed the IP address information within the payload (or in the application data stream) require the support of an ALG. ALGs handle application data stream (Layer 7) protocol-specific services, such as translating embedded IP addresses and port numbers in the packet payload and extracting new connection or session information from control channels.

FTP64 is automatically enabled when Stateful NAT64 is enabled. Use the **no nat64 service ftp** command to disable the NAT64 FTP service.

Note The FTP64 ALG is not supported in Stateless NAT64 translation.

Note The FTP64 ALG does not support IPv4-compatible IPv6 addresses.

Based on *IPv6-to-IPv4 translation FTP considerations draft-ietf-behave-ftp64-02* and RFC 2228, the FTP64 ALG must switch to transparent mode (a device in a transparent mode is invisible in the network; however, this device can act as a bridge and inspect or filter packets), when commands and responses flow between the FTP client and the FTP server. When a client issues the FTP AUTH command, the FTP64 ALG transparently forwards all data on the control channel in both (ingress and egress) directions, until the end of the control channel session. Similarly, during an AUTH negotiation, the ALG must be in transparent mode, whether the negotiation is successful or not.

Based on RFC 6384, the behavior of the FTP64 ALG during a client-server communication is different. During an IPv6-to-IPv4 translation, the FTP64 ALG must transparently copy data transmitted over the control channel so that the transport layer security (TLS) session works correctly. However, the client commands and server responses are hidden from the FTP64 ALG. To ensure a consistent behavior, as soon as the initial FTP AUTH command is issued by a client, the FTP64 ALG must stop translating commands and responses and start transparently copying TCP data that is sent by the server to the client and vice versa. The FTP64 ALG must ignore the AUTH command and not go into transparent mode if the server response is in the 4*xx* or 5*xx* ranges, which comprise FTP error/warning messages.

Prior to CSCtu37975, when an IPv6 FTP client issues an FTP AUTH command, irrespective of whether the IPv4 FTP server accepts or rejects that authorization negotiation, the FTP64 ALG moves the AUTH session to transparent mode (or bypass mode). When a session is in transparent mode, NAT cannot perform translation on the packets within the session. With CSCtu37975, during a client-server communication, the FTP64 ALG's behavior is compliant with RFC 6384.

How to Configure Stateful Network Translation 64 Interchassis Redundancy

Configuring Redundancy Group Protocols

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **redundancy**
4. **application redundancy**
5. **protocol** *id*
6. **name** *group-name*
7. Repeat Steps 3 to 6 to configure a redundancy group protocol on another device.
8. **timers hellotime** *seconds* **holdtime** *seconds*
9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**redundancy** **Example:**	Enters redundancy configuration mode.

	Command or Action	Purpose
	`Device(config)# redundancy`	
Step 4	**application redundancy** **Example:** `Device(red)# application redundancy`	Configures application redundancy and enters redundancy application configuration mode.
Step 5	**protocol** *id* **Example:** `Device(config-red-app)# protocol 1`	Defines a protocol instance for a redundancy group and enters redundancy application protocol configuration mode.
Step 6	**name** *group-name* **Example:** `Device(config-red-app-prtcl)# name RG1`	Configures a name for the redundancy group.
Step 7	Repeat Steps 3 to 6 to configure a redundancy group protocol on another device.	—
Step 8	**timers hellotime** *seconds* **holdtime** *seconds* **Example:** `Device(config-red-app-prtcl)# timers hellotime 1 holdtime 3`	Configures timers for hellotime and holdtime messages for a redundancy group.
Step 9	**end** **Example:** `Device(config-red-app-prtcl)# end`	Exits redundancy application protocol configuration mode and enters privileged EXEC mode.

Configuring Redundancy Groups for Active/Standby Load Sharing

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **redundancy**
4. **application redundancy**
5. **group** *id*
6. **name** *group-name*
7. **control** *interface-type interface-number* **protocol** *id*
8. **data** *interface-type interface-number*
9. Repeat Steps 3 to 8 to configure another redundancy group.
10. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**redundancy** **Example:** `Device(config)# redundancy`	Enters redundancy configuration mode.
Step 4	**application redundancy** **Example:** `Device(config-red)# application redundancy`	Configures application redundancy and enters redundancy application configuration mode.
Step 5	**group** *id* **Example:** `Device(config-red-app)# group 1`	Configures a redundancy application group and enters redundancy application group configuration mode.
Step 6	**name** *group-name* **Example:** `Device(config-red-app-grp)# name RG1`	Configures a name for the redundancy application group.
Step 7	**control** *interface-type interface-number* **protocol** *id* **Example:** `Device(config-red-app-grp)# control gigabitethernet 0/0/1 protocol 1`	Configures a control interface type and number for the redundancy application group.
Step 8	**data** *interface-type interface-number* **Example:** `Device(config-red-app-grp)# data gigabitethernet 0/2/2`	Configures a data interface type and number for the redundancy application group.
Step 9	Repeat Steps 3 to 8 to configure another redundancy group.	—
Step 10	**end** **Example:** `Device(config-red-app-grp)# end`	Exits redundancy application group configuration mode and enters privileged EXEC mode.

Configuring Redundancy Groups for Active/Active Load Sharing

Perform this task to configure two redundancy groups (RGs) on the same device for active/active load sharing.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **redundancy**
4. **application redundancy**
5. **group** *id*
6. **name** *group-name*
7. **priority** *value* [**failover-threshold** *value*]
8. **control** *interface-type interface-number* **protocol** *id*
9. **data** *interface-type interface-number*
10. **end**
11. **configure terminal**
12. **redundancy**
13. **application redundancy**
14. **group** *id*
15. **name** *group-name*
16. **priority** *value* [**failover-threshold** *value*]
17. **control** *interface-type interface-number* **protocol** *id*
18. **data** *interface-type interface-number*
19. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**redundancy** **Example:** `Device(config)# redundancy`	Enters redundancy configuration mode.
Step 4	**application redundancy** **Example:** `Device(config-red)# application redundancy`	Configures application redundancy and enters redundancy application configuration mode.
Step 5	**group** *id* **Example:** `Device(config-red-app)# group 1`	Configures a redundancy application group and enters redundancy application group configuration mode.

	Command or Action	Purpose
Step 6	**name** *group-name* **Example:** `Device(config-red-app-grp)# name RG1`	Configures a name for the redundancy application group.
Step 7	**priority** *value* [**failover-threshold** *value*] **Example:** `Device(config-red-app-grp)# priority 195 failover-threshold 190`	Specifies a group priority and failover threshold value for the redundancy group.
Step 8	**control** *interface-type interface-number* **protocol** *id* **Example:** `Device(config-red-app-grp)# control gigabitethernet 0/0/1 protocol 1`	Configures a control interface type and number for the redundancy application group.
Step 9	**data** *interface-type interface-number* **Example:** `Device(config-red-app-grp)# data gigabitethernet 0/2/2`	Configures a data interface type and number for the redundancy application group.
Step 10	**end** **Example:** `Device(config-red-app-grp)# end`	Exits redundancy application group configuration mode and enters privileged EXEC mode.
Step 11	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 12	**redundancy** **Example:** `Device(config)# redundancy`	Enters redundancy configuration mode.
Step 13	**application redundancy** **Example:** `Device(config-red)# application redundancy`	Configures application redundancy and enters redundancy application configuration mode.
Step 14	**group** *id* **Example:** `Device(config-red-app)# group 2`	Configures a redundancy application group and enters redundancy application group configuration mode.
Step 15	**name** *group-name* **Example:** `Device(config-red-app-grp)# name RG2`	Configures a name for the redundancy application group.
Step 16	**priority** *value* [**failover-threshold** *value*] **Example:**	Specifies a group priority and failover threshold value for the redundancy group.

	Command or Action	Purpose
	`Device(config-red-app-grp)# priority 205 failover-threshold 200`	
Step 17	**control** *interface-type interface-number* **protocol** *id* **Example:** `Device(config-red-app-grp)# control gigabitethernet 0/0/1 protocol 2`	Configures a control interface type and number for the redundancy application group.
Step 18	**data** *interface-type interface-number* **Example:** `Device(config-red-app-grp)# data gigabitethernet 0/2/2`	Configures a data interface type and number for the redundancy application group.
Step 19	**end** **Example:** `Device(config-red-app-grp)# end`	Exits redundancy application group configuration mode and enters privileged EXEC mode.

Configuring a Traffic Interface for Stateful NAT64 Interchassis Redundancy

This task applies to a LAN-LAN scenario.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **redundancy rii** *id*
5. **redundancy group** *group-id* **ipv6** *ipv6-prefix*/*prefix-length* **exclusive decrement** *value*
6. **exit**
7. **interface** *type number*
8. **redundancy rii** *id*
9. **redundancy group** *group-id* **ipv6** *ipv6-prefix*/*prefix-length* **exclusive decrement** *value*
10. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 0/0/0`	Configures an interface and enters interface configuration mode.
Step 4	**redundancy rii** *id* **Example:** `Device(config-if)# redundancy rii 100`	Configures a redundancy interface identifier (RII) for a redundancy group-protected traffic interfaces.
Step 5	**redundancy group** *group-id* **ipv6** *ipv6-prefix*/*prefix-length* **exclusive decrement** *value* **Example:** `Device(config-if)# redundancy group 1 ipv6 2001:DB8:1::1:100/64 exclusive decrement 50`	Enables IPv6 redundancy.
Step 6	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and enters global configuration mode.
Step 7	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 0/1/1`	Configures an interface and enters interface configuration mode.
Step 8	**redundancy rii** *id* **Example:** `Device(config-if)# redundancy rii 120`	Configures an RII for a redundancy group-protected traffic interfaces.
Step 9	**redundancy group** *group-id* **ipv6** *ipv6-prefix*/*prefix-length* **exclusive decrement** *value* **Example:** `Device(config-if)# redundancy group 1 ipv6 2001:DB8:2::1:100/64 exclusive decrement 50`	Enables IPv6 redundancy.
Step 10	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and enters privileged EXEC mode.

Configuring Static Stateful NAT64 for Interchassis Redundancy

Perform this task to configure a static stateful NAT64 with interchassis redundancy. You can configure interchassis redundancy with the following types of NAT configurations: dynamic, static, and Port Address Translation (PAT) translations.

SUMMARY STEPS

1. **enable**

2. **configure terminal**
3. **ipv6 unicast-routing**
4. **interface** *type number*
5. **ipv6 enable**
6. **ipv6 address** *ipv6-address*/*prefix-length*
7. **nat64 enable**
8. **exit**
9. Repeat Steps 3 to 8 to configure NAT64 on another interface.
10. **nat64 prefix stateful** *ipv6-prefix*/*length*
11. **nat64 v6v4 static** *ipv6-address ipv6-address* [**redundancy** *group-id* **mapping-id** *id*]
12. **nat64 v6v4 tcp** *ipv6-address ipv6-port ipv4-address ipv4-port* [**redundancy** *group-id* **mapping-id** *id*]
13. **end**
14. **show nat64 translations protocol tcp**
15. **show nat64 translations redundancy** *group-id*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 unicast-routing** **Example:** `Device(config)# ipv6 unicast-routing`	Enables the forwarding of IPv6 unicast datagrams.
Step 4	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 0/0/1`	Configures an interface and enters interface configuration mode.
Step 5	**ipv6 enable** **Example:** `Device(config-if)# ipv6 enable`	Enables IPv6 processing on an interface.
Step 6	**ipv6 address** *ipv6-address*/*prefix-length* **Example:** `Device(config-if)# ipv6 address 2001:DB8:1::1/96`	Configures an IPv6 address based on an IPv6 general prefix and enables IPv6 processing on an interface.
Step 7	**nat64 enable** **Example:** `Device(config-if)# nat64 enable`	Enables NAT64 translation on an IPv6 interface.

	Command or Action	Purpose
Step 8	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and enters global configuration mode.
Step 9	Repeat Steps 3 to 8 to configure NAT64 on another interface.	—
Step 10	**nat64 prefix stateful** *ipv6-prefix*/*length* **Example:** `Device(config)# nat64 prefix stateful 2001:DB8:1::1/96`	Defines the stateful NAT64 prefix that is to be added to IPv4 hosts to translate the IPv4 address into an IPv6 address. • The stateful NAT64 prefix can be configured at the global configuration level or at the interface configuration level.
Step 11	**nat64 v6v4 static** *ipv6-address ipv6-address* [**redundancy** *group-id* **mapping-id** *id*] **Example:** `Device(config)# nat64 v6v4 static 2001:DB8:1::FFFE 209.165.201.1 redundancy 1 mapping-id 30`	Enables NAT64 IPv6-to-IPv4 static address mapping and interchassis redundancy.
Step 12	**nat64 v6v4 tcp** *ipv6-address ipv6-port ipv4-address ipv4-port* [**redundancy** *group-id* **mapping-id** *id*] **Example:** `Device(config)# nat64 v6v4 tcp 2001:DB8:1::1 redundancy 1 mapping-id 1`	Applies static mapping to TCP protocol packets and enables interchassis redundancy.
Step 13	**end** **Example:** `Device(config)# end`	Exits global configuration mode and enters privileged EXEC mode.
Step 14	**show nat64 translations protocol tcp** **Example:** `Device# show nat64 translations protocol tcp`	Displays information about NAT 64 protocol translations.
Step 15	**show nat64 translations redundancy** *group-id* **Example:** `Device# show nat64 translations redundancy 1`	Displays information about NAT64 redundancy translations.

Example:

The following is sample output from the **show nat64 translations protocol tcp** command:

```
Device# show nat64 translations protocol tcp

 Proto  Original IPv4         Translated IPv4
        Translated IPv6       Original IPv6
 ------------------------------------------------------------------------------
```

```
tcp    209.165.201.2:21      [2001:DB8:1::103]:32847
       10.2.1.1:80           [2001::11]:80
tcp    209.165.201.2:21      [2001:DB8:1::104]:32848
       10.2.1.1:80           [2001::11]:80

Total number of translations: 2
```

The following is sample output from the **show nat64 translations redundancy** command:

```
Device# show nat64 translations redundancy 1

Proto  Original IPv4         Translated IPv4
       Translated IPv6       Original IPv6
------------------------------------------------------------------------------

       209.165.201.2:21      [2001:DB8:1::103]:32847

tcp    10.2.1.11:32863       [2001::3201:10b]:32863
       10.1.1.1:80           [2001::11]:80
tcp    209.165.201.2:21      [2001:DB8:1::104]:32848
       10.1.1.1:80           [2001::11]:80

Total number of translations: 3
```

Configuration Examples for Stateful Network Address Translation 64 Interchassis Redundancy

Example: Configuring Redundancy Group Protocols

```
Device# configure terminal
Device(config)# redundancy
Device(red)# application redundancy
Device(config-red-app)# protocol 1
Device(config-red-app-prtcl)# name RG1
Device(config-red-app-prtcl)# timers hellotime 1 holdtime 3
Device(config-red-app-prtcl)# end
Device# configure terminal
Device(config)# redundancy
Device(red)# application redundancy
Device(config-red-app)# protocol 2
Device(config-red-app-prtcl)# name RG1
Device(config-red-app-prtcl)# end
```

Example: Configuring Redundancy Groups for Active/Standby Load Sharing

The following example shows how to configure redundancy groups (RGs) on two devices for active/standby load sharing:

```
Device1# configure terminal
Device1(config)# redundancy
Device1(config-red)# application redundancy
Device1(config-red-app)# group 1
Device1(config-red-app-grp)# name RG1
```

```
Device1(config-red-app-grp)# control gigabitethernet 0/0/1 protocol 1
Device1(config-red-app-grp)# data gigabitethernet 0/2/2
Device1(config-red-app-grp)# end

Device2# configure terminal
Device2(config)# redundancy
Device2(config-red)# application redundancy
Device2(config-red-app)# group 1
Device2(config-red-app-grp)# name RG1
Device2(config-red-app-grp)# control gigabitethernet 0/0/1 protocol 1
Device2(config-red-app-grp)# data gigabitethernet 0/2/2
Device2(config-red-app-grp)# end
```

Example: Configuring Redundancy Groups for Active/Active Load Sharing

The following example shows how to configure two redundancy groups (RGs) on the same device for active/active load sharing:

```
Device1# configure terminal
Device1(config)# redundancy
Device1(config-red)# application redundancy
Device1(config-red-app)# group 1
Device1(config-red-app-grp)# name RG1
Device1(config-red-app-grp)# priority 195 failover-threshold 190
Device1(config-red-app-grp)# control gigabitethernet 0/0/1 protocol 1
Device1(config-red-app-grp)# data gigabitethernet 0/2/2
Device1(config-red-app-grp)# end
Device1# configure terminal
Device1(config)# redundancy
Device1(config-red)# application redundancy
Device1(config-red-app)# group 2
Device1(config-red-app-grp)# name RG2
Device1(config-red-app-grp)# priority 205 failover-threshold 200
Device1(config-red-app-grp)# control gigabitethernet 0/0/1 protocol 2
Device1(config-red-app-grp)# data gigabitethernet 0/2/2
Device1(config-red-app-grp)# end

Device2# configure terminal
Device2(config)# redundancy
Device2(config-red)# application redundancy
Device2(config-red-app)# group 1
Device2(config-red-app-grp)# name RG1
Device2(config-red-app-grp)# priority 195 failover-threshold 190
Device2(config-red-app-grp)# control gigabitethernet 0/0/1 protocol 1
Device2(config-red-app-grp)# data gigabitethernet 0/2/2
Device2(config-red-app-grp)# end
Device2# configure terminal
Device2(config)# redundancy
Device2(config-red)# application redundancy
Device2(config-red-app)# group 2
Device2(config-red-app-grp)# name RG2
Device2(config-red-app-grp)# priority 205 failover-threshold 200
Device2(config-red-app-grp)# control gigabitethernet 0/0/1 protocol 2
Device2(config-red-app-grp)# data gigabitethernet 0/2/2
Device2(config-red-app-grp)# end
```

Example: Configuring a Traffic Interface for Stateful NAT64 Interchassis Redundancy

```
Device# configure terminal
Device(config)# interface gigabitethernet 0/0/0
Device(config-if)# redundancy rii 100
Device(config-if)# redundancy group 1 ipv6 2001:DB8:1::1:100/64 exclusive decrement 50
Device(config-if)# exit
Device(config)# interface gigabitethernet 0/1/1
Device(config-if)# redundancy rii 100
Device(config-if)# redundancy group 1 ipv6 2001:DB8::2:1:100/64 exclusive decrement 50
Device(config-if)# end
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Master Command List, All Releases
NAT commands	IP Addressing Services Command Reference

Standards/RFCs

Standard/RFC	Title
RFC 4787	*Network Address Translation (NAT) Behavioral Requirements for Unicast UDP*

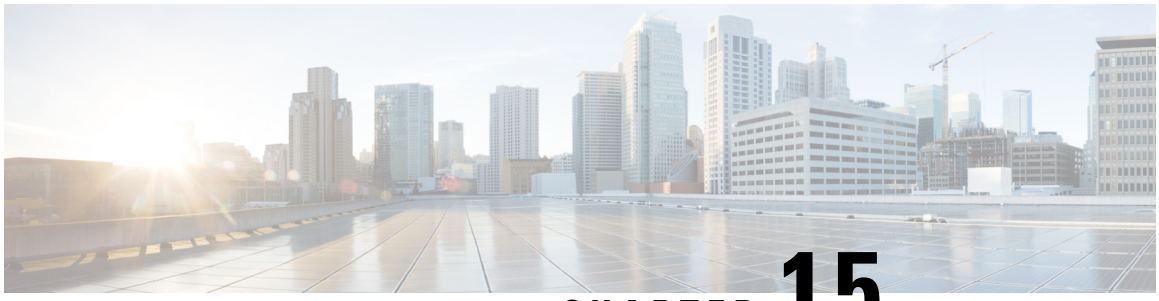
Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Stateful Network Address Translation 64 Interchassis Redundancy

Table 20: Feature Information for Stateful Network Address Translation 64 Interchassis Redundancy

Feature Name	Releases	Feature Information
Stateful Network Address Translation 64 Interchassis Redundancy	Cisco IOS XE Release 3.7S	The Stateful Network Address Translation 64 Interchassis Redundancy feature adds interchassis redundancy support to stateful Network Address Translation 64 (NAT64). The stateful interchassis redundancy enables you to configure pairs of devices to act as backups for each other. The following commands were introduced or modified: **clear nat64 translations**, **nat64 v4v6**, **nat64 v6v4**, **redundancy group (interface)**, **show nat64**, **show nat64 translations redundancy**.



CHAPTER 15

Mapping of Address and Port Using Translation

The Mapping of Address and Port Using Translation feature provides connectivity to IPv4 hosts across IPv6 domains. Mapping of address and port using translation (MAP-T) is a mechanism that performs double translation (IPv4 to IPv6 and vice versa) on customer edge (CE) devices and border routers.

This module provides an overview of MAP-T and explains how to configure this feature.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for Mapping of Address and Port Using Translation

- The mapping of address and port using translation (MAP-T) customer edge (CE) functionality is not supported.
- In Cisco IOS XE Denali 16.2 release, the support for MAP-T domains were extended to 10000 domains. For releases prior to Cisco IOS XE Denali 16.2, a maximum of 128 MAP-T domains are supported.
- Forwarding mapping rule (FMR) is not supported.

Information About Mapping of Address and Port Using Translation

Mapping of Address and Port Using Translation Overview

The Mapping of Address and Port Using Translation feature provides connectivity to IPv4 hosts across IPv6 domains. Mapping of address and port using translation (MAP-T) builds on the existing stateless IPv4 and IPv6 address translation techniques that are specified in RFCs 6052, 6144, and 6145.

MAP-T is a mechanism that performs double translation (IPv4 to IPv6 and vice versa) on customer edge (CE) devices and border routers. The Mapping of Address and Port Using Translation feature supports only the MAP-T border router functionality. This feature does not support the MAP-T CE functionality.

The Mapping of Address and Port Using Translation feature leverages the Network Address Translation 64 (NAT64) translation engine and adds the MAP-T border router function to the NAT64 stateless function. MAP-T is enabled on IPv4 and IPv6 interfaces. MAP-T uses IPv4 and IPv6 forwarding, IPv4 and IPv6 fragmentation functions, and NAT64 translation functions. A MAP-T domain is one or more MAP CE devices and a border router, all connected to the same IPv6 network.

A MAP-T CE device connects a user's private IPv4 address and the native IPv6 network to the IPv6-only MAP-T domain. The MAP-T border router uses the stateless IPv4/IPv6 translation to connect external IPv4 networks to all devices available in the one or more MAP-T domains. MAP-T requires only one IPv6 prefix per network and supports the regular IPv6 prefix/address assignment mechanisms. The MAP-T domain contains regular IPv6-only hosts or servers that have an IPv4-translatable IPv6 address. MAP-T does not require the operation of an IPv4 overlay network or the introduction of a non-native-IPv6 network device or server functionality.

A MAP-T configuration provides the following features:

- Retains the ability for IPv4 end hosts to communicate across the IPv6 domain with other IPv4 hosts.
- Permits both individual IPv4 address assignment and IPv4 address sharing with a predefined port range.
- Allows communication between IPv4-only and IPv6-enabled end hosts and native IPv6-only servers in domains that use IPv4-translatable IPv6 addresses.
- Allows the use of IPv6 native network operations, including the ability to classify IP traffic and perform IP traffic routing optimization policies such as routing optimization based on peering policies for IPv4 destinations outside the domain.

MAP-T Mapping Rules

Mapping rules define the mapping between an IPv4 prefix and an IPv4 address or between a shared IPv4 address and an IPv6 prefix/address. Each mapping of address and port using translation (MAP-T) domain uses a different mapping rule.

A MAP-T configuration has one basic mapping rule (BMR), one default mapping rule (DMR), and one or more forwarding mapping rules (FMRs) for each MAP-T domain. You must configure the DMR before configuring the BMR for a MAP-T domain.

The three types of mapping rules are described below:

- A BMR configures the MAP IPv6 address or prefix. The basic mapping rule is configured for the source address prefix. You can configure only one basic mapping rule per IPv6 prefix. The basic mapping rule is used by the MAP-T CE to configure itself with an IPv4 address, an IPv4 prefix, or a shared IPv4 address from an IPv6 prefix. The basic mapping rule can also be used for forwarding packets, where an IPv4 destination address and a destination port are mapped into an IPv6 address/prefix. Every MAP-T node (a CE device is a MAP-T node) must be provisioned with a basic mapping rule. You can use the **port-parameters** command to configure port parameters for the MAP-T BMR.
- A DMR is a mandatory rule that is used for mapping IPv4 information to IPv6 addresses for destinations outside a MAP-T domain. A 0.0.0.0/0 entry is automatically configured in the MAP rule table (MRT) for this rule.
- An FMR is used for forwarding packets. Each FMR results in an entry in the MRT for the rule IPv4 prefix. FMR is an optional rule for mapping IPv4 and IPv6 destinations within a MAP-T domain.

Note FMR is not supported by the Mapping of Address and Port Using Translation feature.

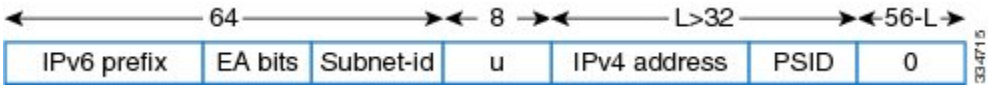
MAP-T Address Formats

The mapping of address and port using translation (MAP-T) customer edge (CE) device address format is defined by the IETF draft Mapping of Address and Port (MAP). Address formats are used during mapping rule operations to construct the source and destination IPv6 addresses.

Note Forwarding mapping rule (FMR) is not supported by the Mapping of Address and Port Using Translation feature.

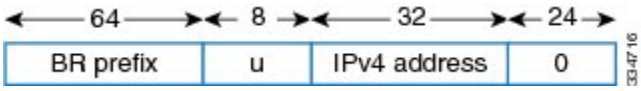
The figure below shows the mapped CE address format as defined in MAP-T configuration. This address format is used in basic mapping rule (BMR) and FMR operations.

Figure 17: IPv4-Translatable Address for BMR and FMR



The figure below shows the address format used by the MAP-T default mapping rule (DMR), an IPv4-translated address that is specific to MAP-T configuration.

Figure 18: IPv4-Translated Address for DMR



Packet Forwarding in MAP-T Customer Edge Devices

Note The Mapping of Address and Port Using Translation feature does not support the MAP-T customer edge (CE) functionality. The CE functionality is provided by third-party devices.

IPv4-to-IPv6 Packet Forwarding

A mapping of address and port using translation (MAP-T) CE device that receives IPv4 packets performs Network Address Translation (NAT) and creates appropriate NAT stateful bindings. The resulting IPv4 packets contain the source IPv4 address and the source transport number defined by MAP-T. This IPv4 packet is forwarded to the CE's MAP-T, which performs IPv4-to-IPv6 stateless translation. IPv6 source and destination addresses are then derived by the MAP-T translation, and IPv4 headers are replaced with IPv6 headers.

IPv6-to-IPv4 Packet Forwarding

A MAP-T CE device that receives an IPv6 packet performs its regular IPv6 operations. Only the packets that are addressed to the basic mapping rule (BMR) address are sent to the CE's MAP-T. All other IPv6 traffic is forwarded based on the IPv6 routing rules on the CE device. The CE device checks if the transport-layer destination port number of the packets received from MAP-T is in the range that was configured and forwards packets that confirm to the port number. The CE device drops all nonconforming packets and responds with an Internet Control Message Protocol Version 6 (ICMPv6) "Address Unreachable" message.

Packet Forwarding in Border Routers

IPv4-to-IPv6 Packet Forwarding

An incoming IPv4 packet is processed by the IPv4 input interface, and the destination route lookup routes the IPv4 packet to the mapping of address and port using translation (MAP-T) virtual interface. The border router compares the packet against the IPv4 prefix lookup unit (PLU) tree to obtain the corresponding basic mapping rule (BMR), the default mapping rule (DMR), and the forwarding mapping rule (FMR). Based on the BMR or FMR rules, the border router constructs the IPv6 destination address by encoding the embedded address (EA) bits and adding a suffix. The IPv6 source address is constructed from the DMR rule.

After the IPv6 source and destination addresses are constructed, the packet uses the Network Address Translation 64 (NAT64) IPv4-to-IPv6 translation to construct the IPv6 packet. A routing lookup is done on the IPv6 packet, and the packet is forwarded to the IPv6 egress interface for processing and transmission.

IPv6-to-IPv4 Packet Forwarding

An incoming IPv6 packet is processed by the IPv6 input interface, and the destination route lookup routes the IPv6 packet to the MAP-T virtual interface. The software compares the packet against the IPv6 PLU tree to obtain the corresponding BMR, DMR, and FMR rules. The border router checks whether the port-set ID (PSID) and the port set match. If the port-set ID and port set match, the DMR rule matches the packet destination of the IPv6 packet. Based on the BMR and FMR, the border router constructs the IPv4 source address and extracts the IPv4 destination address from the IPv6 destination address. The IPv6 packet uses the NAT64 IPv6-to-IPv4 translation engine to construct the IPv4 packet from the IPv6 packet. A routing lookup is done on the IPv4 packet, and the IPv4 packet is forwarded to the IPv4 egress interface for processing and transmission.

ICMP/ICMPv6 Header Translation for MAP-T

Mapping of address and port using translation (MAP-T) customer edge (CE) devices and border routers use the ICMP/ICMPv6 translation for address sharing of port ranges.

Unlike TCP and UDP, which provide two port fields to represent source and destination addresses, the Internet Control Message Protocol (ICMP) and ICMP Version 6 (ICMPv6) query message headers have only one ID field.

When an ICMP query message originates from an IPv4 host that exists beyond a MAP-T CE device, the ICMP ID field is exclusively used to identify the IPv4 host. The MAP-T CE device rewrites the ID field to a port-set value that is obtained through the basic mapping rule (BMR) during the IPv4-to-IPv6 translation, and the border router translates ICMPv6 packets to ICMP.

When a MAP-T border router receives an ICMP packet that contains an ID field that is bound for a shared address in the MAP-T domain, the MAP-T border router uses the ID field as a substitute for the destination port to determine the IPv6 destination address. The border router derives the destination IPv6 address by mapping the destination IPv4 address without the port information for packets that do not contain the ID field, and the corresponding CE device translates the ICMPv6 packets to ICMP.

Path MTU Discovery and Fragmentation in MAP-T

Mapping of address and port using translation (MAP-T) uses path maximum transmission unit (MTU) discovery and fragmentation for IPv4-to-IPv6 translation because the size of IPv4 (more than 20 octets) and IPv6 (40 octets) headers is different. The MTU defines the largest size of a packet that an interface can transmit without the need to fragment the packet. IP packets larger than the MTU must go through IP fragmentation procedures.

When an IPv4 node performs path MTU discovery by setting the Don't Fragment (DF) bit in the packet header, path MTU discovery operates end-to-end across the MAP-T border router and customer edge (CE) translators. During IPv4 path MTU discovery, either the IPv4 device or the IPv6 device can send ICMP “Packet Too Big” messages to the sender. When IPv6 devices send these messages as Internet Control Message Protocol Version 6 (ICMPv6) errors, the packets that follow the message pass through the translator and result in an appropriate ICMP error message sent to the IPv4 sender.

When the IPv4 sender does not set the DF bit, the translator fragments the IPv4 packet and includes the packet with fragment headers to fit the packet in the minimum MTU 1280-byte IPv6 packets. When packets are fragmented, either by the sender or by IPv4 devices, the low-order 16 bits of the fragment identification are carried end-to-end across the MAP-T domain to ensure that packets are reassembled correctly.

How to Configure Mapping of Address and Port Using Translation

Configuring Mapping of Address and Port Using Translation

Before you begin

Prerequisites:

- Configure the **ipv6 enable** command on interfaces on which you configure the Mapping of Address and Port Using Translation feature.

- Configure the default mapping rule before you configure the basic mapping rule.
- While configuring mapping of address and port using translation (MAP-T), the default mapping rule (DMR) prefix, the IPv6 user prefix, and the IPv6 prefix plus the embedded address (EA) bits must be less than or equal to 64 bits, and the share ratio plus the contiguous ports plus the start port must be 16 bits.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **nat64 map-t domain** *number*
4. **default-mapping-rule** *ipv6-prefix*/*prefix-length*
5. **basic-mapping-rule**
6. **ipv6-prefix** *prefix*/*length*
7. **ipv4-prefix** *prefix*/*length*
8. **port-parameters share-ratio** *ratio* [**start-port** *port-number*]
9. **end**
10. **show nat64 map-t domain** *number*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter you password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**nat64 map-t domain** *number* **Example:** `Device(config)# nat64 map-t domain 1`	Configures the Network Address Translation 64 (NAT64) mapping of address and port using translation (MAP-T) domain and enters NAT64 MAP-T configuration mode.
Step 4	**default-mapping-rule** *ipv6-prefix*/*prefix-length* **Example:** `Device(config-nat64-mapt)# default-mapping-rule 2001:DA8:B001:FFFF::/64`	Configures the default domain mapping rule for the MAP-T domain.
Step 5	**basic-mapping-rule** **Example:** `Device(config-nat64-mapt)# basic-mapping-rule`	Configures the basic mapping rule (BMR) for the MAP-T domain and enters NAT64 MAP-T BMR configuration mode.
Step 6	**ipv6-prefix** *prefix*/*length* **Example:**	Configures an IPv6 address and prefix for the MAP-T BMR.

	Command or Action	Purpose
	`Device(config-nat64-mapt-bmr)# ipv6-prefix 2001:DA8:B001::/56`	
Step 7	**ipv4-prefix** *prefix/length* **Example:** `Device(config-nat64-mapt-bmr)# ipv4-prefix 202.1.0.128/28`	Configures an IPv4 address and prefix for the MAP-T BMR.
Step 8	**port-parameters share-ratio** *ratio* [**start-port** *port-number*] **Example:** `Device(config-nat64-mapt-bmr)# port-parameters share-ratio 16 start-port 1024`	Configures port parameters for the MAP-T BMR.
Step 9	**end** **Example:** `Device(config-nat64-mapt-bmr)# end`	Exits NAT64 MAP-T BMR configuration mode and returns to privileged EXEC mode.
Step 10	**show nat64 map-t domain** *number* **Example:** `Device# show nat64 map-t domain 1`	Displays MAP-T domain information.

Example:

The following is sample output from the **show nat64 map-t domain** command:

