Deployment Guide

NAT64/DNS64

AX Series

DG_NAT64_032013.3

TABLE OF CONTENTS

1 INTRODUCTION

This deployment guide is the second in the series of IPv6 deployment guides and covers an example NAT64 deployment. This deployment guide builds upon the Carrier Grade NAT (CGN) Deployment Guide.

2 NAT64/DNS64 OVERVIEW

NAT64 is an IPv6 migration technology that provides native IPv6 clients with access to the IPv4 public Internet using translation technologies for TCP/UDP and ICMP.

NAT64 is implemented in an all-IPv6 access network; allowing a service provider to reclaim precious IPv4 address space and lay the foundation for an all-IPv6 infrastructure (*Figure 1*). IPv6 traffic from clients, destined for IPv4 servers, is translated into all-IPv4 packets by the NAT64 device. Source IP addresses and Layer 4 protocol ports are translated between IPv6 and IPv4 dynamically, creating a stateful binding in the NAT64 device between IPv6 and IPv4 transport addresses. All facets of the packet header including IP header, TCP/UDP header, and ICMP parameters are translated; this translation provides seamless, transparent connectivity between IPv6 clients and IPv4 servers.

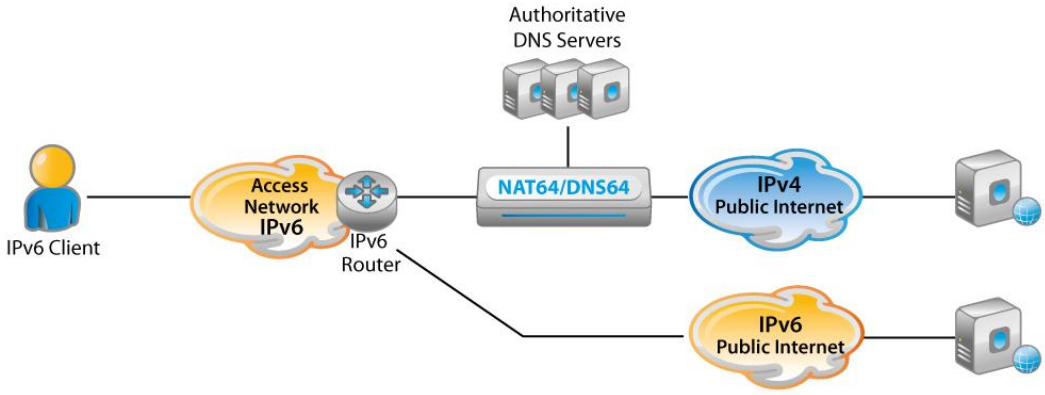


Figure 1: NAT64/DNS64 high-level architecture

A required complement to NAT64 translation is DNS64 functionality. The IPv6 client initiates communication with a host, using the host's fully qualified domain name (FQDN) creating an IPv6-specific AAAA resource request (RR) for the DNS server. If an IPv6 address is returned, the client connects using the given IPv6 address; communications between client and server will continue without any intervention from the NAT64/DNS64 device. However, if the DNS only contains an A record for the FQDN, the DNS64 process will synthesize an IPv6 address for the client, by appending the 32-bit IPv4 address to the configured NAT64 prefix and divert the client's communications to the NAT64 for translation between the

two protocols. The client now has an IPv6 destination address towards the NAT64 CGN device that corresponds with the desired FQDN. *Figure 3* shows an example.

NAT64/DNS64 is a mature and standardized technology, described in the following RFC documents:

- RFC 6147 - DNS64: DNS Extensions for Network Address Translation from IPv6 Clients to IPv4 Servers
- RFC 6146 - Stateful NAT64: Network Address and Protocol Translation from IPv6 Clients to IPv4 Servers
- RFC 6145 - IP/ICMP Translation Algorithm
- RFC 6052 - IPv6 Addressing of IPv4/IPv6 Translators

A10 Network's AX Series includes both the NAT64 translator and DNS64 server functionality, making NAT64 a logical and natural choice as the service provider begins to augment CGN with IPv6 migration technologies. NAT64 does not require specific customer-premise equipment (CPE) functionality, as do DS-Lite and 6rd, and it can operate concurrently (Interplay) with CGN on the AX device. With the AX Series, migration to IPv6 is made simple, with minimal disruption to the customer.

