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 https://cfnng.cisco.com/. 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 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.

Supported Stateful NAT64 Scenarios

The following scenarios are supported by the Stateful NAT64 feature and are described in this section:

- Scenario 1—an IPv6 network to the IPv4 Internet
- Scenario 3—an IPv6 Internet to an IPv4 network
- Scenario 5—an IPv6 network to an IPv4 network

Scenario 1

An IPv6-only network that communicates with a global IPv4 Internet. This type of network is also called a green-field network. In a green-field enterprise network only the the border between its network and the IPv4 Internet can be modified.

Translation is performed between IPv4 and IPv6 packets in unidirectional or bidirectional flows that are initiated from an IPv6 host towards an IPv4 host. Port translation is necessary on the IPv4 side for efficient IPv4 address usage. The stateful translator can service an IPv6 network of any size.

Both Stateful NAT64 and Stateless NAT64 support Scenario 1.

Scenario 3

Scenario 3 shows a legacy IPv4 network that provide services to IPv6 hosts. IPv6-initiated communication can be achieved through stateful translation in this scenario.

Translation is preformed between IPv4 and IPv6 packets in unidirectional or bidirectional flows that are initiated from an IPv6 host towards an IPv4 host. The stateful translator can service an IPv4 network using either private or public IPv4 addresses.

Note Do not use the Well-Known Prefix (WKP) for Scenario 3, because it would lead to using the WKP with non-global IPv4 addresses. Use a network-specific prefix (example, /96 prefix) in Scenario 3. For more information, see *RFC 6052*, section "3.4 Choice of Prefix for Stateful Translation Deployments"

Scenario 5

This scenario has an IPv4 and IPv6 network within the same organization. The IPv4 addresses used are either public IPv4 addresses or RFC 1918-compliant addresses. IPv6 addresses are either public IPv6 addresses or Unique Local Addresses (ULAs) as specified by RFC 4193.

Translation is performed between IPv6 and IPv4 packets in unidirectional or bidirectional flows that are initiated from an IPv6 host towards an IPv4 host. The stateful translator can service both IPv6 and IPv4 networks of any size; however neither networks should not be the Internet.

Both Stateful NAT64 and Stateless NAT64 support Scenario 5.

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.

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**	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 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.

	Command or Action	Purpose
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.
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** **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:** `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.

	Command or Action	Purpose
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 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:**	Enables NAT64 IPv6-to-IPv4 address mapping.

	Command or Action	Purpose
	`Device(config)# nat64 prefix stateful 2001:DB8:1::1/96`	
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.
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.

	Command or Action	Purpose
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.
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.

	Command or Action	Purpose
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
```

Additional References

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
RFC 4291	*IP Version 6 Addressing Architecture*
RFC 6144	*Framework for IPv4/IPv6 Translation*
RFC 6052	*IPv6 Addressing of IPv4/IPv6 Translators*

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 Stateful Network Address Translation 64

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

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.

Table 1: Feature Information for Stateful Network Address Translation 64

Feature Name	Releases	Feature Information
Stateful Network Address Translation 64	15.4(2)T	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**.