```
Device# show nat64 map-t domain 1

MAP-T Domain 1
Mode MAP-T
Default-mapping-rule
Ip-v6-prefix 2001:DA8:B001:FFFF::/64
Basic-mapping-rule
Ip-v6-prefix 2001:DA8:B001::/56
Ip-v4-prefix 202.1.0.128/28
Port-parameters
Share-ratio 16 Contiguous-ports 64 Start-port 1024
Share-ratio-bits 4 Contiguous-ports-bits 6 Port-offset-bits 6
```

Configuration Examples for Mapping of Address and Port Using Translation

Example: Configuring Mapping of Address and Port Using Translation

```
Device# configure terminal
Device(config)# nat64 map-t domain 1
```

```
Device(config-nat64-mapt)# $ping-rule 2001:DA8:B001:FFFF::/64
Device(config-nat64-mapt)# basic-mapping-rule
Device(config-nat64-mapt-bmr)# ipv6-prefix 2001:DA8:B001::/56
Device(config-nat64-mapt-bmr)# ipv4-prefix 202.1.0.128/28
Device(config-nat64-mapt-bmr)# $ters share-ratio 16 start-port 1024
Device(config-nat64-mapt-bmr)# end
```

Example: MAP-T Deployment Scenario

The following illustration shows a mapping of address and port using translation (MAP-T) deployment scenario.

The following is the configuration for the MAP-T deployment scenario:

```
Device(config)# nat64 map-t
Device(config)# nat64 map-t domain 1
Device(config-nat64-mapt)# $ping-rule 2001:DA8:B001:FFFF::/64
Device(config-nat64-mapt)# basic-mapping-rule
Device(config-nat64-mapt-bmr)# ipv6-prefix 2001:DA8:B001::/56
Device(config-nat64-mapt-bmr)# ipv4-prefix 202.1.0.128/28
Device(config-nat64-mapt-bmr)# $ters share-ratio 16 start-port 1024
Device(config-nat64-mapt-bmr)# end
```

At the PC:

An IPv4 packet goes from 202.1.0.130 to 11.1.1.1. At the customer edge (CE) device the Mapping of address and port mapping using translation (MAP-T) function translates the packet to Src: 2201:DA8:B001:2E:0:CA01:82:E00 Dest: 2001:DA8:B001:FFFF:B:0101:0100:0.

At the border router the MAP-T border router translates the packet to

Packet goes from 192.168.1.2 ---> 74.1.1.1, source 4000, destination port : 5000

At the CPE the MAP-T CE function translates the

packet to Src: 2201:DA8:B001:2E:0:CA01:82:E00 Dest: 2001:DA8:B001:FFFF:B:0101:0100:0.

At the BR the MAP-T BR function translates the packet to

Src:203.38.102.130 Dst:74.1.1.1 SrcPort:4000 DstPort:5000

From End device:

Src:74.1.1.1 Dst:203.38.102.130 SrcPort:4000 DstPort:5000

At the BR the MAP-T BR function translates the packet to

Src: 2201:DA8:B001:2E:0:CA01:82:E00 Dest: 2001:DA8:B001:FFFF:B:0101:0100:0.

At the CE the MAP-T CE function translates the packet from

Src: 2201:DA8:B001:2E:0:CA01:82:E00 Dest: 2001:DA8:B001:FFFF:B:0101:0100:0.

To

Src:74.1.1.1 Dst:203.38.102.130 SrcPort:4000 Dstport:5000

Additional References for Mapping of Address and Port Using Translation

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
NAT commands	Cisco IOS IP Addressing Services Command Reference

Standards and RFCs

Standard/RFC	Title
MAP	Mapping of Address and Port (MAP)
MAP Translation	MAP Translation (MAP-T) - specification
RFC 6052	IPv6 Addressing of IPv4/IPv6 Translators
RFC 6144	Framework for IPv4/IPv6 Translation
RFC 6145	IP/ICMP Translation Algorithm

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Mapping of Address and Port Using Translation

Table 21: Feature Information for Mapping of Address and Port Using Translation

Feature Name	Releases	Feature Information
Mapping of Address and Port Using Translation	Cisco IOS XE Release 3.8S Cisco IOS XE Release 3.10S Cisco IOS XE Denali 16.2	The Mapping of Address and Port Using Translation feature provides connectivity to IPv4 hosts across IPv6 domains. MAP-T is a mechanism that performs double translation (IPv4 to IPv6 and vice versa) on CE devices and border routers. In Cisco IOS XE Release 3.10S, support was added for the Cisco CSR 1000V Series Routers. In Cisco IOS XE Denali 16.2, support for MAP-T domains were extended to 10000 domains. The following commands were introduced or modified: **basic-mapping-rule**, **default-mapping-rule**, **ipv4-prefix**, **ipv6-prefix**, **mode (nat64)**, **nat64 map-t domain**, **port-parameters**, and **show nat64 map-t**.

Glossary

EA bits—Embedded address bits. The IPv4 EA bits in the IPv6 address identify an IPv4 prefix/address (or part thereof) or a shared IPv4 address (or part thereof) and a port-set identifier.

IP fragmentation—The process of breaking a datagram into a number of pieces that can be reassembled later. The IP source, destination, identification, total length, and fragment offset fields, along with the More fragments and Don't Fragment (DF) flags in the IP header, are used for IP fragmentation and reassembly. A DF bit is a bit within the IP header that determines whether a device is allowed to fragment a packet.

IPv4-translatable address—IPv6 addresses that are used to represent IPv4 hosts. These addresses have an explicit mapping relationship to IPv6 addresses. This relationship is self-described by mapping the IPv4 address in the IPv6 address. Both stateless and stateful translators use IPv4-translatable (also called IPv4-converted) IPv6 addresses to represent IPv4 hosts.

IPv6-translatable address—IPv6 addresses that are assigned to IPv6 hosts for stateless translation. These IPv6-translatable addresses (also called IPv6-converted addresses) have an explicit mapping relationship to IPv4 addresses. This relationship is self-described by mapping the IPv4 address in the IPv6 address. The stateless translator uses corresponding IPv4 addresses to represent IPv6 hosts. The stateful translator does not use IPv6-translatable addresses because IPv6 hosts are represented by the IPv4 address pool in the translator via dynamic states.

MAP rule—A set of parameters that define the mapping between an IPv4 prefix, an IPv4 address or a shared IPv4 address, and an IPv6 prefix or address. Each MAP domain uses a different mapping rule set.

MAP-T border router—A mapping of address and port using translation (MAP-T)-enabled router or translator at the edge of a MAP domain that provides connectivity to the MAP-T domain. A border relay router has at least one IPv6-enabled interface and one IPv4 interface connected to the native IPv4 network, and this router can serve multiple MAP-T domains.

MAP-T CE—A device that functions as a customer edge (CE) router in a MAP-T deployment. A typical MAP-T CE device that adopts MAP rules serves a residential site with one WAN-side interface and one or more LAN-side interfaces. A MAP-T CE device can also be referred to as a "CE" within the context of a MAP-T domain.

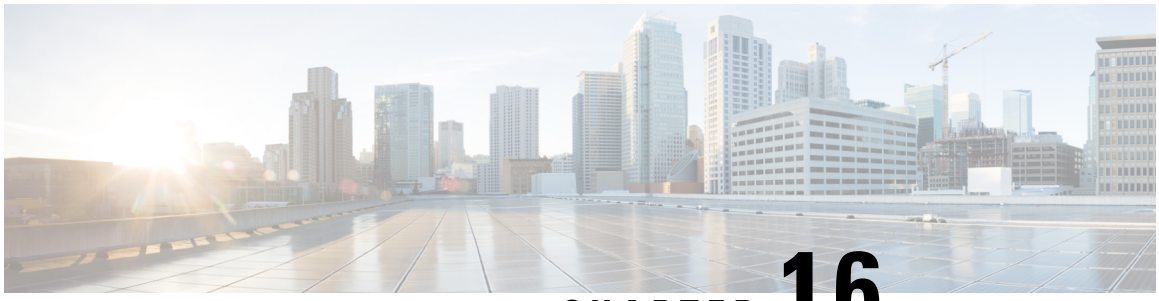
MAP-T domain—Mapping of address and port using translation (MAP-T) domain. One or more customer edge (CE) devices and a border router, all connected to the same IPv6 network. A service provider may deploy a single MAP-T domain or use multiple MAP domains.

MRT—MAP rule table. Address and port-aware data structure that supports the longest match lookups. The MRT is used by the MAP-T forwarding function.

path MTU—Path maximum transmission unit (MTU) discovery prevents fragmentation in the path between endpoints. Path MTU discovery is used to dynamically determine the lowest MTU along the path from a packet's source to its destination. Path MTU discovery is supported only by TCP and UDP. Path MTU discovery is mandatory in IPv6, but it is optional in IPv4. IPv6 devices never fragment a packet—only the sender can fragment packets.

stateful translation—Creates a per-flow state when the first packet in a flow is received. A translation algorithm is said to be stateful if the transmission or reception of a packet creates or modifies a data structure in the relevant network element. Stateful translation allows the use of multiple translators interchangeably and also some level of scalability. Stateful translation enables IPv6 clients and peers without mapped IPv4 addresses to connect to IPv4-only servers and peers.

stateless translation—A translation algorithm that is not stateful. A stateless translation requires configuring a static translation table or may derive information algorithmically from the messages that it is translating. Stateless translation requires less computational overhead than stateful translation. It also requires less memory to maintain the state because the translation tables and the associated methods and processes exist in a stateful algorithm and do not exist in a stateless one. Stateless translation enables IPv4-only clients and peers to initiate connections to IPv6-only servers or peers that are equipped with IPv4-embedded IPv6 addresses. It also enables scalable coordination of IPv4-only stub networks or ISP IPv6-only networks. Because the source port in an IPv6-to-IPv4 translation may have to be changed to provide adequate flow identification, the source port in the IPv4-to-IPv6 direction need not be changed.



CHAPTER 16

Disabling Flow Cache Entries in NAT and NAT64

The Disabling Flow Cache Entries in NAT and NAT64 feature allows you to disable flow cache entries for dynamic and static Network Address Translation (NAT) translations. Disabling flow cache entries for dynamic and static translations saves memory usage and helps in the scaling of NAT translations.

Note Disabling flow cache entries results in lesser performance as this functionality does multiple database searches to find the most specific translation to use.

This module describes the feature and explains how to configure it.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for Disabling Flow Cache Entries in NAT and NAT64

- You cannot disable flow cache entries in interface overload configuration because session entries are created even if flow entry creation is disabled.
- Flow cache entries are created for application layer gateway (ALG) traffic because flow-specific information needs to be stored in the session entry for ALG traffic.

Information About Disabling Flow Cache Entries in NAT and NAT64

Disabling of Flow Cache Entries Overview

By default, Network Address Translation (NAT) creates a session (which is a 5-tuple entry) for every translation. A session is also called a flow cache entry. Flow cache entries create a NAT translation for every Internet Control Message Protocol (ICMP), TCP, and UDP flow and, hence, consume a lot of system memory.

Port Address Translation (PAT) or interface overload configurations must have flow cache entries enabled. However, dynamic and static NAT configurations can disable flow cache entries. Instead of creating sessions, dynamic and static NAT translations can translate a packet off the binding (or bindings if both inside and outside bindings are available). A binding or a half entry is an association between a local IP address and a global IP address.

Note NAT, NAT64 (stateful and stateless), and carrier-grade NAT (CGN) translations support the disabling of flow cache entries.

When flow cache entry is enabled and a user has 100 sessions, 1 bind and 100 session are created. However, when flow cache entry is disabled, only one single bind is created for these sessions. Disabling flow cache entries for dynamic and static translations saves memory usage and provides more scalability for your dynamic or static translations.

Note Disabling flow cache entries will result in lesser performance as this functionality performs multiple database searches to find the most specific translation to use.

When a packet is received for translation, the following processing happens:

- If your NAT configuration is PAT, the configuration to disable flow cache entries is ignored and the packet is processed normally.
- If your configuration is not PAT, the following processing happens:
  - If the packet is an application layer gateway (ALG) packet, a session is created.
  - If the packet is a non-ALG packet, a temporary session is created and this session is sent for translation. The packet is sent to Layer 3 or Layer 4 if your configuration is NAT or to Layer 4 or Layer 7 if your configuration is NAT64 (stateful or stateless).

How to Disable Flow Cache Entries in NAT and NAT64

Disabling Flow Cache Entries in Dynamic NAT

Flow cache entries are enabled by default when Network Address Translation (NAT) is configured. To disable flow cache entries, use the **no ip nat create flow-entries** command. Perform this task to disable flow cache entries in the dynamic translation of inside source address.

Note Port Address Translation (PAT) or interface overload configuration, which is a type of dynamic NAT, requires flow cache entries. You cannot disable flow cache entries for PAT configurations.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip nat pool** *name start-ip end-ip* {**netmask** *netmask* | **prefix-length** *prefix-length*}
4. **access-list** *access-list-number* **permit** *source source-wildcard*
5. **ip nat inside source list** *access-list-number* **pool** *name*
6. **no ip nat create flow-entries**
7. **interface** *type number*
8. **ip address** *ip-address mask*
9. **ip nat inside**
10. **exit**
11. **interface** *type number*
12. **ip address** *ip-address mask*
13. **ip nat outside**
14. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip nat pool** *name start-ip end-ip* {**netmask** *netmask* | **prefix-length** *prefix-length*} **Example:**	Defines a pool of global addresses to be allocated as needed.

	Command or Action	Purpose
	`Device(config)# ip nat pool net-208 172.16.233.208 172.16.233.223 prefix-length 28`	
Step 4	**access-list** *access-list-number* **permit** *source source-wildcard* **Example:** `Device(config)# access-list 1 permit 192.168.34.0 0.0.0.255`	Defines a standard access list that permits IP addresses that are to be translated.
Step 5	**ip nat inside source list** *access-list-number* **pool** *name* **Example:** `Device(config)# ip nat inside source list 1 pool net-208`	Establishes a dynamic source translation by specifying the pool and the access list specified in Steps 3 and 4, respectively.
Step 6	**no ip nat create flow-entries** **Example:** `Device(config)# no ip nat create flow-entries`	Disables the creation of flow cache entries.
Step 7	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 0/0/1`	Specifies an interface and enters interface configuration mode.
Step 8	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 10.114.11.39 255.255.255.0`	Sets a primary IP address for the interface.
Step 9	**ip nat inside** **Example:** `Device(config-if)# ip nat inside`	Connects the interface to the inside network, which is subject to NAT.
Step 10	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 11	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 0/1/1`	Specifies an interface and enters interface configuration mode.
Step 12	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 172.16.232.182 255.255.255.240`	Sets a primary IP address for an interface.
Step 13	**ip nat outside** **Example:** `Device(config-if)# ip nat outside`	Connects an interface to the outside network.

	Command or Action	Purpose
Step 14	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Disabling Flow Cache Entries in Static NAT64

Flow cache entries are enabled by default in NAT. Perform the following task to disable flow entries in your stateful Network Address Translation 64 (NAT64) configuration.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 unicast-routing**
4. **interface** *type number*
5. **description** *string*
6. **ipv6 enable**
7. **ipv6 address** {*ipv6-address*/*prefix-length* | *prefix-name sub-bits*/*prefix-length*}
8. **nat64 enable**
9. **exit**
10. **interface** *type number*
11. **description** *string*
12. **ip address** *ip-address mask*
13. **nat64 enable**
14. **exit**
15. **nat64 prefix stateful** *ipv6-prefix*/*length*
16. **nat64 v6v4 static** *ipv6-address ipv4-address*
17. **nat64 settings flow-entries disable**
18. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 unicast-routing** **Example:**	Enables the forwarding of IPv6 unicast datagrams.

	Command or Action	Purpose
	`Device(config)# ipv6 unicast-routing`	
Step 4	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 0/0/0`	Specifies an interface type and enters interface configuration mode.
Step 5	**description** *string* **Example:** `Device(config-if)# description interface facing ipv6`	Adds a description to an interface configuration.
Step 6	**ipv6 enable** **Example:** `Device(config-if)# ipv6 enable`	Enables IPv6 processing on an interface.
Step 7	**ipv6 address** {*ipv6-address*/*prefix-length* | *prefix-name sub-bits*/*prefix-length*} **Example:** `Device(config-if)# ipv6 address 2001:DB8:1::1/96`	Configures an IPv6 address based on an IPv6 general prefix and enables IPv6 processing on an interface.
Step 8	**nat64 enable** **Example:** `Device(config-if)# nat64 enable`	Enables NAT64 translation on an IPv6 interface.
Step 9	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 10	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 1/2/0`	Specifies an interface type and enters interface configuration mode.
Step 11	**description** *string* **Example:** `Device(config-if)# description interface facing ipv4`	Adds a description to an interface configuration.
Step 12	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 209.165.201.1 255.255.255.0`	Configures an IPv4 address for an interface.
Step 13	**nat64 enable** **Example:** `Device(config-if)# nat64 enable`	Enables NAT64 translation on an IPv4 interface.

	Command or Action	Purpose
Step 14	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 15	**nat64 prefix stateful** *ipv6-prefix/length* **Example:** `Device(config)# nat64 prefix stateful 2001:DB8:1::1/96`	Defines the stateful NAT64 prefix to be added to IPv4 hosts to translate the IPv4 address into an IPv6 address. • The stateful NAT64 prefix can be configured in global configuration mode or in interface mode.
Step 16	**nat64 v6v4 static** *ipv6-address ipv4-address* **Example:** `Device(config)# nat64 v6v4 static 2001:DB8:1::FFFE 209.165.201.1`	Enables NAT64 IPv6-to-IPv4 static address mapping.
Step 17	**nat64 settings flow-entries disable** **Example:** `Device(config)# nat64 settings flow-entries disable`	Disables flow cache entries in the NAT64 configuration.
Step 18	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Disabling Flow Cache Entries in Static CGN

Flow cache entries are enabled by default when Network Address Translation (NAT) is configured. Perform this task to disable flow cache entries in a static carrier-grade NAT (CGN) configuration.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip nat settings mode cgn**
4. **ip nat inside source static** *local-ip global-ip*
5. **no ip nat create flow-entries**
6. **interface virtual-template** *number*
7. **ip nat inside**
8. **exit**
9. **interface** *type number*
10. **ip nat outside**
11. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip nat settings mode cgn** **Example:** `Device(config)#  ip nat settings mode cgn`	Enables CGN operating mode.
Step 4	**ip nat inside source static** *local-ip global-ip* **Example:** `Device(config)# ip nat inside source static 192.168.2.1 192.168.34.2`	Enables static CGN of the inside source address.
Step 5	**no ip nat create flow-entries** **Example:** `Device(config)# no ip nat create flow-entries`	Disables flow cache entries in static CGN mode.
Step 6	**interface virtual-template** *number* **Example:** `Device(config)# interface virtual-template 1`	Creates a virtual template interface that can be configured and applied dynamically when creating virtual access interfaces and enters interface configuration mode.
Step 7	**ip nat inside** **Example:** `Device(config-if)# ip nat inside`	Connects the interface to the inside network, which is subject to NAT.
Step 8	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 9	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 2/1/1`	Specifies an interface and enters interface configuration mode.
Step 10	**ip nat outside** **Example:** `Device(config-if)# ip nat outside`	Connects an interface to the outside network.
Step 11	**end** **Example:**	Exits interface configuration mode and returns to privileged EXEC mode.

	Command or Action	Purpose
	`Device(config-if)# end`	

Configuration Examples for Disabling Flow Cache Entries in NAT and NAT64

Example: Disabling Flow Cache Entries in Dynamic NAT