The AX Series implementation of NAT64 is unsurpassed in flexibility and can support the following models of connectivity concurrently:

- Standard NAT64 deployment providing IPv6-only client access to the IPv4 public Internet
- IPv4/IPv6 dual-stack and legacy device support through concurrent CGN and NAT64 operations

A10 Networks' NAT64 implementation also supports the same industry-leading CGN feature set, including stateful high availability, logging reduction techniques, endpoint-independent mapping and filtering, address-pool paired behavior (sticky NAT), override actions, hairpinning, user/session quotas, and NAT session/STUN timeouts. The *Advanced Configuration Options* section of the Carrier Grade NAT (CGN) Deployment Guide provides more information.

3 DEPLOYMENT EXAMPLE AND PACKET WALKTHROUGH

This section presents a typical NAT64/DNS64 topology, including detailed configuration for DNS64, and NAT64 address translation using both dynamic and deterministic ("fixed") NAT.

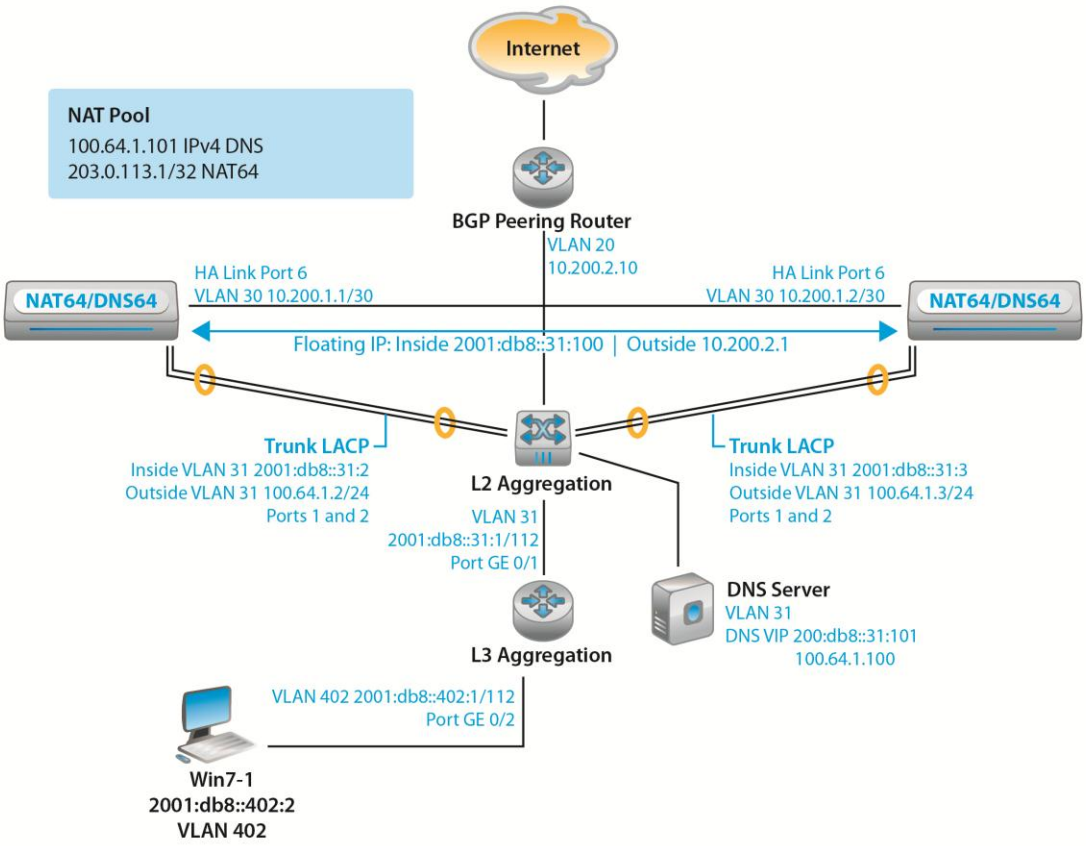


Figure 2: NAT64 reference topology

The configuration example in *Figure 2* illustrates a NAT64 deployment and consists of:

- Client Win 7-1, directly connected to the service provider's network.
- Layer 3 aggregation router.
- Layer 2 switch.
- Pair of AX devices running in High Availability (HA) mode, using link aggregation with LACP. A dedicated HA link is utilized for clarity and simplicity. However, the HA protocol also can use the LACP connections.
- BGP peering router, connected to the aggregation router to provide the connection to the Internet.

This example uses both static routing and dynamic routing protocols for redistributing the NAT pool and the floating IP address. IPv6 static routing is used between the aggregation layers and the AX device; BGP is configured between the BGP peering router and the AX device. The BGP peering router injects a default route towards the AX device. The AX device injects the NAT pool subnets, modifying the next hop to the outside floating IP address (10.200.2.1).

The AX device is configured with a static route for 2001:db8::402:0/112 towards the aggregation router. The aggregation router contains an IPv6 default route to the AX device's floating IP interface (2001:db8::31:100), and must use policy-based-routing to redirect traffic for translation to the AX device's inside floating IP address.

Finally, real IPv4 DNS servers are configured and represented to the clients with a virtual IP address of 2001:db8::31:101.

To further clarify, below is a brief packet walkthrough of the topology.

3.1 DNS64 WALKTHROUGH

DNS servers store IPv6 information in AAAA records. IPv4 addresses are stored in A records. In this example, the destination site (www.example.com) resides on an IPv4-only Web server on the Internet and hence, there is no AAAA record available for www.example.com. (See *Figure 3*)

