



Stateful Network Address Translation 64

Last Updated: November 29, 2012

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, page 1](#)
- [Prerequisites for Configuring Stateful Network Address Translation 64, page 2](#)
- [Restrictions for Configuring Stateful Network Address Translation 64, page 2](#)
- [Information About Stateful Network Address Translation 64, page 2](#)
- [How to Configure Stateful Network Address Translation 64, page 9](#)
- [Configuration Examples for Stateful Network Address Translation 64, page 20](#)
- [Additional References, page 21](#)
- [Feature Information for Stateful Network Address Translation 64, page 23](#)
- [Glossary, page 24](#)

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 at the end of this module.

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.



Americas Headquarters:
Cisco Systems, Inc., 170 West Tasman Drive, San Jose, CA 95134-1706 USA

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.
- Stateful NAT64 supports only cold redundancy. There are two redundancy mechanisms: cold redundancy and hot redundancy. The redundancy mechanisms make the switchover of NAT64 boxes transparent to IPv6 hosts. In cold redundancy, the mapping of states is not synchronized among NAT64 boxes and the already established connections are interrupted during a switchover of NAT64 boxes.
- 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.

Information About Stateful Network Address Translation 64

- [Stateful Network Address Translation 64, page 3](#)
- [Prefixes Format for Stateful Network Address Translation 64, page 3](#)
- [Stateful IPv4-to-IPv6 Packet Flow, page 4](#)
- [Stateful IPv6-to-IPv4 Packet Flow, page 4](#)
- [IP Packet Filtering, page 5](#)
- [Differences Between Stateful NAT64 and Stateless NAT64, page 5](#)
- [High-Speed Logging for NAT64, page 5](#)
- [FTP64 Application-Level Gateway Support, page 7](#)
- [FTP64 NAT ALG Intrabox High Availability Support, page 8](#)
- [Stateful NAT64—Intrachassis Redundancy, page 8](#)

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 port overloading provides a 1:*n* mapping between IPv4 and IPv6 addresses.

According to the Behavior Engineering for Hindrance Avoidance (BEHAVE) drafts about the framework for Stateful NAT64 standards, when an IPv6 node initiates traffic through Stateful NAT64, and the incoming packet does not have an existing state, 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, page 4](#)

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 and described in the BEHAVE RFC draft.

- 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 1 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-ietf-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

Depending on your release, Stateful NAT64 supports high-speed logging (HSL). 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.

Configure the **nat64 logging translations flow-export v9 udp destination** command to enable NAT64 HSL logging.

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 2 *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 3 *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

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 4xx or 5xx 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.

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, page 9](#)
- [Configuring Dynamic Stateful Network Address Translation 64, page 12](#)
- [Configuring Dynamic Port Address Translation Stateful NAT64, page 15](#)
- [Monitoring and Maintaining a Stateful NAT64 Routing Network, page 19](#)

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	<p>enable</p> <p>Example: Device> enable</p>	<p>Enables privileged EXEC mode.</p> <ul style="list-style-type: none"> Enter your password if prompted.
Step 2	<p>configure terminal</p> <p>Example: Device# configure terminal</p>	<p>Enters global configuration mode.</p>
Step 3	<p>ipv6 unicast-routing</p> <p>Example: Device(config)# ipv6 unicast-routing</p>	<p>Enables the forwarding of IPv6 unicast datagrams.</p>
Step 4	<p>interface type number</p> <p>Example: Device(config)# interface gigabitethernet 0/0/0</p>	<p>Configures an interface type and enters interface configuration mode.</p>
Step 5	<p>description string</p> <p>Example: Device(config-if)# description interface facing ipv6</p>	<p>Adds a description to an interface configuration.</p>
Step 6	<p>ipv6 enable</p> <p>Example: Device(config-if)# ipv6 enable</p>	<p>Enables IPv6 processing on an interface.</p>
Step 7	<p>ipv6 address {ipv6-address/prefix-length prefix-name sub-bits/prefix-length}</p> <p>Example: Device(config-if)# ipv6 address 2001:DB8:1::1/96</p>	<p>Configures an IPv6 address based on an IPv6 general prefix and enables IPv6 processing on an interface.</p>
Step 8	<p>nat64 enable</p> <p>Example: Device(config-if)# nat64 enable</p>	<p>Enables NAT64 translation on an IPv6 interface.</p>

	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 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. <ul style="list-style-type: none"> 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.