```
Device# configure terminal
Device(config)# ip nat pool net-208 172.16.233.208 172.16.233.223 prefix-length 28
Device(config)# access-list 1 permit 192.168.34.0 0.0.0.255
Device(config)# ip nat inside source list 1 pool net-208
Device(config)# no ip nat create flow-entries
Device(config)# interface gigabitethernet 0/0/1
Device(config-if)# ip address 10.114.11.39 255.255.255.0
Device(config-if)# ip nat inside
Device(config-if)# exit
Device(config)# interface gigabitethernet 0/1/1
Device(config-if)# ip address 172.16.232.182 255.255.255.240
Device(config-if)# ip nat outside
Device(config-if)# end
```

Example: Disabling Flow Cache Entries in Static NAT64

The following example shows a static stateful Network Address Translation 64 (NAT64):

```
Device# configure terminal
Device(config)# ipv6 unicast-routing
Device(config)# interface gigabitethernet 0/0/0
Device(config-if)# description interface facing ipv6
Device(config-if)# ipv6 enable
Device(config-if)# ipv6 address 2001:DB8:1::1/96
Device(config-if)# nat64 enable
Device(config-if)# exit
Device(config)# interface gigabitethernet 1/2/0
Device(config-if)# description interface facing ipv4
Device(config-if)# ip address 209.165.201.1 255.255.255.0
Device(config-if)# nat64 enable
Device(config-if)# exit
Device(config)# nat64 prefix stateful 2001:DB8:1::1/96
Device(config)# nat64 v6v4 static 2001:DB8:1::FFFE 209.165.201.1
Device(config)# nat64 settings flow-entries disable
Device(config)# end
```

Example: Disabling Flow Cache Entries in Static CGN

The following example shows a stateful carrier-grade NAT (CGN) configuration that disables the creation of flow cache entries:

```
Device# configure terminal
Device(config)# ip nat settings mode cgn
Device(config)# ip nat inside source static 192.168.2.1 192.168.34.2
Device(config)# no ip nat create flow-entries
Device(config)# interface virtual-template 1
Device(config-if)# ip nat inside
Device(config-if)# exit
Device(config)# interface gigabitethernet 2/1/1
Device(config-if)# ip nat outside
Device(config-if)# end
```

Additional References for Disabling Flow Cache Entries in NAT and NAT64

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
NAT commands	Cisco IOS IP Addressing Services Command Reference
Carrier-grade NAT	"Carrier-Grade Network Address Translation" module in *IP Addressing NAT Configuration Guide*
Stateful NAT64	"Stateful Network Address Translation 64" module in *IP Addressing NAT Configuration Guide*
Stateless NAT64	"Stateless Network Address Translation 64" module in *IP Addressing NAT Configuration Guide*

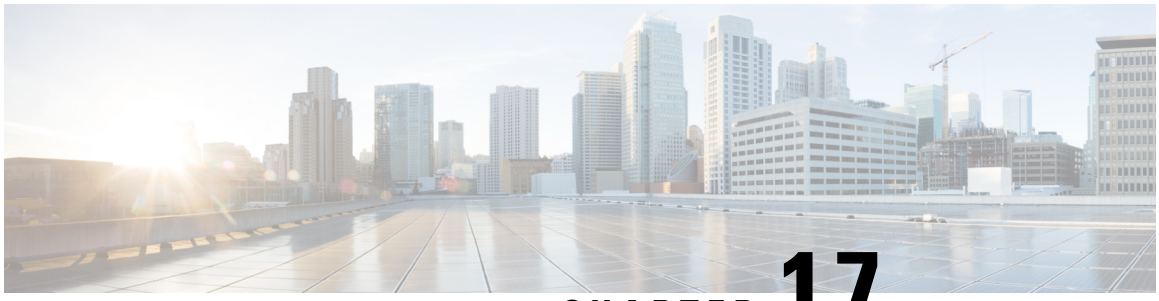
Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/support

Feature Information for Disabling Flow Cache Entries in NAT and NAT64

Table 22: Feature Information for Disabling Flow Cache Entries in NAT and NAT64

Feature Name	Releases	Feature Information
Disabling Flow Cache Entries in NAT and NAT64	Cisco IOS XE Release 3.10S	The Disabling of Flow Cache Entries in NAT and NAT64 feature allows you to disable flow entries for dynamic and static NAT translations. By default, flow entries are created for all Network Address Translation (NAT) translations. The following commands were introduced or modified: **ip nat create flow-entries**, **nat64 settings flow-entries disable**, and **show ip nat translations**.



CHAPTER 17

Paired-Address-Pooling Support in NAT

The ability of Network Address Translation (NAT) to consistently represent a local IP address as a single global IP address is termed paired address pooling. Paired address pooling is supported only on Port Address Translation (PAT).

Prior to the introduction of the Paired-Address-Pooling Support feature, if you have a PAT configuration, and you need a new global address or port, the next available address in the IP address pool is allocated. There was no mechanism to ensure that a local address is consistently mapped to a single global address. The Paired-Address-Pooling Support feature provides the ability to consistently map a local address to a global address.

Starting from IOS XE Polaris 16.8 release, you can specify an NAT pool for which PAP support is to be activated. This feature is helpful when you have to apply PAP support to a specific dynamic NAT traffic stream.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for Paired-Address-Pooling Support in NAT

Paired address pooling uses more memory, and the scaling of translations is much lower than standard Network Address Translation (NAT) configuration due to the following reasons:

- Use of a new data structure that tracks each local address.
- Use of the paired-address-pooling limit. When the number of users on a global address reaches the configured limit, the next global address is used for paired address pooling. The paired-address-pooling limit uses more memory and requires more global addresses in the address pool than standard NAT.

Information About Paired-Address-Pooling Support in NAT

Paired-Address-Pooling Support Overview

An IP address pool is a group of IP addresses. You create an IP address pool by assigning a range of IP addresses and a name to it. You allocate or assign addresses in the pool to users.

The ability of Network Address Translation (NAT) to consistently represent a local IP address as a single global IP address is termed paired address pooling. A local address is any address that appears on the inside of a network, and a global address is any address that appears on the outside of the network. You can configure paired address pooling only for Port Address Translation (PAT) because dynamic and static NAT configurations are paired configurations by default. PAT, also called overloading, is a form of dynamic NAT that maps multiple, unregistered IP addresses to a single, registered IP address (many-to-one) by using different ports. Paired address pooling is supported in both classic (default) and carrier-grade NAT (CGN) mode.

In a paired-address-pooling configuration, a local address is consistently represented as a single global address. For example, if User A is paired with the global address G1, that pairing will last as long as there are active sessions for User A. If there are no active sessions, the pairing is removed. When User A has active sessions again, the user may be paired with a different global address.

If a local address initiates new sessions, and resources (ports) are insufficient for its global address, packets are dropped. When the number of users on a global address reaches the configured limit, the next global address is used for paired address pooling. When a user who is associated with a global address through paired address pooling is unable to get a port number, then the packet is dropped, the NAT drop code is incremented, and Internet Control Message Protocol (ICMP) messages are not sent.

Paired-address-pooling uses the fill-it-up method for address selection. The fill-it-up method fits (adds) the maximum possible users into a single global address before going to the next global address.

How to Configure Paired-Address-Pooling Support

Configuring Paired-Address-Pooling Support in NAT

Note If you change the Network Address Translation (NAT) configuration mode to paired-address-pooling configuration mode and vice versa, all existing NAT sessions are removed.

To configure NAT paired-address-pooling mode, use the **ip nat settings pap** command. To remove it, use the **no ip nat settings pap** command.

After you configure paired-address-pooling mode, all pool-overload mappings will act in the paired-address-pooling manner.

Based on your NAT configuration, you can use NAT static or dynamic rules.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip nat settings pap** [**limit** {**1000** | **120** | **250** | **30** | **500** | **60**}]
4. **ip nat pool** *name start-ip end-ip* {**netmask** *netmask* | **prefix-length** *prefix-length*}
5. **access-list** *access-list-number* **permit** *source* [*source-wildcard*]
6. **ip nat inside source list** *access-list-number* **pool** *name* **overload**
7. **interface** *type number*
8. **ip address** *ip-address mask*
9. **ip nat inside**
10. **exit**
11. **interface** *type number*
12. **ip address** *ip-address mask*
13. **ip nat outside**
14. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip nat settings pap** [**limit** {**1000** | **120** | **250** | **30** | **500** | **60**}] **Example:** `Device(config)# ip nat settings pap`	Configures NAT paired address pooling configuration mode. • Use the **limit** keyword to limit of the number of local addresses you can use per global address. The default is 120.
Step 4	**ip nat pool** *name start-ip end-ip* {**netmask** *netmask* | **prefix-length** *prefix-length*} **Example:** `Device(config)# ip nat pool net-208 192.168.202.129 192.168.202.158 netmask 255.255.255.240`	Defines a pool of global addresses to be allocated as needed.

	Command or Action	Purpose
Step 5	**access-list** *access-list-number* **permit** *source* [*source-wildcard*] **Example:** `Device(config)# access-list 1 permit 192.168.34.0 0.0.0.255`	Defines a standard access list permitting addresses that are to be translated.
Step 6	**ip nat inside source list** *access-list-number* **pool** *name* **overload** **Example:** `Device(config)# ip nat inside source list 1 pool net-208 overload`	Establishes dynamic Port Address Translation (PAT) or NAT overload and specifies the access list and the IP address pool defined in Step 4 and Step 5.
Step 7	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 0/0/1`	Specifies an interface and enters interface configuration mode.
Step 8	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 10.114.11.39 255.255.255.0`	Sets a primary IP address for the interface.
Step 9	**ip nat inside** **Example:** `Device(config-if)# ip nat inside`	Connects the interface to the inside network, which is subject to NAT.
Step 10	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 11	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 0/1/2`	Specifies an interface and enters interface configuration mode.
Step 12	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 172.16.232.182 255.255.255.240`	Sets a primary IP address for the interface.
Step 13	**ip nat outside** **Example:** `Device(config-if)# ip nat outside`	Connects the interface to the outside network.
Step 14	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

How to Configure Paired-Address-Pooling Support For a NAT Pool

Configuring Paired-Address-Pooling Support For a NAT Pool

Note If you change the Network Address Translation (NAT) configuration mode to paired-address-pooling configuration mode and vice versa, all existing NAT sessions are removed.

To configure NAT paired-address-pooling mode, use the **ip nat settings pap** command. To remove it, use the **no ip nat settings pap** command.

After you configure paired-address-pooling mode, all pool-overload mappings will act in the paired-address-pooling manner.

Based on your NAT configuration, you can use NAT static or dynamic rules.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip nat settings pap** [**limit** {**1000** | **120** | **250** | **30** | **500** | **60**}]
4. **ip nat pool** *name start-ip end-ip* {**netmask** *netmask* | **prefix-length** *prefix-length*}
5. **access-list** *access-list-number* **permit** *source* [*source-wildcard*]
6. **ip nat inside source list** *access-list-number* **pool** *name* **overload**
7. **interface** *type number*
8. **ip address** *ip-address mask*
9. **ip nat inside**
10. **exit**
11. **interface** *type number*
12. **ip address** *ip-address mask*
13. **ip nat outside**
14. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**ip nat settings pap** [**limit** {**1000** | **120** | **250** | **30** | **500** | **60**}] **Example:** `Device(config)# ip nat settings pap`	Configures NAT paired address pooling configuration mode. • Use the **limit** keyword to limit of the number of local addresses you can use per global address. The default is 120.
Step 4	**ip nat pool** *name start-ip end-ip* {**netmask** *netmask* | **prefix-length** *prefix-length*} **Example:** `Device(config)# ip nat pool net-208 192.168.202.129 192.168.202.158 netmask 255.255.255.240`	Defines a pool of global addresses to be allocated as needed.
Step 5	**access-list** *access-list-number* **permit** *source* [*source-wildcard*] **Example:** `Device(config)# access-list 1 permit 192.168.34.0 0.0.0.255`	Defines a standard access list permitting addresses that are to be translated.
Step 6	**ip nat inside source list** *access-list-number* **pool** *name* **overload** **Example:** `Device(config)# ip nat inside source list 1 pool net-208 overload`	Establishes dynamic Port Address Translation (PAT) or NAT overload and specifies the access list and the IP address pool defined in Step 4 and Step 5.
Step 7	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 0/0/1`	Specifies an interface and enters interface configuration mode.
Step 8	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 10.114.11.39 255.255.255.0`	Sets a primary IP address for the interface.
Step 9	**ip nat inside** **Example:** `Device(config-if)# ip nat inside`	Connects the interface to the inside network, which is subject to NAT.
Step 10	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 11	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 0/1/2`	Specifies an interface and enters interface configuration mode.

	Command or Action	Purpose
Step 12	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 172.16.232.182 255.255.255.240`	Sets a primary IP address for the interface.
Step 13	**ip nat outside** **Example:** `Device(config-if)# ip nat outside`	Connects the interface to the outside network.
Step 14	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuration Examples for Paired-Address-Pooling Support in NAT

Example: Configuring Paired Address Pooling Support in NAT

The following example shows how to configure paired address pooling along with Network Address Translation (NAT) rules. This example shows a dynamic NAT configuration with access lists and address pools. Based on your NAT configuration, you can configure static or dynamic NAT rules.

```
Device# configure terminal
Device(config)# ip nat settings pap
Device(config)# ip nat pool net-208 192.168.202.129 192.168.202.158 netmask 255.255.255.240
Device(config)# access-list 1 permit 192.168.34.0 0.0.0.255
Device(config)# ip nat inside source list 1 pool net-208 overload
Device(config)# interface gigabitethernet 0/0/1
Device(config-if)# ip address 10.114.11.39 255.255.255.0
Device(config-if)# ip nat inside
Device(config-if)# exit
Device(config)# interface gigabitethernet 0/1/2
Device(config-if)# ip address 172.16.232.182 255.255.255.240
Device(config-if)# ip nat outside
Device(config-if)# end
```

Additional References for Paired-Address-Pooling Support in NAT

Related Documents

Related Topic	Document Title
Cisco IOS Commands	Cisco IOS Master Command List, All Releases
NAT commands	Cisco IOS IP Addressing Services Command Reference

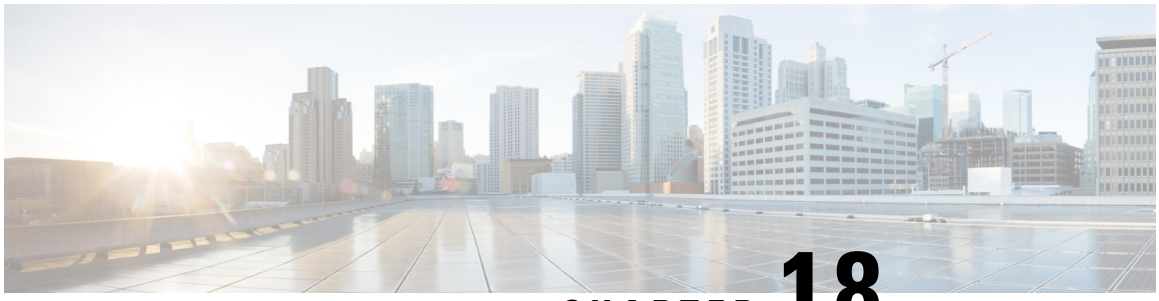
Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/support

Feature Information for Paired-Address-Pooling Support in NAT

Table 23: Feature Information for Paired-Address-Pooling Support in NAT

Feature Name	Releases	Feature Information
Paired-Address-Pooling Support in NAT	Cisco IOS XE Release 3.9S	The ability of Network Address Translation (NAT) to consistently represent a local IP address as a single global IP address is termed paired address pooling. Paired address pooling is supported only on Port Address Translation (PAT). The following command was introduced or modified: **ip nat settings pap**.



CHAPTER 18

Bulk Logging and Port Block Allocation

The Bulk Logging and Port Block Allocation feature allocates a block of ports for translation instead of allocating individual ports. This feature is supported only in carrier-grade Network Address Translation (CGN) mode.

This module provides information about the feature and how to configure it.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Bulk Logging and Port Block Allocation

- Enable the carrier-grade Network Address Translation (CGN) mode before enabling the Bulk Logging and Port Block Allocation feature.
- Enable paired-address pooling for this feature to work.

Restrictions for Bulk Logging and Port Block Allocation

- The Bulk Logging and Port Block Allocation feature is not supported on interface overload configurations because Network Address Translation (NAT) does not own the port space, the device owns it. You can configure an interface-overload mapping with this feature; however, no messages will be logged for the configuration.
- Destination information is not logged.
- Application layer gateways (ALGs) that require consecutive port pairings only work when bulk-port allocation is configured with a step size of one. For more information on step size, see "Bulk Logging and Port Block Allocation Overview, on page 260."
- Only bulk logging of messages is performed when this feature is enabled.
- ALG ports can be used for bulk-port allocation; however, this can cause degraded performance in sessions associated with these ports. If your configuration does not need ALGs, we recommend that you disable ALGs using the CLI.
- Syslog is not supported.
- Low ports, ports below 1024, are not supported; any application that requires a low port does not work with this feature.
- Bulk-port allocation pools must not overlap with static NAT mappings (particularly static mappings with ports) for this feature to work.
- The **ip nat service full-range** command is not supported.

Information About Bulk Logging and Port Block Allocation

Bulk Logging and Port Block Allocation Overview

The Bulk Logging and Port Block Allocation feature allocates ports to users in blocks, instead of allocating individual ports. When a session is started from inside the network, instead of allocating a single global IP address and a global port, multiple global ports of a single global IP address are allocated for Network Address Translation (NAT) of traffic. Based on the volume of translations, additional blocks of ports can be allocated.

To allocate port sets, you can use either the consecutive port-set method or the scattered port-set method. In the consecutive port-set method, a user is allocated a set of ports with consecutive port numbers. It is easy to determine the port numbers in the consecutive method and this as a result, can be a security threat.

The Bulk Logging and Port Block Allocation feature uses the scattered port-set method, which allows you to define a start port number, a step value, and the number of ports to allocate. For example, if the starting port number is 4000, the step value is four, and the number of ports is 512, then the step value of four is added to 4000 to get the second port number. Four is added again to 4004 to get the third port number and this process repeats until you have 512 ports in the port set. This method of port-set allocation provides better security.

Some application layer gateways (ALGs) require two consecutive global ports to operate correctly. These ALGs are supported with this feature only when a step value of one is configured, which allocates a consecutive port set.

You must enable NAT paired-address pooling support for this feature to work. This feature also supports Point-to-Point Tunneling Protocol (PPTP).

Note This feature is supported only in carrier-grade NAT (CGN) mode; therefore only source information is logged when this feature is configured. Destination information is not logged. For more information about CGN, see the "Carrier-Grade Network Address Translation" module in *IP Addressing: NAT Configuration Guide*.

Port Size in Bulk Logging and Port Block Allocation

Port size is configurable and determines the number of ports allocated in each port set. However, ports below 1024, also known as low ports, will not work when bulk logging and port-block allocation is configured.

The first port that is allocated is always the first port in the set. Initially, ports are likely to be allocated in a linear method; however, as sessions are released and ports are freed, the allocation is semi-random. A port set is freed when the last session referencing it is freed.

A few port sets are reserved for users using a specific global IP address. Therefore, when allocated ports are used up, a session can use a reserved port set. If all reserved port sets are used, the session is dropped.

The default port size is 512 ports, but it can differ based on the configured paired-address pooling limit. The following table provides information of the default port size when various paired-address pooling limits are configured:

Table 24: Default Port Size Based on Paired-Address Pooling Support

Paired-Address Pooling Limit	Default Bulk-Port Allocation Port Size	Maximum Port Step Size
120	512 ports	8
30	2048 ports	2
60	1024 ports	4
250	256 ports	4
500	128 ports	8
1000	64 ports	16

High-Speed Logging in Bulk Logging and Port Block Allocation

The Bulk Logging and Port Block Allocation feature reduces the volume of Network Address Translation (NAT) high-speed logging (HSL). The reduction is accomplished by dynamically allocating a block of global ports instead of a single global port.

Messages are usually logged when a session is created and destroyed. In bulk port allocation, messages are logged when a port set is allocated or freed.

The following table provides information about HSL fields, their format and value:

Table 25: HSL Field Description

Field	Format	ID	Value
Source IP address	IPv4 address	8	Varies
Translated source address	IPv4 address	225	Varies
VRF[1] ID	32-bit ID	234	Varies
Protocol	8-bit value	4	Varies
Event	8-bit value	230	• 0—Invalid • 1—Add event • 2—Delete event
UNIX timestamp in milliseconds	64-bit value	323	Varies
Port block start	16-bit port	361	Varies
Port block step size	16-bit step size	363	Varies
Number of ports in the block	16-bit number	364	Varies

[1] virtual routing and forwarding

How to Configure Bulk Logging and Port Block Allocation

Configuring Bulk Logging and Port-Block Allocation

Before you configure bulk logging and port-block allocation, you must:

- Enable carrier-grade Network Address Translation (CGN) mode.
- Enable NAT paired-address pooling.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ip nat inside**
5. **exit**
6. **interface** *type number*
7. **ip nat outside**
8. **exit**

9. **ip nat settings mode cgn**
10. **ip nat pool** *name start-ip end-ip* {**netmask** *netmask* | **prefix-length** *prefix-length*}
11. **access-list** *access-list-number* **permit source** [*source-wildcard*]
12. **ip nat inside source list** *access-list-number* **pool** *name*
13. **ip nat settings pap bpa set-size 512 step-size 8**
14. **ip nat log translations flow-export v9 udp destination** *addr port*
15. **end**
16. **show ip nat translations**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 0/0/0`	Specifies an interface and enters interface configuration mode.
Step 4	**ip nat inside** **Example:** `Device(config-if)# ip nat inside`	Connects the interface to the inside network, which is subject to Network Address Translation (NAT).
Step 5	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 6	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 1/0/1`	Specifies an interface and enters interface configuration mode.
Step 7	**ip nat outside** **Example:** `Device(config-if)# ip nat outside`	Connects the interface to the outside network.
Step 8	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 9	**ip nat settings mode cgn** **Example:**	Enables CGN mode.

	Command or Action	Purpose
	`Device(config)# ip nat settings mode cgn`	
Step 10	**ip nat pool** *name start-ip end-ip* {**netmask** *netmask* | **prefix-length** *prefix-length*} **Example:** `Device(config)# ip nat pool net-208 192.168.202.129 192.168.202.132 prefix-length 24`	Defines a pool of global addresses to be allocated as needed.
Step 11	**access-list** *access-list-number* **permit source** [*source-wildcard*] **Example:** `Device(config)# access-list 1 permit source 192.168.34.0 0.0.0.255`	Defines a standard access list that permits addresses that are to be translated.
Step 12	**ip nat inside source list** *access-list-number* **pool** *name* **Example:** `Device(config)# ip nat inside source list 1 pool net-208`	Establishes dynamic NAT by specifying the access list and the IP address pool defined in Step 10 and Step 11.
Step 13	**ip nat settings pap bpa set-size 512 step-size 8** **Example:** `Device(config)# ip nat settings pap bpa set-size 512 step-size 8`	Configures bulk-port allocation.
Step 14	**ip nat log translations flow-export v9 udp destination** *addr port* **Example:** `Device(config)# ip nat log translations flow-export v9 udp destination 10.1.1.1 2055`	Enables the high-speed logging (HSL) of all NAT translations.
Step 15	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.
Step 16	**show ip nat translations** **Example:** `Device# show ip nat translations`	Displays active NAT translations.

Configuration Examples for Bulk Logging and Port Block Allocation

Example: Configuring Bulk Logging and Port Block Allocation

In the following example, dynamic carrier-grade NAT (CGN) and paired-address pooling is configured for bulk-port allocation.

