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.

In Cisco IOS-XE release 17.4 release, support is introduced to map a VRF to an IPv4 to IPv6 prefix mapping. Multiple source and destination prefix can be mapped to a VRF.

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

- Restrictions for Stateless Network Address Translation 64, on page 1
- Restrictions for Stateless Network Address Translation 64, on page 2
- Information About Stateless Network Address Translation 64, on page 2
- Support to Map a VRF to an IPv4 to IPv6 Prefix Mapping , on page 4
- How to Configure Stateless Network Address Translation 64, on page 5
- Configuring a VRF for Stateless NAT64 Translation, on page 13
- Configuration Examples for Stateless Network Address Translation 64, on page 16
- Additional References for Stateless Network Address Translation 64, on page 17
- Glossary, on page 17

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.

Restrictions for Stateless Network Address Translation 64

In addition to the restrictions indicated for Stateless Network Address Translation 64, the followingrestrictions are applicable to mapping a VRF to an IPv4 to IPv6 prefix mapping:

- High-Speed Logging is not supported.
- Multicast is not supported.
- This feature cannot be configured in a high availability scenario.
- This feature is only supported in autonomous mode.
- NAT64 static command is not supported for this feature.

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

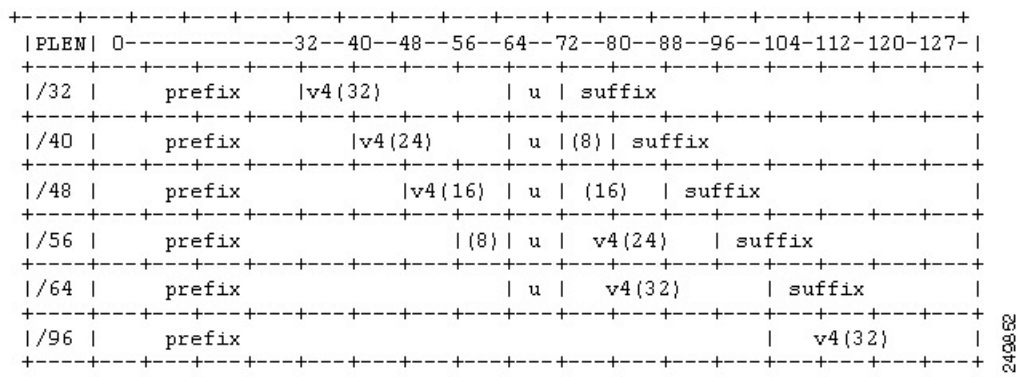
The IETF draft on the IP/ICMP translation algorithm describes the ICMP types or codes that should be translated between IPv4 and IPv6. ICMP errors embed the actual IP header and the transport header. Because the ICMP errors are embedded in the IP header, the IP header is not translated properly. For 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 1: 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.

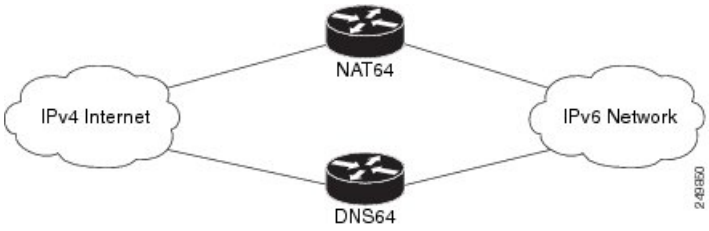
Supported Stateless NAT64 Scenarios

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.

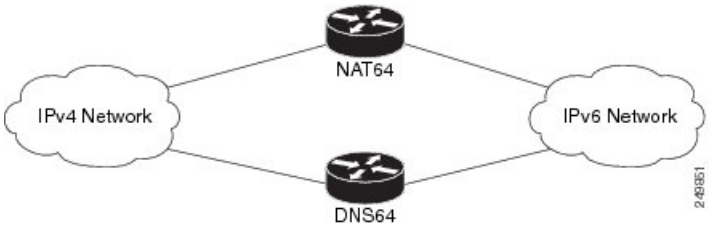
Figure 2: Stateless Translation for Scenarios 1 and 2



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 3: 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.

Support to Map a VRF to an IPv4 to IPv6 Prefix Mapping

The stateless NAT64 IPv4 to IPv6 prefix mappings is now VRF-aware. Additionally multiple source and destination prefix can be mapped to a VRF.

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:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
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::1/128`	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:**	Adds a description to an interface configuration.

	Command or Action	Purpose
	`Device(config-if)# description interface facing ipv4`	
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:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
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:**	Configures an IPv6 address based on an IPv6 general prefix and enables IPv6 processing on an interface.

	Command or Action	Purpose
	`Device(config-if)# ipv6 address 2001:DB8::1/128`	
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:** `Device(config-if)# nat64 prefix stateless v6v4 2001:0db8:0:1::/96`	Maps an IPv6 address to an IPv4 host for Stateless NAT 64 translation. • 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:**	Exits interface configuration mode and returns to global configuration mode.

	Command or Action	Purpose
	`Router(config-if)# exit`	
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.
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]
```

Configuring a VRF for Stateless NAT64 Translation

Perform this task to configure a VRF for IPv4 to IPv6 prefix mapping.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 unicast-routing**
4. **interface** *type number*
5. **description** *string*
6. **ipv6 address** {*ipv6-address*/*prefix-length* | *prefix-name sub-bits*/*prefix-length*}
7. **ipv6 enable**
8. **nat64 enable**
9. **exit**
10. **interface** *type number*
11. **ip address** *ip-address mask*
12. **negotiation auto**
13. **nat64 enable**
14. **exit**
15. **nat64 prefix stateless v4 v6** *ipv6-prefix*/*length src-prefix* /*dst-prefix* **vrf***dst-prefix*
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:** `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 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 7	**ipv6 enable** **Example:** `Device(config-if)# ipv6 enable`	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.

	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	**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 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:** `Device(config-if)# nat64 enable`	Enables Stateless NAT64 translation on an IPv4 interface.
Step 14	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 15	**nat64 prefix stateless v4 v6** *ipv6-prefix/length src-prefix /dst-prefix* **vrf***dst-prefix* **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** *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:**	Exits global configuration mode and returns to privileged EXEC mode.

	Command or Action	Purpose
	`Device(config)# end`	

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

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.