Command or Action	Purpose
Step 17 <code>end</code> Example: <code>Device(config)# end</code>	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	<p>enable</p> <p>Example: Device> enable</p>	<p>Enables privileged EXEC mode.</p> <ul style="list-style-type: none"> • Enter your password if prompted.
Step 2	<p>configure terminal</p> <p>Example: Device# configure terminal</p>	<p>Enters global configuration mode.</p>
Step 3	<p>ipv6 unicast-routing</p> <p>Example: Device(config)# ipv6 unicast-routing</p>	<p>Enables the forwarding of IPv6 unicast datagrams.</p>
Step 4	<p>interface <i>type number</i></p> <p>Example: Device(config)# interface gigabitethernet 0/0/0</p>	<p>Configures an interface type and enters interface configuration mode.</p>
Step 5	<p>description <i>string</i></p> <p>Example: Device(config-if)# description interface facing ipv6</p>	<p>Adds a description to an interface configuration.</p>
Step 6	<p>ipv6 enable</p> <p>Example: Device(config-if)# ipv6 enable</p>	<p>Enables IPv6 processing on an interface.</p>
Step 7	<p>ipv6 {<i>ipv6-address/prefix-length</i> <i>prefix-name sub-bits/prefix-length</i>}</p> <p>Example: Device(config-if)# ipv6 2001:DB8:1::1/96</p>	<p>Configures an IPv6 address based on an IPv6 general prefix and enables IPv6 processing on an interface.</p>
Step 8	<p>nat64 enable</p> <p>Example: Device(config-if)# nat64 enable</p>	<p>Enables Stateful NAT64 translation on an IPv6 interface.</p>

Command or Action	Purpose
Step 9 <code>exit</code> Example: Device(config-if)# exit	Exits interface configuration mode and enters global configuration mode.
Step 10 <code>interface type number</code> Example: Device(config)# interface gigabitethernet 1/2/0	Configures an interface type and enters interface configuration mode
Step 11 <code>description string</code> Example: Device(config-if)# description interface facing ipv4	Adds a description to an interface configuration.
Step 12 <code>ip address ip-address mask</code> Example: Device(config-if)# ip address 209.165.201.24 255.255.255.0	Configures an IPv4 address for an interface.
Step 13 <code>nat64 enable</code> Example: Device(config-if)# nat64 enable	Enables Stateful NAT64 translation on an IPv4 interface.
Step 14 <code>exit</code> Example: Device(config-if)# exit	Exits interface configuration mode and enters global configuration mode.
Step 15 <code>ipv6 access-list access-list-name</code> Example: Device(config)# ipv6 access-list nat64-acl	Defines an IPv6 access list and enters IPv6 access list configuration mode.
Step 16 <code>permit ipv6 ipv6-address any</code> Example: Device(config-ipv6-acl)# permit ipv6 2001:DB8:2::/96 any	Sets permit conditions for an IPv6 access list.

Command or Action	Purpose
Step 17 <code>exit</code> Example: Device(config-ipv6-acl)# exit	Exits IPv6 access list configuration mode and enters global configuration mode.
Step 18 <code>nat64 prefix stateful ipv6-prefix/length</code> Example: Device(config)# nat64 prefix stateful 2001:DB8:1::1/96	Enables NAT64 IPv6-to-IPv4 address mapping.
Step 19 <code>nat64 v4 pool pool-name start-ip-address end-ip-address</code> Example: Device(config)# nat64 v4 pool pool1 209.165.201.1 209.165.201.254	Defines the Stateful NAT64 IPv4 address pool.
Step 20 <code>nat64 v6v4 list access-list-name pool pool-name</code> 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 <code>end</code> 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. <ul style="list-style-type: none"> • 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 <i>type number</i> Example: Device(config)# interface gigabitethernet 0/0/0	Configures an interface type and enters interface configuration mode.
Step 5	description <i>string</i> 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 { <i>ipv6-address/prefix-length</i> <i>prefix-name sub-bits/prefix-length</i> } 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 <i>type number</i> Example: Device(config)# interface gigabitethernet 1/2/0	Configures an interface type and enters interface configuration mode
Step 11	description <i>string</i> Example: Device(config-if)# description interface facing ipv4	Adds a description to an interface configuration.