```
Device# configure terminal
Device(config)# interface gigabitethernet 0/0/0
Device(config-if)# ip nat inside
Device(config-if)# exit
Device(config)# interface gigabitethernet 1/0/1
Device(config-if)# ip nat outside
Device(config-if)# exit
Device(config)# ip nat settings mode cgn
Device(config)# ip nat pool net-208 192.168.202.129 192.168.202.132 prefix-length 24
Device(config)# access-list 1 permit source 192.168.34.0 0.0.0.255
Device(config)# ip nat inside source list 1 pool net-208
Device(config)# ip nat settings pap bpa set-size 512 step-size 8
Device(config)# ip nat log translations flow-export v9 udp destination 10.1.1.1 2055
Device(config)# end
```

Verifying Bulk Logging and Port Block Allocation

SUMMARY STEPS

1. **show ip nat bpa**
2. **show ip nat pool name***pool-name*

DETAILED STEPS

Step 1 **show ip nat bpa**

Example:

```
Device# show ip nat bpa
```

Displays Network Address Translation (NAT) bulk logging and port-block allocation settings.

The following is sample output from the **show ip nat bpa** command:

```
Device# show ip nat bpa

Paired Address Pooling (PAP)
Limit: 120 local addresses per global address
Bulk Port Allocation (BPA)
Port set size: 1024 ports in each port set allocation
Port step size: 1
Single set: True
```

Step 2 **show ip nat pool name***pool-name*

Example:

```
Device# show ip nat pool name pool1
```

Displays NAT pool and port statistics.

The following is sample output from the **show ip nat pool name pool1** command:

```
Device# show ip nat pool name pool1

NAT Pool Statistics
Pool name pool1, id 1
Assigned Available
Addresses 0 5
UDP Low Ports 0 0
TCP Low Ports 0 0
UDP High Ports 0 150
TCP High Ports 0 150
(Low ports are less than 1024. High ports are greater than or equal to 1024.)
```

The following is sample output from the **show ip nat pool name pool3** command:

```
Device# show ip nat pool name pool3

NAT Pool Statistics
Pool name pool3, id 4
Assigned Available
Addresses 0 9
UDP Low Ports 0 0
TCP Low Ports 0 0
UDP High Ports 0 1080
TCP High Ports 0 1080
(Low ports are less than 1024. High ports are greater than or equal to 1024.)
```

Additional References for Bulk Logging and Port Block Allocation

Related Documents

Related Topic	Document Title
Cisco IOS Commands	Master Command List
NAT commands	Cisco IOS IP Addressing Services Command Reference
Carrier-grade NAT	“Carrier-Grade Network Address Translation” module in the *IP Addressing NAT Configuration Guide*
Paired-address pooling support	“Paired-Address Pooling Support in NAT” module in the *IP Addressing NAT Configuration Guide*

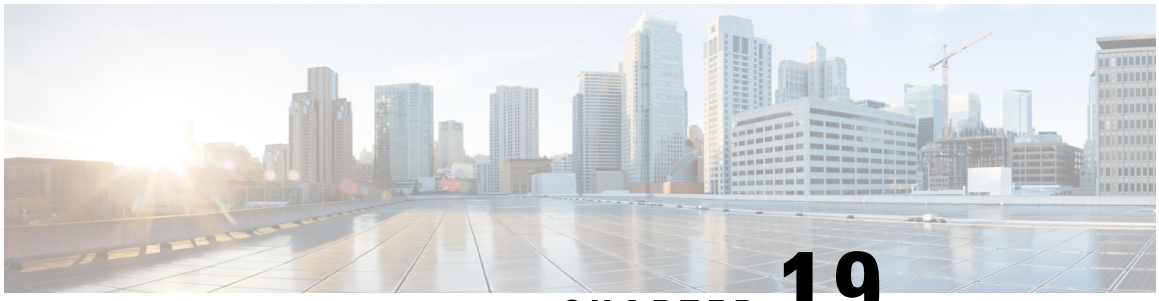
Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/support

Feature Information for Bulk Logging and Port Block Allocation

Table 26: Feature Information for Bulk Logging and Port Block Allocation

Feature Name	Releases	Feature Information
Bulk Logging and Port Block Allocation	Cisco IOS XE Release 3.10S	The Bulk Logging and Port Block Allocation feature allocates a block of ports for translation instead of allocating individual ports. The following commands were introduced or modified: **ip nat settings pap**, **ip nat settings pap bpa**, **show ip nat bpa**, and **show ip nat pool name**. In Cisco IOS XE Release 3.10S, support was added for the Cisco CSR 1000V Series Routers. In Cisco IOS XE Release 3.10S, support was added for the Cisco ISR 4400 Series Routers.



CHAPTER 19

MSRPC ALG Support for Firewall and NAT

The MSRPC ALG Support for Firewall and NAT feature provides support for the Microsoft (MS) Remote Procedure Call (RPC) application-level gateway (ALG) on the firewall and Network Address Translation (NAT). The MSRPC ALG provides deep packet inspection (DPI) of the MSRPC protocol. The MSRPC ALG works in conjunction with a provisioning system to allow the network administrator to configure match filters to define match criteria that can be searched in an MSRPC packet.

The MSRPC ALG additionally supports the Virtual Transport Control Protocol (vTCP) functionality which provides a framework for various ALG protocols to appropriately handle the TCP segmentation and parse the segments in the Cisco IOS zone-based firewall, Network Address Translation (NAT) and other applications.

Prerequisites for MSRPC ALG Support for Firewall and NAT

- You must enable the Cisco IOS XE firewall and Network Address Translation (NAT) before applying the Microsoft (MS) Remote Procedure Call (RPC) application-level gateway (ALG) on packets.

Note MSRPC ALG is automatically enabled if traffic is sent to TCP port 135 by either Cisco IOS XE firewall or NAT, or both.

Restrictions for MSRPC ALG Support for Firewall and NAT

- Only TCP-based MSRPC is supported.
- You cannot configure the **allow** and **reset** commands together.
- You must configure the **match protocol msrpc** command for DPI.

- Only traffic that reaches destination port 135 is supported. This setting can be changed by configuration.

Information About MSRPC ALG Support for Firewall and NAT

Application-Level Gateways

An application-level gateway (ALG), also known as an application-layer gateway, is an application that translates the IP address information inside the payload of an application packet. An ALG is used to interpret the application-layer protocol and perform firewall and Network Address Translation (NAT) actions. These actions can be one or more of the following depending on your configuration of the firewall and NAT:

- Allow client applications to use dynamic TCP or UDP ports to communicate with the server application.
- Recognize application-specific commands and offer granular security control over them.
- Synchronize multiple streams or sessions of data between two hosts that are exchanging data.
- Translate the network-layer address information that is available in the application payload.

The firewall opens a pinhole, and NAT performs translation service on any TCP or UDP traffic that does not carry the source and destination IP addresses in the application-layer data stream. Specific protocols or applications that embed IP address information require the support of an ALG.

MSRPC

MSRPC is a framework that developers use to publish a set of applications and services for servers and enterprises. RPC is an interprocess communication technique that allows the client and server software to communicate over the network. MSRPC is an application-layer protocol that is used by a wide array of Microsoft applications. MSRPC supports both connection-oriented (CO) and connectionless (CL) Distributed Computing Environment (DCE) RPC modes over a wide variety of transport protocols. All services of MSRPC establish an initial session that is referred to as the primary connection. A secondary session over a port range between 1024 to 65535 as the destination port is established by some services of MSRPC.

For MSRPC to work when firewall and NAT are enabled, in addition to inspecting MSRPC packets, the ALG is required to handle MSRPC specific issues like establishing dynamic firewall sessions and fixing the packet content after the NAT.

By applying MSRPC protocol inspection, most MSRPC services are supported, eliminating the need for Layer 7 policy filters.

MSRPC ALG on Firewall

After you configure the firewall to inspect the MSRPC protocol, the MSRPC ALG starts parsing MSRPC messages. The following table describes the types of Protocol Data Units (PDU) supported by the MSRPC ALG Support on Firewall and NAT feature:

Table 27: Supported PDU Types

PDU	Number	Type	Description
REQUEST	0	call	Initiates a call request.
RESPONSE	2	call	Responds to a call request.
FAULT	3	call	Indicates an RPC runtime, RPC stub, or RPC-specific exception.
BIND	11	association	Initiates the presentation negotiation for the body data.
BIND_ACK	12	association	Accepts a bind request.
BIND_NAK	13	association	Rejects an association request.
ALTER_CONTEXT	14	association	Requests additional presentation negotiation for another interface and/or version, or to negotiate a new security context, or both.
ALTER_CONTEXT_RESP	15	association	Responds to the ALTER_CONTEXT PDU. Valid values are accept or deny.
SHUTDOWN	17	call	Requests a client to terminate the connection and free the related resources.
CO_CANCEL	18	call	Cancels or orphans a connection. This message is sent when a client encounters a cancel fault.
ORPHANED	19	call	Aborts a request that in progress and that has not been entirely transmitted yet, or aborts a (possibly lengthy) response that is in progress.

MSRPC ALG on NAT

When NAT receives an MSRPC packet, it invokes the MSRPC ALG that parses the packet payload and forms a token to translate any embedded IP addresses. This token is passed to NAT, which translates addresses or ports as per your NAT configuration. The translated addresses are then written back into the packet payload by the MSRPC ALG.

If you have configured both the firewall and NAT, NAT calls the ALG first.

MSRPC Stateful Parser

The MSRPC state machine or the parser is the brain of the MSRPC ALG. The MSRPC stateful parser keeps all stateful information within the firewall or NAT depending on which feature invokes the parser first. The parser provides DPI of MSRPC protocol packets. It checks for protocol conformance and detects

out-of-sequence commands and malformed packets. As the packet is parsed, the state machine records various data and fills in the correct token information for NAT and firewall inspection.

How to Configure MSRPC ALG Support for Firewall and NAT

Note By default, MSRPC ALG is automatically enabled when NAT is enabled. There is no need to explicitly enable MSRPC ALG in the NAT-only configuration. You can use the **no ip nat service msrpc** command to disable MSRPC ALG on NAT.

Configuring a Layer 4 MSRPC Class Map and Policy Map

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **class-map type inspect match-any** *class-map-name*
4. **match protocol** *protocol-name*
5. **exit**
6. **policy-map type inspect** *policy-map-name*
7. **class type inspect** *class-map-name*
8. **inspect**
9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**class-map type inspect match-any** *class-map-name* **Example:** `Router(config)# class-map type inspect match-any msrpc-cmap`	Creates an inspect type class map for the traffic class and enters QoS class-map configuration mode.

	Command or Action	Purpose
Step 4	**match protocol** *protocol-name* **Example:** `Router(config-cmap)# match protocol msrpc`	Configures the match criteria for a class map on the basis of a specified protocol. • Only Cisco IOS XE stateful packet inspection-supported protocols can be used as match criteria in inspect type class maps.
Step 5	**exit** **Example:** `Router(config-cmap)# exit`	Exits QoS class-map configuration mode and enters global configuration mode.
Step 6	**policy-map type inspect** *policy-map-name* **Example:** `Router(config)# policy-map type inspect msrpc-pmap`	Creates a Layer 3 or Layer 4 inspect type policy map and enters QoS policy-map configuration mode.
Step 7	**class type inspect** *class-map-name* **Example:** `Router(config-pmap)# class type inspect msrpc-class-map`	Specifies the traffic (class) on which an action is to be performed and enters QoS policy-map class configuration mode.
Step 8	**inspect** **Example:** `Router(config-pmap-c)# inspect`	Enables Cisco IOS XE stateful packet inspection.
Step 9	**end** **Example:** `Router(config-pmap-c)# end`	Exits QoS policy-map class configuration mode and enters privileged EXEC mode.

Configuring a Zone Pair and Attaching an MSRPC Policy Map

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **zone security** *security-zone-name*
4. **exit**
5. **zone security** *security-zone-name*
6. **exit**
7. **zone-pair security** *zone-pair-name* [**source** *source-zone* **destination** [*destination-zone*]]
8. **service-policy type inspect** *policy-map-name*

9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Rotuer# configure terminal`	Enters global configuration mode.
Step 3	**zone security** *security-zone-name* **Example:** `Router(config)# zone security in-zone`	Creates a security zone to which interfaces can be assigned and enters security zone configuration mode.
Step 4	**exit** **Example:** `Router(config-sec-zone)# exit`	Exits security zone configuration mode and enters global configuration mode.
Step 5	**zone security** *security-zone-name* **Example:** `Router(config)# zone security out-zone`	Creates a security zone to which interfaces can be assigned and enters security zone configuration mode.
Step 6	**exit** **Example:** `Router(config-sec-zone)# exit`	Exits security zone configuration mode and enters global configuration mode.
Step 7	**zone-pair security** *zone-pair-name* [**source** *source-zone* **destination** [*destination-zone*]] **Example:** `Router(config)# zone-pair security in-out source in-zone destination out-zone`	Creates a zone pair and enters security zone pair configuration mode. **Note** To apply a policy, you must configure a zone pair.
Step 8	**service-policy type inspect** *policy-map-name* **Example:** `Router(config-sec-zone-pair)# service-policy type inspect msrpc-pmap`	Attaches a firewall policy map to the destination zone pair. **Note** If a policy is not configured between a pair of zones, traffic is dropped by default.

	Command or Action	Purpose
Step 9	**end** **Example:** `Router(config-sec-zone-pair)# end`	Exits security zone pair configuration mode and enters privileged EXEC mode.

Enabling vTCP Support for MSRPC ALG

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **alg vtcp service msrpc**
4. **exit**
5. **set platform hardware qfp active feature alg msrpc tolerance on**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**alg vtcp service msrpc** **Example:** `Rotuer(config)# alg vtcp service msrpc`	Enables vTCP functionality for MSRPC ALG. **Note** By default, MSRPC ALG supports vTCP.
Step 4	**exit** **Example:** `Rotuer(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.
Step 5	**set platform hardware qfp active feature alg msrpc tolerance on** **Example:** `Rotuer# set platform hardware qfp active feature alg msrpc tolerance on`	Enables MSRPC unknown message tolerance. **Note** By default, the tolerance is switched off.

Disabling vTCP Support for MSRPC ALG

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **no alg vtcp service msrpc**
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**no alg vtcp service msrpc** **Example:** `Rotuer(config)# no alg vtcp service msrpc`	Disables vTCP functionality for MSRPC ALG.
Step 4	**end** **Example:** `Rotuer(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Configuration Examples for MSRPC ALG Support for Firewall and NAT

Example: Configuring a Layer 4 MSRPC Class Map and Policy Map

```
Router# configure terminal
Router(config)# class-map type inspect match-any msrpc-cmap
Router(config-cmap)# match protocol msrpc
Router(config-cmap)# exit
Router(config)# policy-map type inspect msrpc-pmap
Router(config-pmap)# class type inspect msrpc-cmap
Router(config-pmap-c)# inspect
```

```
Router(config-pmap-c)# end
```

Example: Configuring a Zone Pair and Attaching an MSRPC Policy Map

```
Router# configure terminal
Router(config)# zone security in-zone
Router(config-sec-zone)# exit
Router(config)# zone security out-zone
Router(config-sec-zone)# exit
Router(config)# zone-pair security in-out source in-zone destination out-zone
Router(config-sec-zone-pair)# service-policy type inspect msrpc-pmap
Router(config-sec-zone-pair)# end
```

Example: Enabling vTCP Support for MSRPC ALG

```
Router# configure terminal
Router(config)# alg vtcp service msrpc
Router(config)# end
```

Example: Disabling vTCP Support for MSRPC ALG

```
Router# configure terminal
Router(config)# no alg vtcp service msrpc
Router(config)# end
```

Additional References for MSRPC ALG Support for Firewall and NAT

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
NAT commands	Cisco IOS IP Addressing Services Command Reference

Related Topic	Document Title
Security commands	• Cisco IOS Security Command Reference: Commands A to C • Cisco IOS Security Command Reference: Commands D to L • Cisco IOS Security Command Reference: Commands M to R • Cisco IOS Security Command Reference: Commands S to Z
NAT ALGs	"Using Application-Level Gateways with NAT" module
ALG support	*NAT and Firewall ALG Support on Cisco ASR 1000 Series Routers*

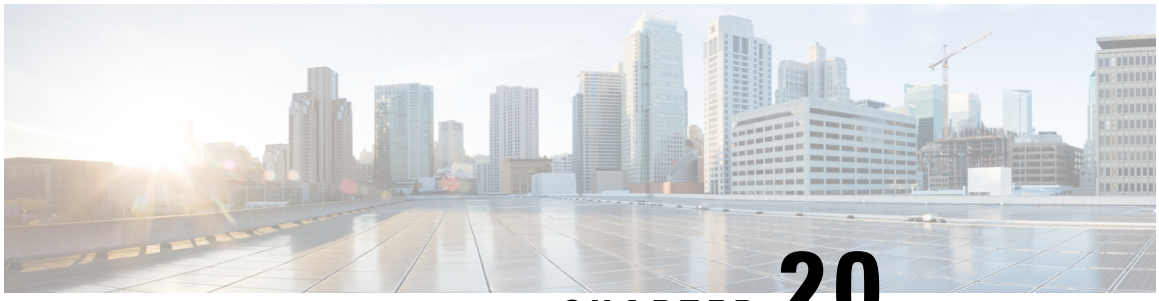
Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for MSRPC ALG Support for Firewall and NAT

Table 28: Feature Information for MSRPC ALG Support for Firewall and NAT

Feature Name	Releases	Feature Information
MSRPC ALG Support for Firewall and NAT	Cisco IOS XE Release 3.5S	The MSRPC ALG Support for Firewall and NAT feature provides support for the MSRPC ALG on the firewall and NAT. The MSRPC ALG provides deep packet inspection of the MSRPC protocol. The MSRPC ALG works in conjunction with a provisioning system to allow the network administrator to configure match filters that define match criteria that can be searched in an MSRPC packet. The following commands were introduced or modified: **ip nat service msrpc**, **match protocol msrpc**.
MSRPC ALG Inspection Improvements for Zone-based Firewall and NAT	Cisco IOS XE Release 3.14S	The MSRPC ALG Inspection Improvements for Zone-based Firewall and NAT feature supports Virtual Transport Control Protocol (vTCP) functionality which provides a framework for various ALG protocols to appropriately handle the TCP segmentation and parse the segments in the Cisco firewall, Network Address Translation (NAT) and other applications. The following command was introduced: **alg vtcp service msrpc**.



CHAPTER 20

Sun RPC ALG Support for Firewalls and NAT

The Sun RPC ALG Support for Firewalls and NAT feature adds support for the Sun Microsystems remote-procedure call (RPC) application-level gateway (ALG) on the firewall and Network Address Translation (NAT). Sun RPC is an application layer protocol that enables client programs to call functions in a remote server program. This module describes how to configure the Sun RPC ALG.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for Sun RPC ALG Support for Firewalls and NAT

- Depending on your release, the following configuration will not work on Cisco ASR 1000 Aggregation Services Routers. If you configure the inspect action for Layer 4 or Layer 7 class maps, packets that match the Port Mapper Protocol well-known port (111) pass through the firewall without the Layer 7 inspection. Without the Layer 7 inspection, firewall pinholes are not open for traffic flow, and the Sun remote-procedure call (RPC) is blocked by the firewall. As a workaround, configure the **match program-number** command for Sun RPC program numbers.
- Only Port Mapper Protocol Version 2 is supported; none of the other versions are supported.
- Only RPC Version 2 is supported.

Information About Sun RPC ALG Support for Firewalls and NAT

Application-Level Gateways

An application-level gateway (ALG), also known as an application-layer gateway, is an application that translates the IP address information inside the payload of an application packet. An ALG is used to interpret the application-layer protocol and perform firewall and Network Address Translation (NAT) actions. These actions can be one or more of the following depending on your configuration of the firewall and NAT:

- Allow client applications to use dynamic TCP or UDP ports to communicate with the server application.
- Recognize application-specific commands and offer granular security control over them.
- Synchronize multiple streams or sessions of data between two hosts that are exchanging data.
- Translate the network-layer address information that is available in the application payload.

The firewall opens a pinhole, and NAT performs translation service on any TCP or UDP traffic that does not carry the source and destination IP addresses in the application-layer data stream. Specific protocols or applications that embed IP address information require the support of an ALG.

Sun RPC

The Sun remote-procedure call (RPC) application-level gateway (ALG) performs a deep packet inspection of the Sun RPC protocol. The Sun RPC ALG works with a provisioning system that allows network administrators to configure match filters. Each match filter define a match criterion that is searched in a Sun RPC packet, thereby permitting only packets that match the criterion.

In an RPC, a client program calls procedures in a server program. The RPC library packages the procedure arguments into a network message and sends the message to the server. The server, in turn, uses the RPC library and takes the procedure arguments from the network message and calls the specified server procedure. When the server procedure returns to the RPC, return values are packaged into a network message and sent back to the client.

For a detailed description of the Sun RPC protocol, see RFC 1057, *RPC: Remote Procedure Call Protocol Specification Version 2*.

Sun RPC ALG Support for Firewalls

You can configure the Sun RPC ALG by using the zone-based firewall that is created by using policies and class maps. A Layer 7 class map allows network administrators to configure match filters. The filters specify the program numbers to be searched for in Sun RPC packets. The Sun RPC Layer 7 policy map is configured as a child policy of the Layer 4 policy map with the **service-policy** command.

When you configure a Sun RPC Layer 4 class map without configuring a Layer 7 firewall policy, the traffic returned by the Sun RPC passes through the firewall, but sessions are not inspected at Layer 7. Because sessions are not inspected, the subsequent RPC call is blocked by the firewall. Configuring a Sun RPC Layer 4 class map and a Layer 7 policy allows Layer 7 inspection. You can configure an empty Layer 7 firewall policy, that is, a policy without any match filters.

Sun RPC ALG Support for NAT

By default, the Sun RPC ALG is automatically enabled when Network Address Translation (NAT) is enabled. You can use the **no ip nat service alg** command to disable the Sun RPC ALG on NAT.

How to Configure Sun RPC ALG Support for Firewalls and NAT

For Sun RPC to work when the firewall and NAT are enabled, the ALG must inspect Sun RPC packets. The ALG also handles Sun RPC-specific issues such as establishing dynamic firewall sessions and fixing the packet content after NAT translation.

Configuring the Firewall for the Sun RPC ALG

You must configure a Layer 7 Sun remote-procedure call (RPC) policy map if you have configured the inspect action for the Sun RPC protocol (that is, if you have specified the **match protocol sunrpc** command in a Layer 4 class map).

We recommend that you do not configure both security zones and inspect rules on the same interface because this configuration may not work.

Perform the following tasks to configure a firewall for the Sun RPC ALG:

Configuring a Layer 4 Class Map for a Firewall Policy

Perform this task to configure a Layer 4 class map for classifying network traffic. When you specify the **match-all** keyword with the **class-map type inspect** command, the Sun RPC traffic matches all Sun remote-procedure call (RPC) Layer 7 filters (specified as program numbers) in the class map. When you specify the **match-any** keyword with the **class-map type inspect** , the Sun RPC traffic must match at least one of the Sun RPC Layer 7 filters (specified as program numbers) in the class map.

To configure a Layer 4 class map, use the **class-map type inspect {match-any | match-all}** *classm-map-name* command.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **class-map type inspect** {**match-any** | **match-all**} *class-map-name*
4. **match protocol** *protocol-name*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**class-map type inspect** {**match-any** | **match-all**} *class-map-name* **Example:** `Device(config)# class-map type inspect match-any sunrpc-l4-cmap`	Creates a Layer 4 inspect type class map and enters QoS class-map configuration mode.
Step 4	**match protocol** *protocol-name* **Example:** `Device(config-cmap)# match protocol sunrpc`	Configures a match criterion for a class map on the basis of the specified protocol.
Step 5	**end** **Example:** `Device(config-cmap)# end`	Exits QoS class-map configuration mode and enters privileged EXEC mode.

Configuring a Layer 7 Class Map for a Firewall Policy

Perform this task to configure a Layer 7 class map for classifying network traffic. This configuration enables programs such as mount (100005) and Network File System (NFS) (100003) that use Sun RPC. 100005 and 100003 are Sun RPC program numbers. By default, the Sun RPC ALG blocks all programs.

For more information about Sun RPC programs and program numbers, see RFC 1057, *RPC: Remote Procedure Call Protocol Specification Version 2*.

Use the **class-map type inspect** *protocol-name* command to configure a Layer 7 class map.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **class-map type inspect** *protocol-name* {**match-any** | **match-all**} *class-map-name*
4. **match program-number** *program-number*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
Step 3	**class-map type inspect** *protocol-name* {**match-any** | **match-all**} *class-map-name* **Example:** `Device(config)# class-map type inspect sunrpc match-any sunrpc-l7-cmap`	Creates a Layer 7 (application-specific) inspect type class map and enters QoS class-map configuration mode.
Step 4	**match program-number** *program-number* **Example:** `Device(config-cmap)# match program-number 100005`	Specifies the allowed RPC protocol program number as a match criterion.
Step 5	**end** **Example:** `Device(config-cmap)# end`	Exits QoS class-map configuration mode and enters privileged EXEC mode.

Configuring a Sun RPC Firewall Policy Map

Perform this task to configure a Sun remote-procedure call (RPC) firewall policy map. Use a policy map to allow packet transfer for each Sun RPC Layer 7 class that is defined in a class map for a Layer 7 firewall policy.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map type inspect** *protocol-name policy-map-name*
4. **class type inspect** *protocol-name class-map-name*
5. **allow**
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**policy-map type inspect** *protocol-name policy-map-name* **Example:**	Creates a Layer 7 (protocol-specific) inspect type policy map and enters QoS policy-map configuration mode.

	Command or Action	Purpose
	`Device(config)# policy-map type inspect sunrpc sunrpc-l7-pmap`	
Step 4	**class type inspect** *protocol-name class-map-name* **Example:** `Device(config-pmap)# class type inspect sunrpc sunrpc-l7-cmap`	Specifies the traffic class on which an action is to be performed and enters QoS policy-map class configuration mode.
Step 5	**allow** **Example:** `Device(config-pmap-c)# allow`	Allows packet transfer.
Step 6	**end** **Example:** `Device(config-pmap-c)# end`	Exits QoS policy-map class configuration mode and returns to privileged EXEC mode.

Attaching a Layer 7 Policy Map to a Layer 4 Policy Map

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map type inspect** *policy-map-name*
4. **class** {*class-map-name* | **class-default**}
5. **inspect** [*parameter-map-name*]
6. **service-policy** *protocol-name policy-map-name*
7. **exit**
8. **class class-default**
9. **drop**
10. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**policy-map type inspect** *policy-map-name* **Example:**	Creates a Layer 4 inspect type policy map and enters QoS policy-map configuration mode.

	Command or Action	Purpose
	`Device(config)# policy-map type inspect sunrpc-l4-pmap`	
Step 4	**class** {*class-map-name* | **class-default**} **Example:** `Device(config-pmap)# class sunrpc-l4-cmap`	Associates (class) on which an action is to be performed and enters QoS policy-map class configuration mode.
Step 5	**inspect** [*parameter-map-name*] **Example:** `Device(config-pmap-c)# inspect`	Enables stateful packet inspection.
Step 6	**service-policy** *protocol-name policy-map-name* **Example:** `Device(config-pmap-c)# service-policy sunrpc sunrpc-l7-pmap`	Attaches the Layer 7 policy map to a top-level Layer 4 policy map.
Step 7	**exit** **Example:** `Device(config-pmap-c)# exit`	Exits QoS policy-map class configuration mode and returns to QoS policy-map configuration mode.
Step 8	**class class-default** **Example:** `Device(config-pmap)# class class-default`	Specifies the default class (commonly known as the class-default class) before you configure its policy and enters QoS policy-map class configuration mode.
Step 9	**drop** **Example:** `Device(config-pmap-c)# drop`	Configures a traffic class to discard packets belonging to a specific class.
Step 10	**end** **Example:** `Device(config-pmap-c)# end`	Exits QoS policy-map class configuration mode and returns to privileged EXEC mode.

Creating Security Zones and Zone Pairs and Attaching a Policy Map to a Zone Pair

You need two security zones to create a zone pair. However, you can create only one security zone and the second one can be the system-defined security zone. To create the system-defined security zone or self zone, configure the **zone-pair security** command with the **self** keyword.

Note If you select a self zone, you cannot configure the inspect action.

In this task, you will do the following:

- Create security zones.
- Define zone pairs.

- Assign interfaces to security zones.
- Attach a policy map to a zone pair.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **zone security** {*zone-name* | **default**}
4. **exit**
5. **zone security** {*zone-name* | **default**}
6. **exit**
7. **zone-pair security** *zone-pair-name* **source** *source-zone-name* **destination** *destination-zone-name*
8. **service-policy type inspect** *policy-map-name*
9. **exit**
10. **interface** *type number*
11. **ip address** *ip-address mask* [**secondary** [**vrf** *vrf-name*]]
12. **zone-member security** *zone-name*
13. **exit**
14. **interface** *type number*
15. **ip address** *ip-address mask* [**secondary** [**vrf** *vrf-name*]]
16. **zone-member security** *zone-name*
17. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**zone security** {*zone-name* | **default**} **Example:** `Device(config)# zone security z-client`	Creates a security zone and enters security zone configuration mode. • Your configuration must have two security zones to create a zone pair: a source zone and a destination zone. • In a zone pair, you can use the default zone or self zone as either the source or destination zone.
Step 4	**exit** **Example:**	Exits security zone configuration mode and returns to global configuration mode.

	Command or Action	Purpose
	`Device(config-sec-zone)# exit`	
Step 5	**zone security** {*zone-name* | **default**} **Example:** `Device(config)# zone security z-server`	Creates a security zone and enters security zone configuration mode. • Your configuration must have two security zones to create a zone pair: a source zone and a destination zone. • In a zone pair, you can use the default zone as either the source or destination zone.
Step 6	**exit** **Example:** `Device(config-sec-zone)# exit`	Exits security zone configuration mode and returns to global configuration mode.
Step 7	**zone-pair security** *zone-pair-name* **source** *source-zone-name* **destination** *destination-zone-name* **Example:** `Device(config)# zone-pair security clt2srv source z-client destination z-server`	Creates a zone pair and enters security zone-pair configuration mode.
Step 8	**service-policy type inspect** *policy-map-name* **Example:** `Device(config-sec-zone-pair)# service-policy type inspect sunrpc-l4-pmap`	Attaches a firewall policy map to a zone pair.
Step 9	**exit** **Example:** `Device(config-sec-zone-pair)# exit`	Exits security zone-pair configuration mode and returns to global configuration mode.
Step 10	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 2/0/0`	Configures an interface type and enters interface configuration mode.
Step 11	**ip address** *ip-address mask* [**secondary** [**vrf** *vrf-name*]] **Example:** `Device(config-if)# ip address 192.168.6.5 255.255.255.0`	Sets a primary or secondary IP address for an interface.
Step 12	**zone-member security** *zone-name* **Example:** `Device(config-if)# zone-member security z-client`	Attaches an interface to a security zone.
Step 13	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.

	Command or Action	Purpose
Step 14	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 2/1/1`	Configures an interface type and enters interface configuration mode.
Step 15	**ip address** *ip-address mask* [**secondary** [**vrf** *vrf-name*]] **Example:** `Device(config-if)# ip address 192.168.6.1 255.255.255.0`	Sets a primary or secondary IP address for an interface.
Step 16	**zone-member security** *zone-name* **Example:** `Device(config-if)# zone-member security z-server`	Attaches an interface to a security zone.
Step 17	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuration Examples for Sun RPC ALG Support for Firewall and NAT

Example: Configuring a Layer 4 Class Map for a Firewall Policy