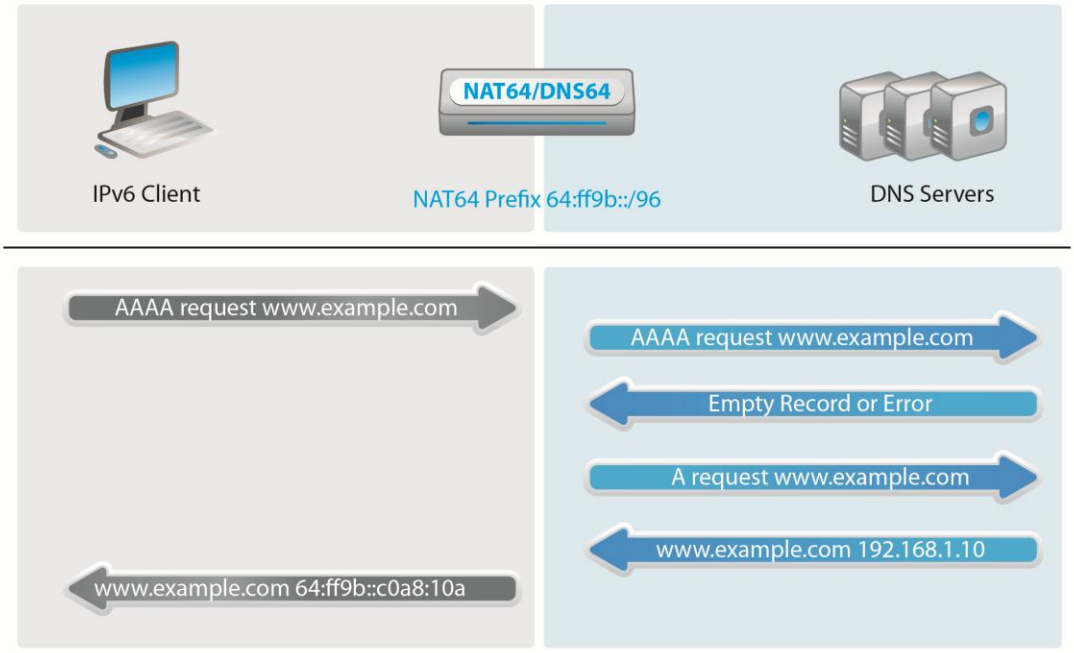


Figure 3: DNS64 simplified operation

1. The IPv6 client sends a DNS request to its designated DNS server (2001:db8::31:101), querying for the IPv6 address of www.example.com. As the destination is on another subnet, the client sends the packet to the Layer 3 aggregation router. The aggregation router routes the request to the active AX device.

2. The DNS64 server on the active AX device receives the AAAA query and forwards it to the configured local DNS server (100.64.1.100). To reach the DNS server, the source address is translated (using source NAT) to 100.64.1.101.

3. The DNS server returns an empty record or error indication, because there is no AAAA record available for www.example.com. After this, the AX device sends a new query, now for an A record (IPv4 address) for www.example.com.