Command or Action	Purpose
<p>Step 12 <code>ip address <i>ip-address mask</i></code></p> <p>Example: Device(config-if)# ip address 209.165.201.24 255.255.255.0</p>	Configures an IPv4 address for an interface.
<p>Step 13 <code>nat64 enable</code></p> <p>Example: Device(config-if)# nat64 enable</p>	Enables Stateful NAT64 translation on an IPv6 interface.
<p>Step 14 <code>exit</code></p> <p>Example: Device(config-if)# exit</p>	Exits interface configuration mode and enters global configuration mode.
<p>Step 15 <code>ipv6 access-list <i>access-list-name</i></code></p> <p>Example: Device(config)# ipv6 access-list nat64-acl</p>	Defines an IPv6 access list and places the device in IPv6 access list configuration mode.
<p>Step 16 <code>permit ipv6 <i>ipv6-address any</i></code></p> <p>Example: Device(config-ipv6-acl)# permit ipv6 2001:db8:2::/96 any</p>	Sets permit conditions for an IPv6 access list.
<p>Step 17 <code>exit</code></p> <p>Example: Device(config-ipv6-acl)# exit</p>	Exits IPv6 access list configuration mode and enters global configuration mode.
<p>Step 18 <code>nat64 prefix stateful <i>ipv6-prefix/length</i></code></p> <p>Example: Device(config)# nat64 prefix stateful 2001:db8:1::1/96</p>	Enables NAT64 IPv6-to-IPv4 address mapping.
<p>Step 19 <code>nat64 v4 pool <i>pool-name start-ip-address end-ip-address</i></code></p> <p>Example: Device(config)# nat64 v4 pool pool1 209.165.201.1 209.165.201.254</p>	Defines the Stateful NAT64 IPv4 address pool.

	Command or Action	Purpose
Step 20	nat64 v6v4 list <i>access-list-name pool pool-name overload</i> 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, page 20](#)
- [Example: Configuring Dynamic Stateful Network Address Translation 64, page 21](#)
- [Example: Configuring Dynamic Port Address Translation Stateful NAT64, page 21](#)

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(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)# 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
Framework for IPv4/IPv6 Translation	<i>Framework for IPv4/IPv6 Translation draft-ietf-behave-v6v4-framework-06</i>
FTP ALG for IPv6-to-IPv4 translation	<i>An FTP ALG for IPv6-to-IPv4 translation draft-ietf-behave-ftp64-06</i>
IP/ICMP Translation Algorithm	<i>IP/ICMP Translation Algorithm draft-ietf-behave-v6v4-xlate-10</i>
IPv6 Addressing of IPv4/IPv6 Translators	<i>IPv6 Addressing of IPv4/IPv6 Translators draft-ietf-behave-address-format-07</i>
RFC 2228	<i>FTP Security Extensions</i>
RFC 2373	<i>IP Version 6 Addressing Architecture</i>
RFC 2464	<i>Transmission of IPv6 Packets over Ethernet Networks</i>
RFC 2765	<i>Stateless IP/ICMP Translation Algorithm (SIIT)</i>
RFC 2766	<i>Network Address Translation - Protocol Translation (NAT-PT)</i>
RFC 4787	<i>Network Address Translation (NAT) Behavioral Requirements for Unicast UDP</i>
RFC 4966	<i>Reasons to Move the Network Address Translator - Protocol Translator (NAT-PT) to Historic Status</i>
RFC 6384	<i>An FTP Application Layer Gateway (ALG) for IPv6-to-IPv4 Translation</i>
Stateful NAT64: Network Address and Protocol Translation from IPv6 Clients to IPv4 Servers	<i>Stateful NAT64: Network Address and Protocol Translation from IPv6 Clients to IPv4 Servers draft-ietf-behave-v6v4-xlate-stateful-12</i>

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

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 4 Feature Information for Stateful Network Address Translation 64

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 3.5S 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	<p>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.</p> <p>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.</p>

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.

Cisco and the Cisco logo are trademarks or registered trademarks of Cisco and/or its affiliates in the U.S. and other countries. To view a list of Cisco trademarks, go to this URL: www.cisco.com/go/trademarks. Third-party trademarks mentioned are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company. (1110R)

Any Internet Protocol (IP) addresses and phone numbers used in this document are not intended to be actual addresses and phone numbers. Any examples, command display output, network topology diagrams, and other figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses or phone numbers in illustrative content is unintentional and coincidental.

© 2012 Cisco Systems, Inc. All rights reserved.