```
Device# configure terminal
Device(config)# class-map type inspect match-any sunrpc-l4-cmap
Device(config-cmap)# match protocol sunrpc
Device(config-cmap)# end
```

Example: Configuring a Layer 7 Class Map for a Firewall Policy

```
Device# configure terminal
Device(config)# class-map type inspect sunrpc match-any sunrpc-l7-cmap
Device(config-cmap)# match program-number 100005
Device(config-cmap)# end
```

Example: Configuring a Sun RPC Firewall Policy Map

```
Device# configure terminal
Device(config)# policy-map type inspect sunrpc sunrpc-l7-pmap
Device(config-pmap)# class type inspect sunrpc sunrpc-l7-cmap
Device(config-pmap-c)# allow
Device(config-pmap-c)# end
```

Example: Attaching a Layer 7 Policy Map to a Layer 4 Policy Map

```
Device# configure terminal
Device(config)# policy-map type inspect sunrpcl4-pmap
Device(config-pmap)# class sunrpcl4-cmap
Device(config-pmap-c)# inspect
Device(config-pmap-c)# service-policy sunrpc sunrpc-l7-pmap
Device(config-pmap-c)# exit
Device(config-pmap)# class class-default
Device(config-pmap-c)# drop
Device(config-pmap-c)# end
```

Example: Creating Security Zones and Zone Pairs and Attaching a Policy Map to a Zone Pair

```
Device# configure terminal
Device(config)# zone security z-client
Device(config-sec-zone)# exit
Device(config)# zone security z-server
Device(config-sec-zone)# exit
Device(config)# zone-pair security clt2srv source z-client destination z-server
Device(config-sec-zone-pair)# service-policy type inspect sunrpc-l4-pmap
Device(config-sec-zone-pair)# exit
Device(config)# interface gigabitethernet 2/0/0
Device(config-if)# ip address 192.168.6.5 255.255.255.0
Device(config-if)# zone-member security z-client
Device(config-if)# exit
Device(config)# interface gigabitethernet 2/1/1
Device(config-if)# ip address 192.168.6.1 255.255.255.0
Device(config-if)# zone-member security z-server
Device(config-if)# end
```

Example: Configuring the Firewall for the Sun RPC ALG

The following is a sample firewall configuration for the Sun remote-procedure call (RPC) application-level gateway (ALG) support:

```
class-map type inspect sunrpc match-any sunrpc-l7-cmap
 match program-number 100005
!
class-map type inspect match-any sunrpc-l4-cmap
 match protocol sunrpc
!
!
policy-map type inspect sunrpc sunrpc-l7-pmap
 class type inspect sunrpc sunrpc-l7-cmap
  allow
!
!
policy-map type inspect sunrpc-l4-pmap
 class type inspect sunrpc-l4-cmap
  inspect
  service-policy sunrpc sunrpc-l7-pmap
!
 class class-default
```

```
  drop
!
!
zone security z-client
!
zone security z-server
!
zone-pair security clt2srv source z-client destination z-server
 service-policy type inspect sunrpc-l4-pmap
!
interface GigabitEthernet 2/0/0
 ip address 192.168.10.1 255.255.255.0
 zone-member security z-client
!
interface GigabitEthernet 2/1/1
 ip address 192.168.23.1 255.255.255.0
 zone-member security z-server
!
```

Additional References for Sun RPC ALG Support for Firewall and NAT

Related Documents

Related Topic	Document Title
Cisco IOS commands	Master Command List, All Releases
IP Addressing commands	IP Addressing Services Command Reference
Security commands	• Security Command Reference: Commands A to C • Security Command Reference: Commands D to L • Security Command Reference: Commands M to R • Security Command Reference: Commands S to Z

Standards and RFCs

Standard/RFC	Title
RFC 1057	*RPC: Remote Procedure Call Protocol Specification Version 2*

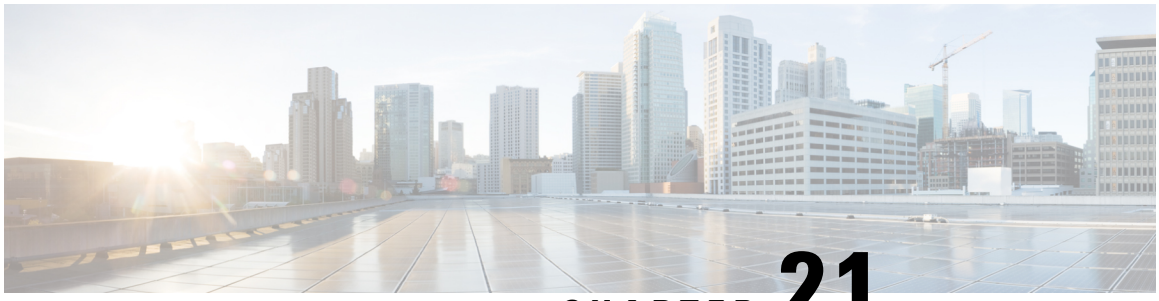
Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Sun RPC ALG Support for Firewalls and NAT

Table 29: Feature Information for Sun RPC ALG Support for Firewalls and NAT

Feature Name	Releases	Feature Information
Sun RPC ALG Support for Firewalls and NAT	Cisco IOS XE Release 3.2S	The Sun RPC ALG Support for Firewalls and NAT feature adds support for the Sun RPC ALG on the firewall and NAT. The following command was introduced or modified: **match protocol**.



CHAPTER 21

vTCP for ALG Support

Virtual Transport Control Protocol (vTCP) functionality provides a framework for various Application Layer Gateway (ALG) protocols to appropriately handle the Transport Control Protocol (TCP) segmentation and parse the segments in the Cisco firewall, Network Address Translation (NAT) and other applications.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for vTCP for ALG Support

Your system must be running Cisco IOS XE Release 3.1 or a later Cisco IOS XE software release. The latest version of NAT or firewall ALG should be configured.

Restrictions for vTCP for ALG Support

- vTCP does not support data channel traffic. To protect system resources vTCP does not support reassembled messages larger than 8K.

- vTCP does not support the high availability functionality. High availability mainly relies on the firewall or Network Address Translation (NAT) to synchronize the session information to the standby forwarding engine.
- vTCP does not support asymmetric routing. vTCP validates and assembles packet segments based on their sequence number. If packet segments that belong to the same Layer 7 message go through different devices, vTCP will not record the proper state or do an assembly of these segments.

Information About vTCP for ALG Support

Overview of vTCP for ALG Support

When a Layer 7 protocol uses TCP for transportation, the TCP payload can be segmented due to various reasons, such as application design, maximum segment size (MSS), TCP window size, and so on. The application-level gateways (ALGs) that the firewall and NAT support do not have the capability to recognize TCP fragments for packet inspection. vTCP is a general framework that ALGs use to understand TCP segments and to parse the TCP payload.

vTCP helps applications like NAT and Session Initiation Protocol (SIP) that require the entire TCP payload to rewrite the embedded data. The firewall uses vTCP to help ALGs support data splitting between packets.

When you configure firewall and NAT ALGs, the vTCP functionality is activated.

vTCP currently supports Real Time Streaming Protocol (RTSP) and DNS ALGs.

TCP Acknowledgment and Reliable Transmission

Because vTCP resides between two TCP hosts, a buffer space is required to store TCP segments temporarily, before they are sent to other hosts. vTCP ensures that data transmission occurs properly between hosts. vTCP sends a TCP acknowledgment (ACK) to the sending host if vTCP requires more data for data transmission. vTCP also keeps track of the ACKs sent by the receiving host from the beginning of the TCP flow to closely monitor the acknowledged data.

vTCP reassembles TCP segments. The IP header and the TCP header information of the incoming segments are saved in the vTCP buffer for reliable transmission.

vTCP can make minor changes in the length of outgoing segments for NAT-enabled applications. vTCP can either squeeze the additional length of data to the last segment or create a new segment to carry the extra data. The IP header or the TCP header content of the newly created segment is derived from the original incoming segment. The total length of the IP header and the TCP header sequence numbers are adjusted accordingly.

vTCP with NAT and Firewall ALGs

ALG is a subcomponent of NAT and the firewall. Both NAT and the firewall have a framework to dynamically couple their ALGs. When the firewall performs a Layer 7 inspection or NAT performs a Layer 7 fix-up, the parser function registered by the ALGs is called and ALGs take over the packet inspection. vTCP mediates between NAT and the firewall and the ALGs that use these applications. In other words, packets are first processed by vTCP and then passed on to ALGs. vTCP reassembles the TCP segments in both directions within a TCP connection.

How to Configure vTCP for ALG Support

The RTSP, DNS, NAT, and the firewall configurations enable vTCP functionality by default. Therefore no new configuration is required to enable vTCP functionality.

Enabling RTSP on Cisco ASR 1000 Series Routers to Activate vTCP

Perform this task to enable RTSP packet inspection.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **class-map type inspect match-any** *class-map-name*
4. **match protocol** *protocol-name*
5. **exit**
6. **policy-map type inspect** *policy-map-name*
7. **class type inspect** *class-map-name*
8. **inspect**
9. **class class-default**
10. **exit**
11. **exit**
12. **zone security** *zone-name1*
13. **exit**
14. **zone security** *zone-name2*
15. **exit**
16. **zone-pair security** *zone-pair-name* **source** *source-zone-name* **destination** *destination-zone-name*
17. **service-policy type inspect** *policy-map-name*
18. **exit**
19. **interface** *type number*
20. **zone-member security** *zone-name1*
21. **exit**
22. **interface** *type number*
23. **zone-member security** *zone-name*
24. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**class-map type inspect match-any** *class-map-name* **Example:** `Router(config)# class-map type inspect match-any rtsp_class1`	Creates an inspect type class map and enters class-map configuration mode.
Step 4	**match protocol** *protocol-name* **Example:** `Router(config-cmap)# match protocol rtsp`	Configures the match criteria for a class map on the basis of the named protocol. • Use DNS in place of RTSP to configure DNS as the match protocol.
Step 5	**exit** **Example:** `Router(config-cmap)# exit`	Returns to global configuration mode.
Step 6	**policy-map type inspect** *policy-map-name* **Example:** `Router(config)# policy-map type inspect rtsp_policy`	Creates an inspect type policy map and enters policy-map configuration mode.
Step 7	**class type inspect** *class-map-name* **Example:** `Router(config-pmap)# class type inspect rtsp_class1`	Specifies the class on which the action is performed and enters policy-map-class configuration mode.
Step 8	**inspect** **Example:** `Router(config-pmap-c)# inspect`	Enables stateful packet inspection.
Step 9	**class class-default** **Example:** `Router(config-pmap-c)# class class-default`	Specifies that these policy map settings apply to the predefined default class. If traffic does not match any of the match criteria in the configured class maps, it is directed to the predefined default class.
Step 10	**exit** **Example:** `Router(config-pmap-c)# exit`	Returns to policy-map configuration mode.

	Command or Action	Purpose
Step 11	**exit** **Example:** `Router(config-pmap)# exit`	Returns to global configuration mode.
Step 12	**zone security** *zone-name1* **Example:** `Router(config)# zone security private`	Creates a security zone to which interfaces can be assigned and enters security-zone configuration mode.
Step 13	**exit** **Example:** `Router(config-sec-zone)# exit`	Returns to global configuration mode.
Step 14	**zone security** *zone-name2* **Example:** `Router(config)# zone security public`	Creates a security zone to which interfaces can be assigned and enters security-zone configuration mode.
Step 15	**exit** **Example:** `Router(config-sec-zone)# exit`	Returns to global configuration mode.
Step 16	**zone-pair security** *zone-pair-name* **source** *source-zone-name* **destination** *destination-zone-name* **Example:** `Router(config)# zone-pair security pair-two source private destination public`	Creates a pair of security zones and enters security-zone-pair configuration mode. • To apply a policy, you must configure a zone pair.
Step 17	**service-policy type inspect** *policy-map-name* **Example:** `Router(config-sec-zone-pair)# service-policy rtsp_policy`	Attaches a firewall policy map to the destination zone pair. • If a policy is not configured between a pair of zones, traffic is dropped by default.
Step 18	**exit** **Example:** `Router(config-sec-zone-pair)# exit`	Returns to global configuration mode.
Step 19	**interface** *type number* **Example:** `Router(config)# GigabitEthernet0/1/0`	Specifies an interface for configuration. • Enters interface configuration mode.

	Command or Action	Purpose
Step 20	**zone-member security** *zone-name1* **Example:** `Router(config-if)# zone-member security private`	Assigns an interface to a specified security zone. • When you make an interface a member of a security zone, all traffic into and out of that interface (except traffic bound for the router or initiated by the router) is dropped by default. To let traffic through the interface, you must make the zone part of a zone pair to which you apply a policy. If the policy permits traffic, traffic can flow through that interface.
Step 21	**exit** **Example:** `Router(config-if)# exit`	Returns to global configuration mode.
Step 22	**interface** *type number* **Example:** `Router(config)# GigabitEthernet0/1/0`	Specifies an interface for configuration. • Enters interface configuration mode.
Step 23	**zone-member security** *zone-name* **Example:** `Router(config-if)# zone-member security public`	Assigns an interface to a specified security zone. • When you make an interface a member of a security zone, all traffic into and out of that interface (except traffic bound for the router or initiated by the router) is dropped by default. To let traffic through the interface, you must make the zone part of a zone pair to which you apply a policy. If the policy permits traffic, traffic can flow through that interface.
Step 24	**end** **Example:** `Router(config-if)# end`	Returns to privileged EXEC mode.

Troubleshooting Tips

The following commands can be used to troubleshoot your RTSP-enabled configuration:

- **clear zone-pair**
- **show policy-map type inspect zone-pair**
- **show zone-pair security**

Configuration Examples for vTCP for ALG Support

Example RTSP Configuration on Cisco ASR 1000 Series Routers

The following example shows how to configure the Cisco ASR 1000 Series Routers to enable RTSP inspection:

```
class-map type inspect match-any rtsp_class1
match protocol rtsp
policy-map type inspect rtsp_policy
class type inspect rtsp_class1
inspect
class class-default
zone security private
zone security public
zone-pair security pair-two source private destination public
service-policy type inspect rtsp_policy
interface GigabitEthernet0/1/0
 ip address 10.0.0.1 255.0.0.0
zone-member security private
!
interface GigabitEthernet0/1/1
 ip address 10.0.1.1 255.0.0.0
  zone-member security public
```

Additional References for vTCP for ALG Support

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
Cisco IOS firewall commands	• Security Command Reference: Commands A to C • Security Command Reference: Commands D to L • Security Command Reference: Commands M to R • Security Command Reference: Commands S to Z
Cisco Firewall--SIP Enhancements: ALG	*Security Configuration Guide: Securing the Data Plane*
Network Address Translation	*IP Addressing Services Configuration*

Standards and RFCs

Standard/RFC	Title
RFC 793	*Transport Control Protocol*
RFC 813	*Window and Acknowledge Strategy in TCP*

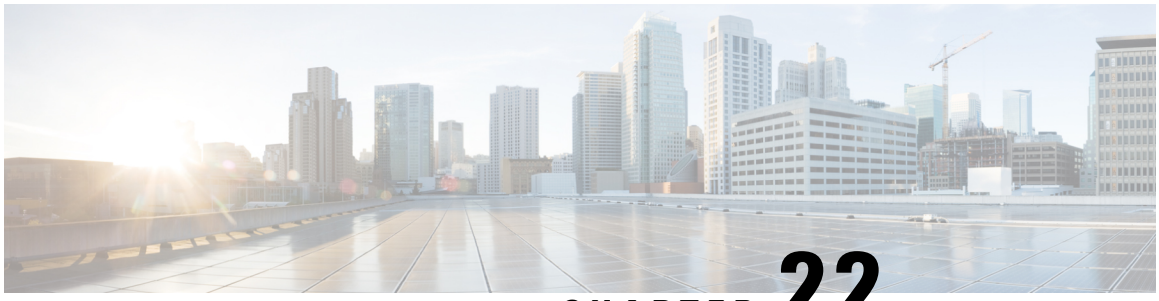
Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for vTCP for ALG Support

Table 30: Feature Information for vTCP for ALG Support

Feature Name	Releases	Feature Information
vTCP for ALG Support	Cisco IOS XE Release 3.1S	This functionality provides an enhancement to handle the TCP segmentation and reassembling for the firewall and NAT ALGs, in Cisco IOS XE software on the Cisco ASR 1000 Series Routers.



CHAPTER 22

ALG—H.323 vTCP with High Availability Support for Firewall and NAT

The ALG—H.323 vTCP with High Availability Support for Firewall and NAT feature enhances the H.323 application-level gateway (ALG) to support a TCP segment that is not a single H.323 message. Virtual TCP (vTCP) supports TCP segment reassembly. Prior to this introduction of the feature, the H.323 ALG processed a TCP segment only if it was a complete H.323 message. If the TCP segment was more than one message, the H.323 ALG ignored the TCP segment and the packet was passed without processing.

This module describes how to configure the ALG—H.323 vTCP with high availability (HA) support for firewalls.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for ALG—H.323 vTCP with High Availability Support for Firewall and NAT

- When an incoming TCP segment is not a complete H.323 message, the H.323 ALG buffers the TCP segment while waiting for the rest of the message. The buffered data is not synchronized to the standby device for high availability (HA).
- The performance of the H.323 ALG may get impacted when vTCP starts to buffer data.

Information About ALG—H.323 vTCP with High Availability Support for Firewall and NAT

Application-Level Gateways

An application-level gateway (ALG), also known as an application-layer gateway, is an application that translates the IP address information inside the payload of an application packet. An ALG is used to interpret the application-layer protocol and perform firewall and Network Address Translation (NAT) actions. These actions can be one or more of the following depending on your configuration of the firewall and NAT:

- Allow client applications to use dynamic TCP or UDP ports to communicate with the server application.
- Recognize application-specific commands and offer granular security control over them.
- Synchronize multiple streams or sessions of data between two hosts that are exchanging data.
- Translate the network-layer address information that is available in the application payload.

The firewall opens a pinhole, and NAT performs translation service on any TCP or UDP traffic that does not carry the source and destination IP addresses in the application-layer data stream. Specific protocols or applications that embed IP address information require the support of an ALG.

Basic H.323 ALG Support

H.323 is a recommendation published by the ITU-T defining a series of network elements and protocols for multimedia transmission through packet-based networks. H.323 defines a number of network elements used in multimedia transmission.

Although most H.323 implementations today utilize TCP as the transport mechanism for signaling, H.323 Version 2 enables basic UDP transport.

- H.323 Terminal—This element is an endpoint in the network, providing two-way communication with another H.323 terminal or gateway.
- H.323 Gateway—This element provides protocol conversion between H.323 terminals and other terminals that do not support H.323.
- H.323 Gatekeeper—This element provides services like address translation, network access control, and bandwidth management and account for H.323 terminals and gateways.

The following core protocols are described by the H.323 specification:

- H.225—This protocol describes call signaling methods used between any two H.323 entities to establish communication.
- H.225 Registration, Admission, and Status (RAS)—This protocol is used by the H.323 endpoint and gateway for address resolution and admission control services.
- H.245—This protocol is used for exchanging the capabilities of multimedia communication and for the opening and closing of logical channels for audio, video, and data.

In addition to the protocols listed, the H.323 specification describes the use of various IETF protocols like the Real Time Transport (RTP) protocol and audio (G.711, G.729, and so on) and video (H.261, H.263, and H.264) codecs.

NAT requires a variety of ALGs to handle Layer 7 protocol-specific services such as translating embedded IP addresses and port numbers in the packet payload and extracting new connection/session information from control channels. The H.323 ALG performs these specific services for H.323 messages.

Overview of vTCP for ALG Support

When a Layer 7 protocol uses TCP for transportation, the TCP payload can be segmented due to various reasons, such as application design, maximum segment size (MSS), TCP window size, and so on. The application-level gateways (ALGs) that the firewall and NAT support do not have the capability to recognize TCP fragments for packet inspection. vTCP is a general framework that ALGs use to understand TCP segments and to parse the TCP payload.

vTCP helps applications like NAT and Session Initiation Protocol (SIP) that require the entire TCP payload to rewrite the embedded data. The firewall uses vTCP to help ALGs support data splitting between packets.

When you configure firewall and NAT ALGs, the vTCP functionality is activated.

vTCP currently supports Real Time Streaming Protocol (RTSP) and DNS ALGs.

TCP Acknowledgment and Reliable Transmission

Because vTCP resides between two TCP hosts, a buffer space is required to store TCP segments temporarily, before they are sent to other hosts. vTCP ensures that data transmission occurs properly between hosts. vTCP sends a TCP acknowledgment (ACK) to the sending host if vTCP requires more data for data transmission. vTCP also keeps track of the ACKs sent by the receiving host from the beginning of the TCP flow to closely monitor the acknowledged data.

vTCP reassembles TCP segments. The IP header and the TCP header information of the incoming segments are saved in the vTCP buffer for reliable transmission.

vTCP can make minor changes in the length of outgoing segments for NAT-enabled applications. vTCP can either squeeze the additional length of data to the last segment or create a new segment to carry the extra data. The IP header or the TCP header content of the newly created segment is derived from the original incoming segment. The total length of the IP header and the TCP header sequence numbers are adjusted accordingly.

vTCP with NAT and Firewall ALGs

ALG is a subcomponent of NAT and the firewall. Both NAT and the firewall have a framework to dynamically couple their ALGs. When the firewall performs a Layer 7 inspection or NAT performs a Layer 7 fix-up, the parser function registered by the ALGs is called and ALGs take over the packet inspection. vTCP mediates

between NAT and the firewall and the ALGs that use these applications. In other words, packets are first processed by vTCP and then passed on to ALGs. vTCP reassembles the TCP segments in both directions within a TCP connection.

Overview of ALG—H.323 vTCP with High Availability Support

The ALG-H.323 vTCP with High Availability Support for Firewall and NAT feature enhances the H.323 application-level gateway (ALG) to support a TCP segment that is not a single H.323 message. After the H.323 ALG is coupled with vTCP, the firewall and NAT interact with the H.323 ALG through vTCP. When vTCP starts to buffer data, the high availability (HA) function is impacted, because vTCP cannot synchronize the buffered data to a standby device. If the switchover to the standby device happens when vTCP is buffering data, the connection may be reset if the buffered data is not synchronized to the standby device. After the buffered data is acknowledged by vTCP, the data is lost and the connection is reset. The firewall and NAT synchronize the data for HA. vTCP only synchronizes the status of the current connection to the standby device, and in case of errors, the connection is reset.

How to Configure ALG—H.323 vTCP with High Availability Support for Firewall and NAT

Configuring ALG-H.323 vTCP with High Availability Support for NAT

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ip nat inside**
5. **exit**
6. **interface** *type number*
7. **ip nat outside**
8. **exit**
9. **ip nat pool** *pool-name start-ip end-ip* **prefix-length** *prefix-length*
10. **ip nat inside source list pool** *pool-name*
11. **access-list** *access-list-number* **permit** *source* [*source-wildcard*]
12. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 0/0/1`	Configures an interface and enters interface configuration mode.
Step 4	**ip nat inside** **Example:** `Device(config-if)# ip nat inside`	Indicates that the interface is connected to the inside network (the network that is subject to NAT translation).
Step 5	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and enters global configuration mode.
Step 6	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 0/1/1`	Configures an interface and enters interface configuration mode.
Step 7	**ip nat outside** **Example:** `Device(config-if)# ip nat outside`	Indicates that the interface is connected to the outside network.
Step 8	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and enters global configuration mode.
Step 9	**ip nat pool** *pool-name start-ip end-ip* **prefix-length** *prefix-length* **Example:** `Device(config)# ip nat pool pool1 10.1.1.10 10.1.1.100 prefix-length 24`	Defines a pool of IP addresses for NAT.
Step 10	**ip nat inside source list pool** *pool-name* **Example:** `Device(config)# ip nat inside source list pool pool1`	Enables NAT of the inside source address.
Step 11	**access-list** *access-list-number* **permit** *source* [*source-wildcard*] **Example:** `Device(config)# access-list 1 permit 10.0.0.0 255.255.255.0`	Defines a standard IP access list and permits access to packets if conditions are matched.

	Command or Action	Purpose
Step 12	**end** **Example:** `Device(config)# end`	Exits global configuration mode and enters privileged EXEC mode.

Example

The following is sample output from the **show ip nat statistics** command:

```
Device# show ip nat statistics

Total active translations: 2 (0 static, 2 dynamic; 1 extended)
Outside interfaces:
  GigabitEthernet0/0/1
Inside interfaces:
  GigabitEthernet0/1/1
Hits: 0  Misses: 25
CEF Translated packets: 0, CEF Punted packets: 0
Expired translations: 0
Dynamic mappings:
-- Inside Source
[Id: 1] access-list 1 pool pool1 refcount 2
 pool pool1: netmask 255.255.255.0
        start 10.1.1.10 end 10.1.1.100
        type generic, total addresses 91, allocated 1 (1%), misses 0
nat-limit statistics:
 max entry: max allowed 0, used 0, missed 0
Pool stats drop: 0  Mapping stats drop: 0
Port block alloc fail: 0
IP alias add fail: 0
Limit entry add fail: 0
```

The following is sample output from the **show ip nat translations** command:

```
Device# show ip nat translations

Pro  Inside global         Inside local          Outside local         Outside global
---  10.1.1.10             10.2.1.2              ---                   ---
udp  10.1.1.10:75          10.2.1.2:75           10.1.1.1:69           10.1.1.1:69
Total number of translations: 2
```

Configuration Examples for ALG—H.323 vTCP with High Availability Support for Firewall and NAT

Example: Configuring ALG-H.323 vTCP with High Availability Support for NAT

```
Device# configure terminal
Device(config)# interface gigabitethernet 0/0/1
Device(config-if)# ip nat inside
Device(config-if)# exit
Device(config)# interface gigabitethernet 0/1/1
Device(config-if)# ip nat outside
```

```
Device(config-if)# exit
Device(config)# ip nat pool pool1 10.1.1.10 10.1.1.100 prefix-length 24
Device(config)# ip nat inside source list pool pool1
Device(config)# access-list 1 permit 10.0.0.0 255.255.255.0
Device(config)# end
```

Additional References for ALG-H.323 vTCP with High Availability Support for Firewall and NAT

Related Documents

Related Topic	Document Title
Cisco IOS commands	Master Commands List, All Releases
Firewall commands	• Security Command Reference: Commands A to C • Security Command Reference: Commands D to L • Security Command Reference: Commands M to R • Security Command Reference: Commands S to Z
NAT commands	IP Addressing Services Command Reference

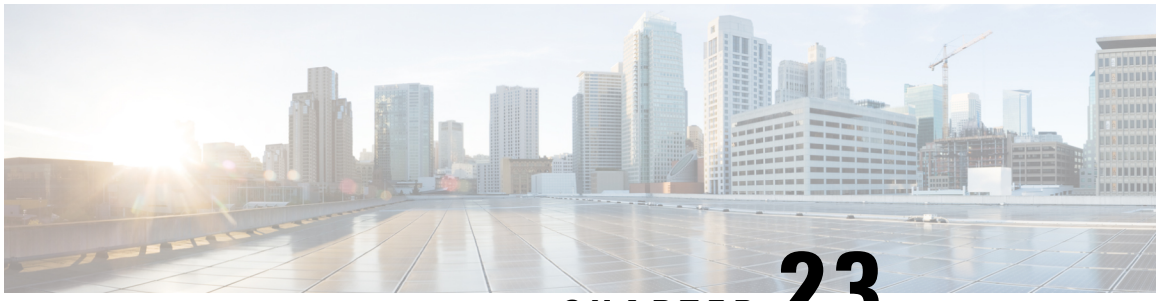
Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for ALG—H.323 vTCP with High Availability Support for Firewall and NAT

Table 31: Feature Information for ALG—H.323 vTCP with High Availability Support for Firewall and NAT

Feature Name	Releases	Feature Information
ALG—H.323 vTCP with High Availability Support for Firewall and NAT	Cisco IOS XE Release 3.7S	The ALG—H.323 vTCP with High Availability Support for Firewall and NAT feature enhances the H.323 ALG to support a TCP segment that is not a single H.323 message. vTCP supports segment reassembly. Prior to the introduction of this feature, the H.323 ALG processed a TCP segment only if it was a complete H.323 message. If the TCP segment was more than one message, the H.323 ALG ignored the TCP segment and the packet was passed without processing.



CHAPTER 23

SIP ALG Hardening for NAT and Firewall

The SIP ALG Hardening for NAT and Firewall feature provides better memory management and RFC compliance over the existing Session Initiation Protocol (SIP) application-level gateway (ALG) support for Network Address Translation (NAT) and firewall. This feature provides the following enhancements:

- Management of the local database for all SIP Layer 7 data
- Processing of the Via header
- Support for logging additional SIP methods
- Support for Provisional Response Acknowledgment (PRACK) call flow
- Support for the Record-Route header

The above enhancements are available by default; no additional configuration is required on NAT or firewall.

This module explains the SIP ALG enhancements and describes how to enable NAT and firewall support for SIP.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for SIP ALG Hardening for NAT and Firewall

- Session Initiation Protocol (SIP) application-level gateway (ALG) does not provide any security features.
- SIP ALG manages the local database based on call IDs. There might be a corner case involving two calls coming from two different clients with the same call ID, resulting in call ID duplication.

Information About SIP ALG Hardening for NAT and Firewall

SIP Overview

Session Initiation Protocol (SIP) is an application-layer control (signaling) protocol for creating, modifying, and terminating sessions with one or more participants. These sessions could include Internet telephone calls, multimedia distribution, and multimedia conferences. SIP is based on an HTTP-like request/response transaction model. Each transaction consists of a request that invokes a particular method or function on the server and at least one response.

SIP invitations that are used to create sessions carry session descriptions that allow participants to agree on a set of compatible media types. SIP makes use of elements called proxy servers to help route requests to users' current locations, authenticate and authorize users for services, implement provider call-routing policies, and provide features to users. SIP also provides a registration function that allows users to upload their current locations for use by proxy servers. SIP runs on top of several different transport protocols.

Application-Level Gateways

An application-level gateway (ALG), also known as an application-layer gateway, is an application that translates the IP address information inside the payload of an application packet. An ALG is used to interpret the application-layer protocol and perform firewall and Network Address Translation (NAT) actions. These actions can be one or more of the following depending on your configuration of the firewall and NAT:

- Allow client applications to use dynamic TCP or UDP ports to communicate with the server application.
- Recognize application-specific commands and offer granular security control over them.
- Synchronize multiple streams or sessions of data between two hosts that are exchanging data.
- Translate the network-layer address information that is available in the application payload.

The firewall opens a pinhole, and NAT performs translation service on any TCP or UDP traffic that does not carry the source and destination IP addresses in the application-layer data stream. Specific protocols or applications that embed IP address information require the support of an ALG.

SIP ALG Local Database Management

A Session Initiation Protocol (SIP) trunk is a direct connection of an IP PBX to a service provider over an IP network using SIP. There can be numerous concurrent calls in a SIP trunk. During the call setup process, all calls use the same control channel for call establishment. More than one call uses the same control channel for call setup. When the same control channel is used by more than one call, the stateful information stored

in the control-channel sessions becomes unreliable. SIP stateful information consists of media channel information such as the IP address and port number used by client and server endpoints to send media data. The media channel information is used to create a firewall pinhole and a Network Address Translation (NAT) door for the data channel in firewall and NAT, respectively. Because multiple calls use the same control channel for call setup, there will be multiple sets of media data.

In a SIP trunk, more than one call shares the same firewall and NAT session. NAT and firewall identify and manage a SIP session by using the 5 tuple in a SIP packet—source address, destination address, source port, destination port, and protocol. The conventional method of using the 5 tuple to identify and match calls does not completely support SIP trunking and often leads to Layer 7 data memory leaks and call matching issues.

In contrast to other application-level gateways (ALGs), SIP ALG manages the SIP Layer 7 data by using a local database to store all media-related information contained in normal SIP calls and in SIP calls embedded in a SIP trunk. SIP ALG uses the Call-ID header field contained in a SIP message to search the local database for call matching and to manage and terminate calls. The Call-ID header field is a dialog identifier that identifies messages belonging to the same SIP dialog.

SIP ALG uses the call ID to perform search in the local database and to manage memory resources. In certain scenarios where SIP ALG is unable to free up a Layer 7 data record from the database, a session timer is used to manage and free resources to ensure that there are no stalled call records in the database.

Note Because all Layer 7 data is managed by SIP ALG by using a local database, SIP ALG never replies on firewall and NAT to free SIP Layer 7 data; SIP ALG frees the data by itself. If you use the **clear** command to clear all NAT translations and firewall sessions, the SIP Layer 7 data in the local database is not freed.

SIP ALG Via Header Support

A Session Initiation Protocol (SIP) INVITE request contains a Via header field. The Via header field indicates the transport paths taken by a SIP request. The Via header also contains information about the return path for subsequent SIP responses, which includes the IP address and the port to which the response message is to be sent.

SIP ALG creates a firewall pinhole or a Network Address Translation (NAT) door based on the first value in the Via header field for each SIP request received, except the acknowledge (ACK) message. If the port number information is missing from the first Via header, the port number is assumed to be 5060.

SIP ALG Method Logging Support

The SIP ALG Hardening for NAT and Firewall feature provides support for detailed logging of the following methods in Session Initiation Protocol (SIP) application-level gateway (ALG) statistics:

- PUBLISH
- OPTIONS
- 1XX (excluding 100,180,183)
- 2XX (excluding 200)

The existing SIP methods that are logged in SIP ALG statistics include ACK, BYE, CANCEL, INFO, INVITE, MESSAGE, NOTIFY, REFER, REGISTER, SUBSCRIBE, and 1XX-6XX.

SIP ALG PRACK Call-Flow Support

Session Initiation Protocol (SIP) defines two types of responses: final and provisional. Final responses convey the result of processing a request and are sent reliably. Provisional responses, on the other hand, provide information about the progress of processing a request but are not sent reliably.

Provisional Response Acknowledgement (PRACK) is a SIP method that provides an acknowledgment (ACK) system for provisional responses. PRACK allows reliable exchanges of SIP provisional responses between SIP endpoints. SIP reliable provisional responses ensure that media information is exchanged and resource reservation can occur before connecting the call.

SIP uses the connection, media, and attribute fields of the Session Description Protocol (SDP) during connection negotiation. SIP application-level gateway (ALG) supports SDP information within a PRACK message. If media information exists in a PRACK message, SIP ALG retrieves and processes the media information. SIP ALG also handles the creation of media channels for subsequent media streams. SIP ALG creates a firewall pinhole and a NAT door based on the SDP information in PRACK messages.

SIP ALG Record-Route Header Support

The Record-Route header field is added by a Session Initiation Protocol (SIP) proxy to a SIP request to force future requests in a SIP dialog to be routed through the proxy. Messages sent within a dialog then traverse all SIP proxies, which add a Record-Route header field to the SIP request. The Record-Route header field contains a globally reachable Uniform Resource Identifier (URI) that identifies the proxy.

SIP application-level gateway (ALG) parses the Contact header and uses the IP address and the port value in the Contact header to create a firewall pinhole and a Network Address Translation (NAT) door. In addition, SIP ALG supports the parsing of the Record-Route header to create a firewall pinhole and a NAT door for future messages that are routed through proxies.

With the parsing of the Record-Route header, SIP ALG supports the following scenarios:

- A Cisco ASR 1000 Aggregation Services Router is deployed between two proxies.
- A Cisco ASR 1000 Aggregation Services Router is deployed between a User Agent Client (UAC) and a proxy.
- A Cisco ASR 1000 Aggregation Services Router is deployed between a proxy and a User Agent Server (UAS).
- No proxy exists between the client and the server. No record routing occurs in this scenario.

How to Configure SIP ALG Hardening for NAT and Firewall

Enabling NAT for SIP Support

NAT support for SIP is enabled by default on port 5060. If this feature has been disabled, perform this task to re-enable NAT support for SIP. To disable the NAT support for SIP, use the **no ip nat service sip** command.

SUMMARY STEPS

1. **enable**

2. **configure terminal**
3. **ip nat service sip** {**tcp** | **udp**} **port** *port-number*
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip nat service sip** {**tcp**	**udp**} **port** *port-number* **Example:** `Device(config)# ip nat service sip tcp port 5060`	Enables NAT support for SIP.
Step 4	**end** **Example:** `Device(config)# end`	Exist global configuration mode and returns to privileged EXEC mode.

Enabling SIP Inspection

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **class-map type inspect match-any** *class-map-name*
4. **match protocol** *protocol-name*
5. **exit**
6. **policy-map type inspect** *policy-map-name*
7. **class type inspect** *class-map-name*
8. **inspect**
9. **exit**
10. **class class-default**
11. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
	`Device> enable`	
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**class-map type inspect match-any** *class-map-name* **Example:** `Device(config)# class-map type inspect match-any sip-class1`	Creates an inspect type class map and enters class-map configuration mode.
Step 4	**match protocol** *protocol-name* **Example:** `Device(config-cmap)# match protocol sip`	Configures the match criterion for a class map based on the named protocol.
Step 5	**exit** **Example:** `Device(config-cmap)# exit`	Exits class-map configuration mode.
Step 6	**policy-map type inspect** *policy-map-name* **Example:** `Device(config)# policy-map type inspect sip-policy`	Creates an inspect type policy map and enters policy-map configuration mode.
Step 7	**class type inspect** *class-map-name* **Example:** `Device(config-pmap)# class type inspect sip-class1`	Specifies the class on which the action is performed and enters policy-map class configuration mode.
Step 8	**inspect** **Example:** `Device(config-pmap-c)# inspect`	Enables stateful packet inspection.
Step 9	**exit** **Example:** `Device(config-pmap-c)# exit`	Exits policy-map class configuration mode and returns to policy-map configuration mode.
Step 10	**class class-default** **Example:** `Device(config-pmap)# class class-default`	Specifies that these policy map settings apply to the predefined default class. • If traffic does not match any of the match criteria in the configured class maps, it is directed to the predefined default class.
Step 11	**end** **Example:** `Device(config-pmap)# end`	Exits policy-map configuration mode and returns to privileged EXEC mode.

Configuring a Zone Pair and Attaching a SIP Policy Map

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **zone security** {*zone-name* | **default**}
4. **exit**
5. **zone security** {*zone-name* | **default**}
6. **exit**
7. **zone-pair security** *zone-pair-name* [**source** {*source-zone-name* | **self** | **default**} **destination** [*destination-zone-name* | **self** | **default**]]
8. **service-policy type inspect** *policy-map-name*
9. **exit**
10. **interface** *type number*
11. **zone-member security** *zone-name*
12. **exit**
13. **interface** *type number*
14. **zone-member security** *zone-name*
15. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**zone security** {*zone-name* | **default**} **Example:** `Device(config)# zone security zone1`	Creates a security zone to which interfaces can be assigned and enters security zone configuration mode.
Step 4	**exit** **Example:** `Device(config-sec-zone)# exit`	Exits security zone configuration mode and returns to global configuration mode.
Step 5	**zone security** {*zone-name* | **default**} **Example:** `Device(config)# zone security zone2`	Creates a security zone to which interfaces can be assigned and enters security zone configuration mode.

	Command or Action	Purpose
Step 6	**exit** **Example:** `Device(config-sec-zone)# exit`	Exits security zone configuration mode and returns to global configuration mode.
Step 7	**zone-pair security** *zone-pair-name* [**source** {*source-zone-name* | **self** | **default**} **destination** [*destination-zone-name* | **self** | **default**]] **Example:** `Device(config)# zone-pair security in-out source zone1 destination zone2`	Creates a zone pair and returns to security zone-pair configuration mode. **Note** To apply a policy, you must configure a zone pair.
Step 8	**service-policy type inspect** *policy-map-name* **Example:** `Device(config-sec-zone-pair)# service-policy type inspect sip-policy`	Attaches a firewall policy map to the destination zone pair. **Note** If a policy is not configured between a pair of zones, traffic is dropped by default.
Step 9	**exit** **Example:** `Device(config-sec-zone-pair)# exit`	Exits security zone-pair configuration mode and returns to global configuration mode.
Step 10	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 0/0/0`	Configures an interface and enters interface configuration mode.
Step 11	**zone-member security** *zone-name* **Example:** `Device(config-if)# zone-member security zone1`	Assigns an interface to a specified security zone. **Note** When you make an interface a member of a security zone, all traffic in and out of that interface (except traffic bound for the device or initiated by the device) is dropped by default. To let traffic through the interface, you must make the zone part of a zone pair to which you apply a policy. If the policy permits traffic, traffic can flow through that interface.
Step 12	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 13	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 0/1/1`	Configures an interface and enters interface configuration mode.
Step 14	**zone-member security** *zone-name* **Example:** `Device(config-if)# zone-member security zone2`	Assigns an interface to a specified security zone.

	Command or Action	Purpose
Step 15	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuration Examples for SIP ALG Hardening for NAT and Firewall

Example: Enabling NAT for SIP Support

```
Device> enable
Device# configure terminal
Device(config)# ip nat service sip tcp port 5060
Device(config)# end
```

Example: Enabling SIP Inspection

```
class-map type inspect match-any sip-class1
 match protocol sip
!
policy-map type inspect sip-policy
 class type inspect sip-class1
  inspect
!
class class-default
```

Example: Configuring a Zone Pair and Attaching a SIP Policy Map

```
zone security zone1
!
zone security zone2
!
zone-pair security in-out source zone1 destination zone2
 service-policy type inspect sip-policy
!
interface gigabitethernet 0/0/0
 zone security zone1
!
interface gigabitethernet 0/1/1
 zone security zone2
```

Additional References for SIP ALG Hardening for NAT and Firewall

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
NAT configuration	*IP Addressing: NAT Configuration Guide*
Firewall configuration	*Security Configuration Guide: Zone-Based Policy Firewall*
NAT commands	Cisco IOS IP Addressing Services Command Reference
Firewall commands	• Cisco IOS Security Command Reference: Commands A to C • Cisco IOS Security Command Reference: Commands D to L • Cisco IOS Security Command Reference: Commands M to R • Cisco IOS Security Command Reference: Commands S to Z
NAT and firewall ALG support	NAT and Firewall ALG and AIC Support on Cisco ASR 1000 Series Aggregation Services Routers matrix

Standards and RFCs

Standard/RFC	Title
RFC 3261	*SIP: Session Initiation Protocol*

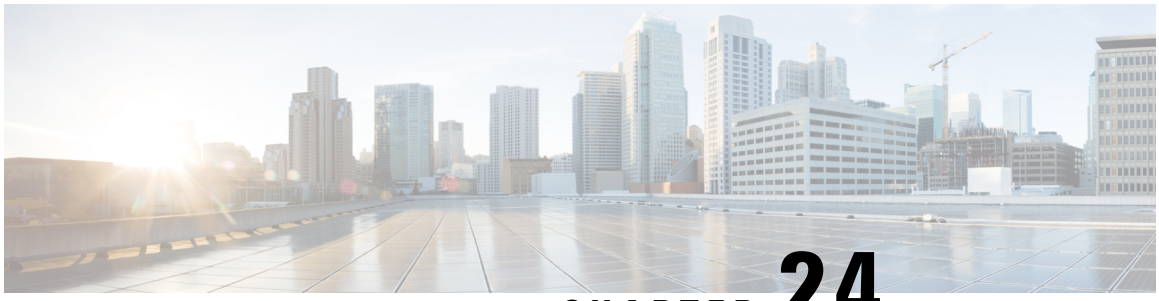
Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for SIP ALG Hardening for NAT and Firewall

Table 32: Feature Information for SIP ALG Hardening for NAT and Firewall

Feature Name	Releases	Feature Information
SIP ALG Hardening for NAT and Firewall	Cisco IOS XE Release 3.8S	The SIP ALG Hardening for NAT and Firewall feature provides better memory management and RFC compliance over the existing SIP ALG support for NAT and firewall.



CHAPTER 24

SIP ALG Resilience to DoS Attacks

The SIP ALG Resilience to DoS Attacks feature provides protection against Session Initiation Protocol (SIP) application layer gateway (ALG) denial of service (DoS) attacks. This feature supports a configurable lock limit, a dynamic blacklist, and configurable timers to prevent DoS attacks.

This module explains the feature and how to configure DoS prevention for the SIP application layer gateway (ALG). Network Address Translation and zone-based policy firewalls support this feature.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About SIP ALG Resilience to DoS Attacks

SIP ALG Resilience to DoS Attacks Overview

The SIP ALG Resilience to DoS Attacks feature provides protection against denial of service (DoS) attacks to the Session Initiation Protocol (SIP) application layer gateway (ALG). This feature supports a configurable lock limit, a dynamic blacklist, and configurable timers to prevent DoS attacks. This feature is supported by Network Address Translation (NAT) and zone-based policy firewalls.

SIP is an application-level signaling protocol for setting up, modifying, and terminating real-time sessions between participants over an IP data network. These sessions could include Internet telephone calls, multimedia distribution, and multimedia conferences. SIP DoS attacks are a major threat to networks.

The following are types of SIP DoS attacks:

- SIP register flooding: A registration flood occurs when many VoIP devices try to simultaneously register to a network. If the volume of registration messages exceeds the device capability, some messages are lost. These devices then attempt to register again, adding more congestion. Because of the network congestion, users may be unable to access the network for some time.
- SIP INVITE flooding: An INVITE flood occurs when many INVITE messages are sent to servers that cannot support all these messages. If the attack rate is very high, the memory of the server is exhausted.
- SIP broken authentication and session attack: This attack occurs when an attacker presumes the identity of a valid user, using digest authentication. When the authentication server tries to verify the identity of the attacker, the verification is ignored and the attacker starts a new request with another session identity. These attacks consume the memory of the server.

SIP ALG Dynamic Blacklist

One of the common methods of denial of service (DoS) attacks involves saturating the target network with external communication requests making the network unable to respond to legitimate traffic. To solve this issue, the SIP ALG Resilience to DoS Attacks feature uses configurable blacklists. A blacklist is a list of entities that are denied a particular privilege, service, or access. Dynamic blacklists are disabled by default. When requests to a destination address exceed a predefined trigger criteria in the configured blacklist, the Session Initiation Protocol (SIP) application layer gateway (ALG) will drop these packets.

The following abnormal SIP session patterns are monitored by dynamic blacklists:

- In the configured period of time if a source sends multiple requests to a destination and receives non-2xx (as per RFC 3261, any response with a status code between 200 and 299 is a "2xx response") final responses from the destination.
- In the configured period of time if a source sends multiple requests to a destination and does not receive any response from the destination.

SIP ALG Lock Limit

Both Network Address Translation (NAT) and the firewall use the Session Initiation Protocol (SIP) application layer gateway (ALG) to parse SIP messages and create sessions through tokens. To maintain session states, the SIP ALG uses a per call data structure and Layer 7 data to store call-related information that is allocated when a session is initiated and freed when a session is released. If the SIP ALG does not receive a message that indicates that the call has ended, network resources are held for the call.

Because Layer 7 data is shared between threads, a lock is required to access the data. During denial of service (DoS) and distributed DoS attacks, many threads wait to get the same lock, resulting in heavy CPU usage, which makes the system unstable. To prevent the system from becoming unstable, a limit is added to restrict the number of threads that can wait for a lock. SIP sessions are established by request/response mode. When there are too many concurrent SIP messages for one SIP call, packets that exceed the lock limit are dropped.

SIP ALG Timers

To exhaust resources on Session Initiation Protocol (SIP) servers, some denial of service (DoS) attacks do not indicate the end of SIP calls. To prevent these types of DoS attacks, a protection timer is added.

The SIP ALG Resilience to DoS Attacks feature uses the following timers:

- Call-duration timer that controls the maximum length of an answered SIP call.
- Call-proceeding timer that controls the maximum length of an unanswered SIP call.

When the configured maximum time is reached, the SIP application layer gateway (ALG) releases resources for this call, and future messages related to this call may not be properly parsed by the SIP ALG.

How to Configure SIP ALG Resilience to DoS Attacks

Configuring SIP ALG Resilience to DoS Attacks

You can configure the prevention of denial of service (DoS) parameters for the Session Initiation Protocol (SIP) application layer gateway (ALG) that is used by Network Address Translation (NAT) and the zone-based policy firewall.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **alg sip processor session max-backlog** *concurrent-processor-usage*
4. **alg sip processor global max-backlog** *concurrent-processor-usage*
5. **alg sip blacklist trigger-period** *trigger-period* **trigger-size** *minimum-events* **destination** *ip-address*
6. **alg sip blacklist trigger-period** *trigger-period* **trigger-size** *minimum-events* **block-time** *block-time* [**destination** *ip-address*]
7. **alg sip timer call-proceeding-timeout** *time*
8. **alg sip timer max-call-duration** *seconds*
9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**alg sip processor session max-backlog** *concurrent-processor-usage* **Example:** `Device(config)# alg sip processor session max-backlog 5`	Sets a per session limit for the number of backlog messages waiting for shared resources.

	Command or Action	Purpose
Step 4	**alg sip processor global max-backlog** *concurrent-processor-usage* **Example:** `Device(config)# alg sip processor global max-backlog 5`	Sets the maximum number of backlog messages waiting for shared resources for all SIP sessions.
Step 5	**alg sip blacklist trigger-period** *trigger-period* **trigger-size** *minimum-events* **destination** *ip-address* **Example:** `Device(config)# alg sip blacklist trigger-period 90 trigger-size 30 destination 10.1.1.1`	Configures dynamic SIP ALG blacklist criteria for the specified destination IP address.
Step 6	**alg sip blacklist trigger-period** *trigger-period* **trigger-size** *minimum-events* **block-time** *block-time* [**destination** *ip-address*] **Example:** `Device(config)# alg sip blacklist trigger-period 90 trigger-size 30 block-time 30`	Configures the time period, in seconds, when packets from a source are blocked if the configured limit is exceeded.
Step 7	**alg sip timer call-proceeding-timeout** *time* **Example:** `Device(config)# alg sip timer call-proceeding-timeout 35`	Sets the maximum time interval, in seconds, to end SIP calls that do not receive a response.
Step 8	**alg sip timer max-call-duration** *seconds* **Example:** `Device(config)# alg sip timer max-call-duration 90`	Sets the maximum call duration, in seconds, for a successful SIP call.
Step 9	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Verifying SIP ALG Resilience to DoS Attacks

Use the following commands to troubleshoot the feature.

SUMMARY STEPS

1. **enable**
2. **show alg sip**
3. **show platform hardware qfp** {**active** | **standby**} **feature alg statistics sip**
4. **show platform hardware qfp** {**active** | **standby**} **feature alg statistics sip dbl**
5. **show platform hardware qfp** {**active** | **standby**} **feature alg statistics sip dblcfg**
6. **show platform hardware qfp** {**active** | **standby**} **feature alg statistics sip processor**
7. **show platform hardware qfp** {**active** | **standby**} **feature alg statistics sip timer**

8. **debug alg** {**all** | **info** | **trace** | **warn**}

DETAILED STEPS

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **show alg sip**

Displays all Session Initiation Protocol (SIP) application layer gateway (ALG) information.

Example:

```
Device# show alg sip

sip timer configuration
   Type                      Seconds
   max-call-duration         380
   call-proceeding-timeout   620

sip processor configuration
   Type           Backlog number
   session         14
   global          189

sip blacklist configuration
   dst-addr         trig-period(ms)    trig-size     block-time(sec)
   10.0.0.0             60                 30              2000
   10.1.1.1             20                 30              30
   192.0.2.115          1000               5               30
   198.51.100.34        20                 30              388
```

Step 3 **show platform hardware qfp** {**active** | **standby**} **feature alg statistics sip**

Displays SIP ALG-specific statistics information in the Cisco Quantum Flow Processor (QFP).

Example:

```
Device# show platform hardware qfp active feature alg statistics sip

Events
...
Cr dbl entry:                10   Del dbl entry:                 10
Cr dbl cfg entry:             8   Del dbl cfg entry:              4
start dbl trig tmr:          10   restart dbl trig tmr:        1014
stop dbl trig tmr:           10   dbl trig timeout:            1014
start dbl blk tmr:            0   restart dbl blk tmr:            0
stop dbl blk tmr:             0   dbl blk tmr timeout:            0
start dbl idle tmr:          10   restart dbl idle tmr:         361
stop dbl idle tmr:            1   dbl idle tmr timeout:           9

DoS Errors
Dbl Retmem Failed:            0    Dbl Malloc Failed:             0
DblCfg Retm Failed:           0    DblCfg Malloc Failed:          0
Session wlock ovflw:          0    Global wlock ovflw:            0
Blacklisted:                561
```

Step 4 **show platform hardware qfp {active | standby} feature alg statistics sip dbl**

Displays brief information about all SIP blacklist data.

Example:

```
Device# show platform hardware qfp active feature alg statistics sip dbl

SIP dbl pool used chunk entries number: 1

 entry_id          src_addr         dst_addr         remaining_time(sec)
 a4a051e0a4a1ebd   10.74.30.189     10.74.5.30               25
```

Step 5 **show platform hardware qfp {active | standby} feature alg statistics sip dblcfg**

Displays all SIP blacklist settings.

Example:

```
Device# show platform hardware qfp active feature alg statistics sip dblcfg

SIP dbl cfg pool used chunk entries number: 4
 dst_addr        trig_period(ms)    trig_size    block_time(sec)
 10.1.1.1              20              30             30
 10.74.5.30            1000             5             30
 192.0.2.2             60              30           2000
 198.51.100.115        20              30            388
```

Step 6 **show platform hardware qfp {active | standby} feature alg statistics sip processor**

Displays SIP processor settings.

Example:

```
Device# show platform hardware qfp active feature alg statistics sip processor

Session:     14        Global:    189

Current global wlock count:      0
```

Step 7 **show platform hardware qfp {active | standby} feature alg statistics sip timer**

Displays SIP timer settings.

Example:

```
Device# show platform hardware qfp active feature alg statistics sip timer

call-proceeding:    620      call-duration:    380
```

Step 8 **debug alg {all | info | trace | warn}**

Example:

```
Device# debug alg warn
```

Enables the logging of ALG warning messages.

Configuration Examples for SIP ALG Resilience to DoS Attacks

Example: Configuring SIP ALG Resilience to DoS Attacks

```
Device# configure terminal
Device(config)# alg sip processor session max-backlog 5
Device(config)# alg sip processor global max-backlog 5
Device(config)# alg sip blacklist trigger-period 90 trigger-size 30 destination 10.1.1.1
Device(config)# alg sip blacklist trigger-period 90 trigger-size 30 block-time 30
Device(config)# alg sip timer call-proceeding-timeout 35
Device(config)# alg sip timer max-call-duration 90
Device(config)# end
```

Additional References for SIP ALG Resilience to DoS Attacks

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
Firewall commands	• Cisco IOS Security Command Reference: Commands A to C • Cisco IOS Security Command Reference: Commands D to L • Cisco IOS Security Command Reference: Commands M to R • Cisco IOS Security Command Reference: Commands S to Z
NAT commands	IP Addressing Services Command References

Standards and RFCs

Standard/RFC	Title
RFC 4028	*Session Timers in the Session Initiation Protocol (SIP)*

MIBs

MIB	MIBs Link
	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

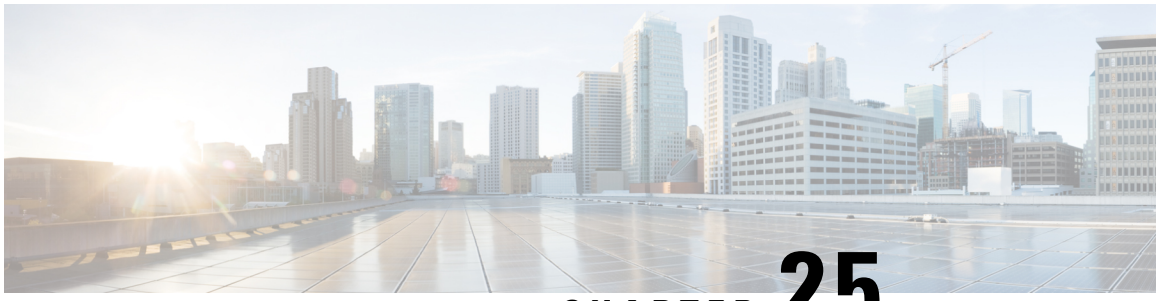
Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/support

Feature Information for SIP ALG Resilience to DoS Attacks

Table 33: Feature Information for SIP ALG Resilience to DoS Attacks

Feature Name	Releases	Feature Information
SIP ALG Resilience to DoS Attacks	Cisco IOS XE Release 3.11S	The SIP ALG Resilience to DoS Attacks feature provides protection against Session Initiation Protocol (SIP) denial of service (DoS) attacks. This feature supports a configurable lock limit, a dynamic blacklist, and configurable timers to prevent DoS attacks. Network Address Translation (NAT) and zone-based policy firewalls support this feature. In Cisco IOS XE Release 3.11S, the SIP ALG Resilience to DoS Attacks feature is implemented on Cisco ASR 1000 Series Aggregation Services Routers, Cisco Cloud Services Routers 1000V Series, and Cisco 4400 Series Integrated Services Routers. The following commands were introduced or modified: **alg sip processor**, **alg sip blacklist**, **alg sip timer**, **show alg sip**, **debug alg**, **debug platform software alg configuration all**, **set platform software trace forwarding-manager alg**, and **show platform hardware qfp feature alg statistics sip**.



CHAPTER 25

Match-in-VRF Support for NAT

The Match-in-VRF Support for NAT feature supports Network Address Translation (NAT) of packets that communicate between two hosts within the same VPN routing and forwarding (VRF) instance. In intra-VPN NAT, both the local and global address spaces for end hosts are isolated to their respective VPNs, and as a result, the translated addresses for the hosts overlap each other. The Match-in-VRF Support for NAT feature helps separate the address space for translated addresses among VPNs.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for Match-in-VRF Support for NAT

- The Match-in-VRF Support for NAT feature is not supported on interface overload configuration.
- The **match-in-vrf** keyword for intra-VPN NAT is not supported with CGN.

Information About Match-in-VRF Support for NAT

Match-in-VRF Support for NAT

In Cisco IOS XE Release 3.5S and later releases, the Match-in-VRF Support for NAT feature supports NAT of packets that communicate between two hosts within the same VPN.

The VRF-aware NAT enables communication between hosts in the private address space in different VPN routing and forwarding (VRF) instances and common servers in the Internet or the global domain. Because IP addresses of the inside hosts overlap with each other, the VRF-aware NAT facilitates communication between these hosts by converting overlapped inside IP addresses into globally unique addresses. The Match-in-VRF Support for NAT feature extends VRF-aware NAT by supporting intra-VPN NAT capability. In the intra-VPN NAT, both the local and global address spaces for end hosts are isolated to their respective VPNs, and as a result translated addresses for hosts overlap each other. To separate the address space for translated addresses among VPNs, configure the **match-in-vrf** keyword in the NAT mapping (**ip nat inside source** command) configuration. Both static and dynamic NAT configurations support the **match-in-vrf** keyword.

Note All NAT commands that support VRF support the **match-in-vrf** keyword. Because NAT outside rules (**ip nat outside source** command) support the match-in-VRF functionality by default, the **match-in-vrf** keyword is not supported by NAT outside rules.

In VRF-aware NAT, the IP alias and Address Resolution Protocol (ARP) entries for inside global addresses are configured in the global domain. For intra-VPN NAT, the IP alias and ARP entries for inside global addresses are configured in the VRF through which the translation happens. In intra-VPN NAT, configuration of the **match-in-vrf** keyword implies that at least one NAT outside interface is configured in the same VRF. The ARP entry in that VRF replies to the ARP request from the outside host.

If inside addresses are configured, the match-in-VRF is determined through inside mappings during the address translation of VRF traffic. If you have configured only outside mapping of IP addresses for address translations, the match-in-VRF will work. When a translation entry is created with both inside and outside mappings, the **match-in-vrf** keyword is determined by the inside mapping.

The Match-in-VRF Support for NAT feature supports the configuration of multiple dynamic mappings with the same IP address pool.

The following table provides you information about VRF support for NAT:

NAT Inside Interface	NAT Outside Interface
Global	Global IPv4 (non-MPLS)
MPLS IP	VRF **Note** You must use the **match-in-vrf** keyword in the configuration to indicate that communication is occurring within the VRF.

NAT Inside Interface	NAT Outside Interface
VRF	VRF **Note** Both VRFs must be in the same inside interface for this configuration to work.
VRF	MPLS **Note** You must use the **match-in-vrf** keyword in the configuration to indicate that communication is occurring within the VRF.
VRF	Global IPv4 (non-MPLS)

How to Configure Match-in-VRF Support for NAT

Configuring Static NAT with Match-in-VRF

Perform the following task to configure a static NAT translation and to enable NAT inside and outside traffic in the same VRF.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip nat inside source static** *local-ip global-ip* [**vrf** *vrf-name* [**match-in-vrf**]]
4. **interface** *type number*
5. **ip address** *ip-address mask* [**secondary**]
6. **ip nat inside**
7. **ip vrf forwarding** *vrf-name*
8. **exit**
9. **interface** *type number*
10. **ip address** *ip-address mask*
11. **ip nat outside**
12. **ip vrf forwarding** *vrf-name*
13. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ip nat inside source static** *local-ip global-ip* [**vrf** *vrf-name* [**match-in-vrf**]] **Example:** `Router(config)# ip nat inside source static 10.10.10.1 172.16.131.1 vrf vrf1 match-in-vrf`	Establishes static translation between an inside local address and an inside global address. • The **match-in-vrf** keyword enables NAT inside and outside traffic in the same VRF.
Step 4	**interface** *type number* **Example:** `Router(config)# interface gigabitethernet 0/0/1`	Specifies an interface and enters interface configuration mode.
Step 5	**ip address** *ip-address mask* [**secondary**] **Example:** `Router(config-if)# ip address 10.114.11.39 255.255.255.0`	Sets a primary IP address for an interface.
Step 6	**ip nat inside** **Example:** `Router(config-if)# ip nat inside`	Marks the interface as connected to the inside.
Step 7	**ip vrf forwarding** *vrf-name* **Example:** `Router(config-if)# ip vrf forwarding vrf1`	Associates a VRF with an interface or subinterface.
Step 8	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 9	**interface** *type number* **Example:** `Router(config)# interface gigabitethernet 0/0/0`	Specifies a different interface and enters interface configuration mode.
Step 10	**ip address** *ip-address mask* **Example:** `Router(config-if)# ip address 172.31.232.182 255.255.255.240`	Sets a primary IP address for an interface.
Step 11	**ip nat outside** **Example:** `Router(config-if)# ip nat outside`	Marks the interface as connected to the outside. **Note** NAT outside rules support the match-in-VRF functionality by default.

	Command or Action	Purpose
Step 12	**ip vrf forwarding** *vrf-name* **Example:** `Router(config-if)# ip vrf forwarding vrf1`	Associates a VRF with an interface or subinterface.
Step 13	**end** **Example:** `Router(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring Dynamic NAT with Match-in-VRF

Perform the following task to configure a dynamic NAT translation with the same address pool and to enable NAT inside and outside traffic in the same VRF.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip nat inside source list** *access-list-number* **pool** *pool-name* [**vrf** *vrf-name* [**match-in-vrf**]]
4. **access-list** *access-list-number* **permit source** [*source-wildcard*]
5. **ip nat inside source list** *access-list-number* **pool** *pool-name* **vrf** *vrf-name* [**match-in-vrf**]
6. **interface** *type number*
7. **ip address** *ip-address mask*
8. **ip nat inside**
9. **ip vrf forwarding** *vrf-name*
10. **exit**
11. **interface** *type number*
12. **ip address** *ip-address mask*
13. **ip nat outside**
14. **ip vrf forwarding** *vrf-name*
15. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**ip nat inside source list** *access-list-number* **pool** *pool-name* [**vrf** *vrf-name* [**match-in-vrf**]] **Example:** `Router(config)# ip nat inside source list 1 pool shared-pool vrf vrf1 match-in-vrf`	Enables multiple dynamic mappings to be configured with the same address pool. • The **match-in-vrf** keyword enables NAT inside and outside traffic in the same VRF.
Step 4	**access-list** *access-list-number* **permit source** [*source-wildcard*] **Example:** `Router(config)# access-list 1 permit 192.168.34.0 0.0.0.255`	Defines a standard access list permitting those addresses that are to be translated.
Step 5	**ip nat inside source list** *access-list-number* **pool** *pool-name* **vrf** *vrf-name* [**match-in-vrf**] **Example:** `Router(config)# ip nat inside source list 1 pool shared-pool vrf vpn1`	Establishes dynamic source translation, specifying the access list defined in the previous step.
Step 6	**interface** *type number* **Example:** `Router(config)# interface gigabitethernet 0/0/1`	Specifies an interface and enters interface configuration mode.
Step 7	**ip address** *ip-address mask* **Example:** `Router(config-if)# ip address 172.31.232.182 255.255.255.240`	Sets a primary IP address for an interface.
Step 8	**ip nat inside** **Example:** `Router(config-if)# ip nat inside`	Marks the interface as connected to the inside.
Step 9	**ip vrf forwarding** *vrf-name* **Example:** `Router(config-if)# ip vrf forwarding vpn1`	Associates a VRF with an interface or subinterface.
Step 10	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 11	**interface** *type number* **Example:** `Router(config)# interface gigabitethernet 0/0/0`	Specifies a different interface and enters interface configuration mode.
Step 12	**ip address** *ip-address mask* **Example:**	Sets a primary IP address for an interface.

	Command or Action	Purpose
	`Router(config-if)# ip address 172.31.232.182 255.255.255.240`	
Step 13	**ip nat outside** **Example:** `Router(config-if)# ip nat outside`	Marks the interface as connected to the outside. **Note** NAT outside rules support the match-in-VRF functionality by default.
Step 14	**ip vrf forwarding** *vrf-name* **Example:** `Router(config-if)# ip vrf forwarding vpn1`	Associates a VRF with an interface or subinterface.
Step 15	**end** **Example:** `Router(config-if)# end`	Exits interface configuration mode and returns to global configuration mode.

Configuration Examples for Match-in-VRF Support for NAT

Example: Configuring Static NAT with Match-in-VRF

The following example shows how to configure a static NAT translation between the local IP address 10.10.10.1 and the global IP address 172.16.131.1. The **match-in-vrf** keyword enables NAT inside and outside traffic in the same VRF.

```
Router# configure terminal
Router(config)# ip nat inside source static 10.10.10.1 172.16.131.1 vrf vrf1 match-in-vrf
Router(config)# interface gigabitethernet 0/0/1
Router(config-if)# ip address 10.114.11.39 255.255.255.0
Router(config-if)# ip nat inside
Router(config-if)# ip vrf forwarding vrf1
Router(config-if)# exit
Router(config)# interface gigabitethernet 0/0/0
Router(config-if)# ip address 172.31.232.182 255.255.255.240
Router(config-if)# ip nat outside
Router(config-if)# ip vrf forwarding vrf1
Router(config-if)# end
```

Example: Configuring Dynamic NAT with Match-in-VRF

The following example shows how to configure dynamic NAT mappings with the same address pool. The **match-in-vrf** keyword enables NAT inside and outside traffic in the same VRF.

```
Router# configure terminal
Router(config)# ip nat inside source list 1 pool shared-pool vrf vrf1 match-in-vrf
Router(config)# access-list 1 permit 192.168.34.0 0.0.0.255
Router(config)# ip nat inside source list 1 pool shared-pool vrf vpn1
Router(config)# interface gigabitethernet 0/0/1
Router(config-if)# ip address 172.31.232.182 255.255.255.240
```

```
Router(config-if)# ip nat inside
Router(config-if)# ip vrf forwarding vpn1
Router(config-if)# exit
Router(config)# interface gigabitethernet 0/0/0
Router(config-if)# ip address 172.31.232.182 255.255.255.240
Router(config-if)# ip nat outside
Router(config-if)# ip vrf forwarding vpn1
Router(config-if)# end
```

Additional References for Static NAT Mapping with HSRP

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
NAT commands: complete command syntax, command mode, command history, usage guidelines, and examples	*Cisco IOS IP Addressing Services Command Reference*
IP Access List Sequence Numbering	*IP Access List Sequence Numbering* document
NAT configuration tasks	"Configuring NAT for IP Address Conservation" module
NAT maintenance	"Monitoring and Maintaining NAT" module
Using NAT with MPLS VPNs	"Integrating NAT with MPLS VPNs" module

Standards and RFCs

Standard/RFC	Title
RFC 903	*Reverse Address Resolution Protocol*
RFC 826	*Ethernet Address Resolution Protocol: Or converting network protocol addresses to 48.bit Ethernet address for transmission on Ethernet hardware*
RFC 1027	*Using ARP to implement transparent subnet gateways*

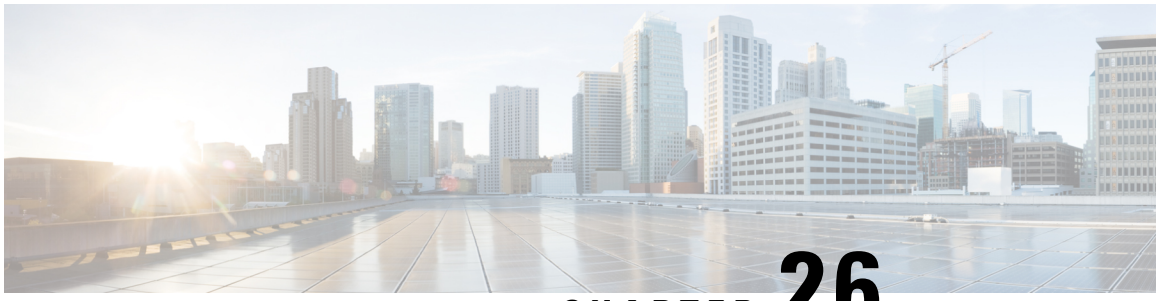
Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Match-in-VRF Support for NAT

Table 34: Feature Information for Match-in-VRF Support for NAT

Feature Name	Releases	Feature Information
Match-in-VRF Support for NAT	Cisco IOS XE Release 3.5S	The Match-in-VRF Support for NAT feature supports the NAT translation of packets that communicate between two hosts within the same VPN.



CHAPTER 26

IP Multicast Dynamic NAT

The IP Multicast Dynamic Network Address Translation (NAT) feature supports the source address translation of multicast packets. You can use source address translation when you want to connect to the Internet, but not all your hosts have globally unique IP addresses. NAT translates the internal local addresses to globally unique IP addresses before sending packets to the outside network. The IP multicast dynamic translation establishes a one-to-one mapping between an inside local address and one of the addresses from the pool of outside global addresses.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for IP Multicast Dynamic NAT

The IP Multicast Dynamic NAT feature does not support:

- IPv4-to-IPv6 address translation.
- Multicast destination address translation.
- Port Address Translation (PAT) overloading for multicast.
- Source and destination address translation.

- Unicast-to-multicast address translation.

Information About IP Multicast Dynamic NAT

How NAT Works

A device that is configured with NAT has at least one interface to the inside network and one to the outside network. In a typical environment, NAT is configured at the exit device between a stub domain and the backbone. When a packet exits the domain, NAT translates the locally significant source address into a globally unique address. When a packet enters the domain, NAT translates the globally unique destination address into a local address. If more than one exit point exists, each NAT must have the same translation table. If NAT cannot allocate an address because it has run out of addresses, it drops the packet. Then, NAT sends an Internet Control Message Protocol (ICMP) host unreachable packet to the destination.

Uses of NAT

NAT can be used for the following applications:

- When you want to connect to the Internet, but not all of your hosts have globally unique IP addresses. NAT enables private IP internetworks that use nonregistered IP addresses to connect to the Internet. NAT is configured on the router at the border of a stub domain (referred to as the *inside network*) and a public network such as the Internet (referred to as the *outside network*). NAT translates internal local addresses to globally unique IP addresses before sending packets to the outside network. As a solution to the connectivity problem, NAT is practical only when relatively few hosts in a stub domain communicate outside of the domain at the same time. When this is the case, only a small subset of the IP addresses in the domain must be translated into globally unique IP addresses when outside communication is necessary, and these addresses can be reused when they are no longer in use.
- When you must change your internal addresses. Instead of changing the internal addresses, which can be a considerable amount of work, you can translate them by using NAT.
- When you want to do basic load sharing of TCP traffic. You can map a single global IP address to many local IP addresses by using the TCP load distribution feature.

NAT Inside and Outside Addresses

The term *inside* in a Network Address Translation (NAT) context refers to networks owned by an organization that must be translated. When NAT is configured, hosts within this network have addresses in one space (known as the *local* address space). These hosts appear to those outside the network as being in another space (known as the *global* address space).

Similarly, the term *outside* refers to those networks to which the stub network connects, and which are not under the control of an organization. Also, hosts in outside networks can be subject to translation, and can thus have local and global addresses. NAT uses the following definitions:

- Inside local address—An IP address that is assigned to a host on the inside network. The address that the Network Information Center (NIC) or service provider assigns is probably not a legitimate IP address.

- Inside global address—A legitimate IP address (assigned by the NIC or service provider) that represents one or more inside local IP addresses to the outside world.
- Outside local address—The IP address of an outside host as it appears to the inside network. Not necessarily a legitimate address, it is allocated from the address space that is routable on the inside.
- Outside global address—The IP address that is assigned to a host on the outside network by the owner of the host. The address is allocated from a globally routable address or network space.

NAT supports the following VRFs:

Table 35: VRF NAT Support

NAT Inside Interface	NAT Outside Interface	Condition
Global VRF (also referred to as a non-VRF interface)	Global VRF (also referred to as a non-VRF interface)	Normal
VRF X	Global VRF (also referred to as a non-VRF interface)	When NAT is not configured for Match-in-VRF support. For more details, see the *Match-in-VRF Support for NAT* chapter.
VRF X	VRF X	When both inside and outside interfaces are in the same VRF, and NAT is configured with Match-in-VRF support.

This section describes the following topics:

- Inside Source Address Translation, on page 7
- Overloading of Inside Global Addresses, on page 8

Dynamic Translation of Addresses

Dynamic translation establishes a mapping between an inside local address and a pool of global addresses. Dynamic translation is useful when multiple users on a private network access the Internet. The dynamically configured pool IP address may be used as needed. It is released for use by other users when access to the Internet is no longer required.

Note When inside global or outside local addresses belong to a directly connected subnet on a NAT router, the router adds IP aliases for them. This action enables answer Address Resolution Protocol (ARP) requests. However, a situation can arise where the router itself answers packets that are not destined for it, possibly causing a security issue. This security issue can happen when an incoming Internet Control Message Protocol (ICMP) or UDP packet that is destined for one of the aliased addresses does not have a corresponding NAT translation in the NAT table. The router itself runs a corresponding service, for example, the Network Time Protocol (NTP). Such a situation might cause minor security risks.

How to Configure IP Multicast Dynamic NAT

Configuring IP Multicast Dynamic NAT

Note IP multicast dynamic translation establishes a one-to-one mapping between an inside local address and one of the addresses from the pool of outside global addresses

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip nat pool** *name start-ip end-ip* {**netmask** *netmask* | **prefix-length** *prefix-length*} [**type** {**match-host** | **rotary**}]
4. **access-list** *access-list-number* **permit** *source-address wildcard-bits* [**any**]
5. **ip nat inside source list** *access-list-number* **pool** *name*
6. **ip multicast-routing distributed**
7. **interface** *type number*
8. **ip address** *ip-address mask*
9. **ip pim sparse-mode**
10. **ip nat inside**
11. **exit**
12. **interface** *type number*
13. **ip address** *ip-address mask*
14. **ip pim sparse-mode**
15. **ip nat outside**
16. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**ip nat pool** *name start-ip end-ip* {**netmask** *netmask* | **prefix-length** *prefix-length*} [**type** {**match-host** | **rotary**}] **Example:** `Router(config)# ip nat pool mypool 10.41.10.1 10.41.10.23 netmask 255.255.255.0`	Defines a pool of global addresses to be allocated as needed.
Step 4	**access-list** *access-list-number* **permit** *source-address wildcard-bits* [**any**] **Example:** `Router(config)# access-list 100 permit 10.3.2.0 0.0.0.255 any`	Defines a standard access list for the inside addresses that are to be translated.
Step 5	**ip nat inside source list** *access-list-number* **pool** *name* **Example:** `Router(config)# ip nat inside source list 100 pool mypool`	Establishes dynamic source translation, specifying the access list defined in the prior step.
Step 6	**ip multicast-routing distributed** **Example:** `Router(config)# ip multicast-routing distributed`	Enables Multicast Distributed Switching (MDS).
Step 7	**interface** *type number* **Example:** `Router(config)# interface gigabitethernet 0/0/0`	Configures an interface and enters interface configuration mode.
Step 8	**ip address** *ip-address mask* **Example:** `Router(config-if)# ip address 10.1.1.1 255.255.255.0`	Sets a primary or secondary IP address for an interface.
Step 9	**ip pim sparse-mode** **Example:** `Router(config-if)# ip pim sparse-mode`	Enables sparse mode operation of Protocol Independent Multicast (PIM) on an interface.
Step 10	**ip nat inside** **Example:** `Router(config-if)# ip nat inside`	Indicates that the interface is connected to the inside network (the network that is subject to NAT translation).
Step 11	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode and enters global configuration mode.
Step 12	**interface** *type number* **Example:** `Router(config)# interface gigabitethernet 0/0/1`	Configures an interface and enters interface configuration mode.

	Command or Action	Purpose
Step 13	**ip address** *ip-address mask* **Example:** `Router(config-if)# ip address 10.2.2.1 255.255.255.0`	Sets a primary or secondary IP address for an interface.
Step 14	**ip pim sparse-mode** **Example:** `Router(config-if)# ip pim sparse-mode`	Enables sparse mode operation of PIM on an interface.
Step 15	**ip nat outside** **Example:** `Router(config-if)# ip nat outside`	Indicates that the interface is connected to the outside network.
Step 16	**end** **Example:** `Router(config-if)# end`	Exits interface configuration mode and enters privileged EXEC mode.

Configuration Examples for IP Multicast Dynamic NAT

Example: Configuring IP Multicast Dynamic NAT

```
Router# configure terminal
Router(config)# ip nat pool mypool 10.41.10.1 10.41.10.23 netmask 255.255.255.0
Router(config)# access-list 100 permit 10.3.2.0 0.0.0.255 any
Router(config)# ip nat inside source list 100 pool mypool
Router(config)# ip multicast-routing distributed
Router(config)# interface gigabitethernet 0/0/0
Router(config-if)# ip address 10.0.0.1 255.255.255.0
Router(config-if)# ip pim sparse-mode
Router(config-if)# ip nat inside
Router(config-if)# exit
Router(config)# interface gigabitethernet 0/0/1
Router(config-if)# ip address 10.2.2.1 255.255.255.0
Router(config-if)# ip pim sparse-mode
Router(config-if)# ip nat outside
Router(config-if)# end
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
NAT commands	Cisco IOS IP Addressing Services Command Reference
Configuring NAT for IP address conservation	Configuring NAT for IP Address Conservation module

Standards and RFCs

Standard/RFC	Title
None	—

MIBs

MIB	MIBs Link
None	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

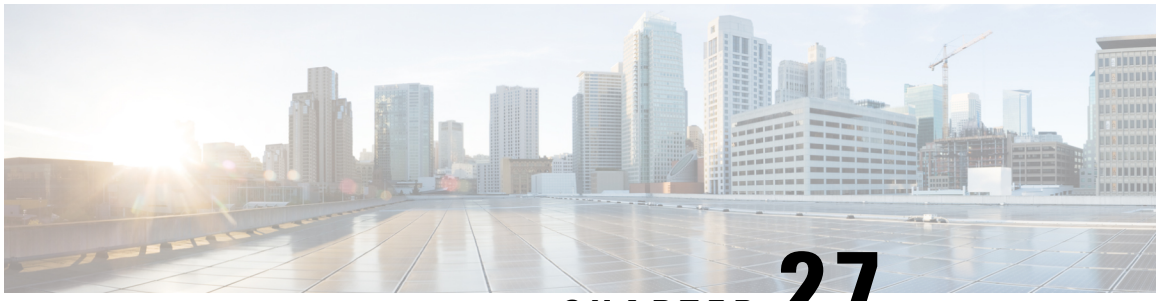
Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for IP Multicast Dynamic NAT

Table 36: Feature Information for IP Multicast Dynamic NAT

Feature Name	Releases	Feature Information
IP Multicast Dynamic NAT	Cisco IOS XE Release 3.4S	The IP Multicast Dynamic Network Address Translation feature supports the source address translation of multicast packets. You can use source address translation when you want to connect to the Internet, but not all your hosts have globally unique IP addresses. NAT translates the internal local addresses to globally unique IP addresses before sending packets to the outside network. The IP multicast dynamic translation establishes a one-to-one mapping between an inside local address and one of the addresses from the pool of outside global addresses.



CHAPTER 27

PPTP Port Address Translation

The PPTP Port Address Translation feature supports the Point-to-Point Tunneling Protocol (PPTP) application layer gateway (ALG) for Port Address Translation (PAT) configuration. PAT configuration requires the PPTP ALG to parse PPTP packets. The PPTP ALG is enabled by default when Network Address Translation (NAT) is configured.

This module provides information about how to configure the PPTP ALG for PAT.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for PPTP Port Address Translation

- The Point-to-Point Tunneling Protocol (PPTP) application layer gateway (ALG) does not support virtual TCP (vTCP) and TCP segments.
- The PPTP ALG will not work in Carrier Grade Network Address Translation (NAT) mode, when the NAT client and server use the same call ID.

Information About PPTP Port Address Translation

PPTP ALG Support Overview

The Point-to-Point Tunneling Protocol (PPTP) is a network protocol that enables the secure transfer of data from a remote client to an enterprise server by creating a VPN across TCP/IP-based data networks. PPTP encapsulates PPP packets into IP datagrams for transmission over the Internet or other public TCP/IP-based networks.

PPTP establishes a tunnel for each communicating PPTP network server (PNS)-PPTP Access Concentrator (PAC) pair. After the tunnel is set up, PPP packets are exchanged using enhanced generic routing encapsulation (GRE). A call ID present in the GRE header indicates the session to which a particular PPP packet belongs.

Network Address Translation (NAT) translates only the IP address and the port number of a PPTP message. Static and dynamic NAT configurations work with PPTP without the requirement of the PPTP application layer gateway (ALG). However, Port Address Translation (PAT) configuration requires the PPTP ALG to parse the PPTP header and facilitate the translation of call IDs in PPTP control packets. NAT then parses the GRE header and translates call IDs for PPTP data sessions. The PPTP ALG does not translate any embedded IP address in the PPTP payload. The PPTP ALG is enabled by default when NAT is configured.

NAT recognizes PPTP packets that arrive on the default TCP port, 1723, and invokes the PPTP ALG to parse control packets. NAT translates the call ID parsed by the PPTP ALG by assigning a global address or port number. Based on the client and server call IDs, NAT creates two doors based on the request of the PPTP ALG. (A door is created when there is insufficient information to create a complete NAT-session entry. A door contains information about the source IP address and the destination IP address and port.) Two NAT sessions are created (one with the server call ID and the other with the client call ID) for two-way data communication between the client and server. NAT translates the GRE packet header for data packets that complies with RFC 2673.

PPTP is a TCP-based protocol. Therefore, when NAT recognizes a TCP packet as a PPTP packet, it invokes the PPTP ALG parse-callback function. The PPTP ALG fetches the embedded call ID from the PPTP header and creates a translation token for the header. The PPTP ALG also creates data channels for related GRE tunnels. After ALG parsing, NAT processes the tokens created by the ALG.

PPTP Default Timer

The default timer for PPTP is 24 hours. This means that a generic routing encapsulation (GRE) session will live for 24 hours when deploying static and dynamic NAT. Based on your PPTP configuration and scaling requirement, you adjust the PPTP default timer.

Some PPTP clients and servers send keepalive messages to keep GRE sessions alive. You can adjust the NAT session timer for PPTP sessions by using the **ip nat translation pptp-timeout** command.

How to Configure PPTP Port Address Translation

Configuring PPTP ALG for Port Address Translation

The Point-to-Point Tunneling Protocol (PPTP) application layer gateway (ALG) is enabled by default when Network Address Translation (NAT) is configured. Use the **no ip nat service pptp** command to disable the PPTP ALG. Use the **ip nat service pptp** command to reenable PPTP ALG translation of applications.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ip nat inside**
5. **exit**
6. **interface** *type number*
7. **ip nat outside**
8. **exit**
9. **ip nat pool** *name start-ip end-ip* {**netmask** *netmask* | **prefix-length** *prefix-length*}
10. **ip nat inside source list** {*access-list-number* | *access-list-name*} **pool** *name* **overload**
11. **ip access-list standard** *access-list-name*
12. **permit** *host-ip*
13. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 0/0/1`	Enables an interface and enters interface configuration mode.
Step 4	**ip nat inside** **Example:** `Device(config-if)# ip nat inside`	Connects the interface to the inside network, which is subject to NAT.

	Command or Action	Purpose
Step 5	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and enters global configuration mode.
Step 6	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 0/1/0`	Enables an interface and enters interface configuration mode.
Step 7	**ip nat outside** **Example:** `Device(config-if)# ip nat outside`	Connects the interface to the outside network.
Step 8	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and enters global configuration mode.
Step 9	**ip nat pool** *name start-ip end-ip* {**netmask** *netmask* | **prefix-length** *prefix-length*} **Example:** `Device(config)# ip nat pool pptp-pool 192.168.0.1 192.168.0.234 prefix-length 24`	Defines a pool of IP addresses for NAT translations.
Step 10	**ip nat inside source list** {*access-list-number* | *access-list-name*} **pool** *name* **overload** **Example:** `Device(config)# ip nat inside source list pptp-acl pool pptp-pool overload`	Enables NAT of the inside source address. • When overloading is configured, the TCP or UDP port number of each inside host distinguishes between multiple conversations by using the same local IP address.
Step 11	**ip access-list standard** *access-list-name* **Example:** `Device(config)# ip access-list standard pptp-acl`	Defines a standard IP access list by name to enable packet filtering and enters standard access-list configuration mode.
Step 12	**permit** *host-ip* **Example:** `Device(config-std-nacl)# permit 10.1.1.1`	Sets conditions in named IP access lists that permit packets.
Step 13	**end** **Example:** `Device(config-std-nacl)# end`	Exits standard access-list configuration mode and enters privileged EXEC mode.

Configuration Examples for PPTP Port Address Translation

Example: Configuring PPTP ALG for Port Address Translation

```
Device# configure terminal
Device(config)# interface gigabitethernet 0/0/1
Device(config-if)# ip nat inside
Device(config-if)# exit
Device(config)# interface gigabitethernet 0/1/0
Device(config-if)# ip nat outside
Device(config-if)# exit
Device(config)# ip nat pool pptp-pool 192.168.0.1 192.168.0.234 prefix-length 24
Device(config)# ip nat inside source list pptp-acl pool pptp-pool overload
Device(config)# ip access-list standard pptp-acl
Device(config-std-nacl)# permit 10.1.1.1
Device(config-std-nacl)# end
```

Additional References for PPTP Port Address Translation

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
NAT commands	Cisco IOS IP Addressing Services Command Reference

Standards and RFCs

Standard/RFC	Title
RFC 2637	*Point-to-Point Tunneling Protocol (PPTP)*

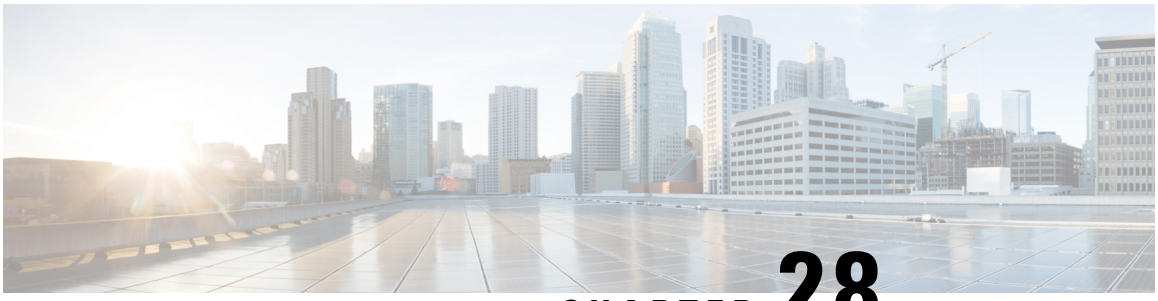
Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/support

Feature Information for PPTP Port Address Translation

Table 37: Feature Information for PPTP Port Address Translation

Feature Name	Releases	Feature Information
PPTP Port Address Translation Support	Cisco IOS XE Release 3.9S	The PPTP Port Address Translation Support feature introduces the Point-to-Point Tunneling Protocol (PPTP) application layer gateway (ALG) for Port Address Translation (PAT) configuration. PAT configuration requires the PPTP ALG to parse PPTP packets. The PPTP ALG is enabled by default when Network Address Translation (NAT) is configured. The following commands were introduced or modified: **debug platform hardware qfp feature alg datapath pptp**, **ip nat service pptp**, **show platform hardware qfp feature alg statistics pptp**.



CHAPTER 28

Network Address Translation Bindings

In Network Address Translation (NAT), the term binding describes the address binding between a local address and the global address to which the local address is translated. A binding is also called a half-entry. The different types of NAT bindings are static, dynamic, and non-PATable binds. The binding behavior of NAT is consistent across all Cisco platforms that use NAT.

This module describes the different types of NAT bindings.

Static NAT Binding

For every static mapping in a Network Address Translation (NAT) configuration, a single static binding is created. Static bindings contain the protocol number and local and global port numbers. Sessions (with full 5-tuple entries) are created when the traffic that matches the binding passes through NAT interfaces.

Use the **ip nat inside source static** command to configure static NAT, and then configure the **show ip nat translations verbose** command to display the NAT mapping ID.

The following is sample output from the **show ip nat translations verbose** command:

```
Device(config)# ip nat inside source static 10.1.1.2 10.2.1.2
Device(config)# exit
Device# show ip nat translations verbose

 Pro  Inside global         Inside local          Outside local         Outside global
 ---  10.1.1.2              10.2.1.2              ---                   ---
  create: 08/03/12 10:08:22, use: 08/03/12 10:08:22, timeout: 00:00:00
  Map-Id(In): 1
  Flags: static
  Mac-Address: 0000.0000.0000    Input-IDB:
  entry-id: 0x0, use_count:0
  Total number of translations: 1
```

Dynamic NAT Binding

Dynamic bindings are created when you configure dynamic Network Address Translation (NAT) without NAT overload configuration. In dynamic binding, the traffic matches the classification that is associated with the NAT mapping ID. Dynamic binding guarantees a one-to-one mapping between the local address and the global address. Each dynamic binding ages out when all its child sessions are aged out.

Use the **ipnat pool** command to configure dynamic NAT, and then use the **show ip nat translations verbose** command to display the mapping IDs.

The following is sample output from the **show ip nat translations verbose** command:

```
Device(config)# ip nat pool pool1 10.1.1.10 10.1.1.150 prefix-length 24
Device(config)# ip nat inside source list nat-list pool pool1
Device(config)# exit
Device# show ip nat translations verbose

Pro  Inside global         Inside local          Outside local         Outside global
---  10.1.1.10             10.2.1.2              ---                   ---
  create: 08/03/12 10:10:04, use: 08/03/12 10:10:04, timeout: 23:59:51
  Map-Id(In): 1
  Mac-Address: 0000.0000.0000    Input-IDB: GigabitEthernet0/3/1
  entry-id: 0x0, use_count:2

udp  10.1.1.10:16385       10.2.1.2:16385        10.1.1.1:4003         10.1.1.1:4003
  create: 08/03/12 10:10:04, use: 08/03/12 10:10:13, timeout: 00:05:00
  Map-Id(In): 1
  Mac-Address: 0000.0000.0000    Input-IDB: GigabitEthernet0/3/1
  entry-id: 0x8bc74680, use_count:1

udp  10.1.1.10:16384       10.2.1.2:16384        10.1.1.1:4003         10.1.1.1:4003
  create: 08/03/12 10:10:03, use: 08/03/12 10:10:13, timeout: 00:05:00
  Map-Id(In): 1
  Mac-Address: 0000.0000.0000    Input-IDB: GigabitEthernet0/3/1
  entry-id: 0x8bc745b0, use_count:1

Total number of translations: 3
```

Non-PATable Binds

Non-PATable binds are created when an application layer gateway (ALG) like Domain Name System (DNS) requests a NAT translation. When you have a dynamic NAT overload or Port-Address Translation (PAT) configuration, and have to translate a frame which is not PATable, non-PATable binds are created. A non-PATable frame is one that do not have any assigned port numbers. All IP headers are provided a protocol field; however, not every protocol is pattable. Cisco IOS XE NAT only handles PAT for the following protocols:

- Internet Control Message Protocol (ICMP)
- ESP_PROT
- Point-to-Point Tunneling Protocol (PPTP)
- TCP
- UDP

Note We recommend not to use non-PATable binds for overload configurations as these configurations have a one-to-one binding, which means that one local address consumes a single global address.

```
Device(config)# ip nat pool pool1 10.1.1.10 10.1.1.150 prefix-length 24
Device(config)# ip nat inside source list 1 pool pool1 overload
Device(config)# exit
Device# show ip nat translation verbose

Pro  Inside global         Inside local          Outside local         Outside global
---  10.1.1.10             10.1.1.2              ---                   ---
  create: 08/03/12 10:11:53, use: 08/03/12 10:11:53, timeout: 23:59:51
  Map-Id(In): 2
  Mac-Address: 0000.0000.0000    Input-IDB: GigabitEthernet0/3/1
  entry-id: 0x0, use_count:1

---  10.1.1.10             10.2.1.2              10.1.1.1              10.1.1.1
  create: 08/03/12 10:11:54, use: 08/03/12 10:12:03, timeout: 24:00:00
  Map-Id(In): 2
  Mac-Address: 0000.0000.0000    Input-IDB: GigabitEthernet0/3/1
  entry-id: 0x8bc74750, use_count:1

Total number of translations: 2
```

Recommendations on NAT Binding Configuration

When a packet arrives, a device (for example, Cisco ASR 1000 Series Aggregation Services Routers) uses the following steps to determine if the packet is subject to a Network Address Translation (NAT) rule to decide whether to use an existing translation entry, create a new translation entry, to not translate the packet. The device first checks the NAT translation table for a matching entry.

- If a matching entry is available, this entry is used for translation.
- If no matching entry is available, the device uses access control lists (ACLs) to find a match. A translation entry is created based on the configured match criteria and the IP address pool.

In the following sample dynamic Network Address Translation (NAT) configuration, the traffic that comes from the 172.16.0.0/24 network is translated by NAT and the traffic destined to 192.0.2.0/24 network is not translated.

```
Device(config)# ip nat pool NAT-POOL 10.98.198.1 10.98.198.15 netmask 255.255.255.240
Device(config)# ip nat inside source list NAT-ACL pool NAT-POOL overload
Device(config)# ip access-list extended NAT-ACL
Device(config-acl)# deny ip any 209.165.201.1 129.25.0.0 255.255.255.224
Device(config-acl)# deny ip any 192.0.2.0 144.118.0.0 255.255.255.0
Device(config-acl)# deny ip any 198.51.100.0 204.238.76.0 255.255.255.0
Device(config-acl)# deny ip any 10.0.0.0 255.0.0.0
Device(config-acl)# deny ip any 203.0.113.0 172.19.0.0 255.255.255.0
Device(config-acl)# deny ip any 192.168.0.0 255.255.0.0
Device(config-acl)# permit ip 172.16.0.0 255.240.0.0 any
```

The following is sample output from the **show ip nat translations inside** command:

```
Device# show ip nat translations inside 172.16.0.16
```

```
Pro  Inside global         Inside local          Outside local         Outside global
---  10.98.198.2           172.16.0.16           ---                   ---
udp  10.98.198.2:137       172.16.0.16:137       192.0.2.4             192.0.2.4
144.118.38.213:137
tcp  10.98.198.2:59901     172.16.0.16:59901     192.0.2.6             192.0.2.6
144.118.38.109:389
udp  10.98.198.2:123       172.16.0.16:123       206.246.122.250:123   206.246.122.250:123
```

When the first packet arrives from 172.16.0.16 to 192.0.2.0 and a translation entry does not exist for this packet, the packet is matched against the configured ACL, and it is not translated by NAT. When the next packet arrives from 172.19.0.16 to 192.0.2.0, then that packet is matched against the NAT binding and is translated.

However, when a Domain Name System (DNS), Lightweight Directory Access Protocol (LDAP), or Netbios packet arrives from 172.19.0.16 to one of the permitted hosts, the application layer gateway (ALG) creates a binding in the translation table. When an IP address that is neither the source or the destination address is embedded in a packet payload, and the packet does not have any port numbers (for example, DNS packet), the response packet also will have an IP address that is neither the source or the destination IP address. Traffic other than Internet Control Message Protocol (ICMP), TCP, and UDP can also create NAT bindings.

Using VRF-Aware Software Infrastructure to Bypass NAT

Use the VPN routing and forwarding (VRF)-Aware Software Infrastructure (VASI), to bypass traffic translation by Network Address Translation (NAT). In scenarios where traffic matches the deny statements in access control lists (ACLs), use VASI to bypass translation by NAT.

VASI is implemented as virtual interface pairs called Vasileft and Vasiright. These interface pairs are logically wired back-to-back and are symmetrical. For example, Vasileft1 interface and Vasiright1 interface are automatically paired. This means that a packet that enters Vasileft1 interface is internally handed over to the Vasiright1 interface without any user configuration. You can VASI for hairpinning, inter-VRF communication locally on the box, and so on. For more information about VASI, see “Configuring the VRF-Aware Software Infrastructure” module of the *Zone-Based Policy Firewall Configuration Guide*.

This section provides a sample configuration that shows how to bypass traffic translation by NAT.

- Configure an ACL with the list of subnets that must be bypassed by NAT. The traffic destined to the following subnets is not translated by NAT:
  - 209.165.201.0 255.255.255.224
  - 192.0.2.0 255.255.255.0
  - 198.51.100.0 255.255.255.0
  - 203.0.113.0 255.255.255.0
  - 192.168.0.0 255.255.0.0
- Configure policy-based routing (PBR) on the NAT inside interface to forward the traffic destined to the subnets listed above to the VASI interface. This traffic is routed from the VASI interface to the network.
- On the NAT inside interface, PBR takes precedence over NAT, and as a result, traffic that matches a PBR policy is forwarded to the Vasileft1 interface before it reaches NAT. The packet is then internally handed over to the Vasiright1 interface.
- On the NAT outside interface, the traffic appears as coming from the VASI interface that does not have a NAT configuration, and NAT translation is bypassed.

Because of traffic bypass configuration, the NAT configuration can remove all deny statements in the ACL; however, retain the permit statements:

```
ip nat inside source list NAT-ACL pool NAT-POOL overload
!
ip access-list extended NAT-ACL
 permit ip 172.19.0.0 0.0.0.255 any
```

The following is additional sample configuration required to bypass NAT translation through VASI:

```
!
interface GigabitEthernet0/0/0
 description nat outside interface
 ip address 10.2.1.1 255.255.255.0
 ip nat outside
!
interface GigabitEthernet0/0/1
 description nat inside interface
 ip address 10.2.2.1 255.255.255.0
 ip nat inside
 ip policy route-map no-NAT-rmap
!
interface vasileft1
 ip address 10.1.1.1 255.255.255.0
!
interface vasiright1
 ip address 10.1.2.1 255.255.255.0
!
ip access-list extended bypass-NAT
 permit ip any 209.165.201.0 255.255.255.224
 permit ip any 192.0.2.0 255.255.255.0
 permit ip any 198.51.100.0 255.255.255.0
 permit ip any 203.0.113.0 255.255.255.0
 permit ip any 192.168.0.0 255.255.0.0
!
route-map no-NAT-rmap permit 10
 match ip address bypass-nat
 set interface vasileft1
!
```