4. The DNS server replies with an A record of 192.168.1.10.

5. The DNS64 feature now synthesizes a AAAA response for the IPv6 client, and represents the IPv4 server's IP address as 64:ff9b::c0a8:10a. This is the server's IPv4 address converted into IPv6 address notation (hexadecimal), and appended to the NAT64 prefix (64:ff9b::/96). Another way to denote an IPv4 address as part of an IPv6 address is 64:ff9b::192.168.1.10.

6. The AX device routes the synthesized response to the IPv6 client, which consequently sends all traffic for www.example.com to the AX device.

3.2 NAT64 PACKET WALKTHROUGH

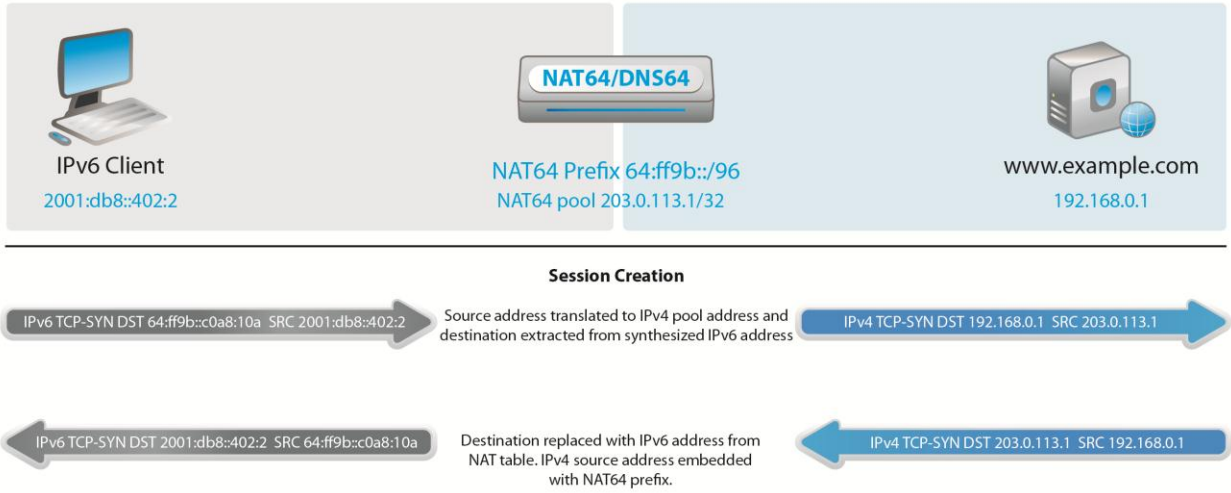


Figure 4: NAT64 simplified operation

1. The client sends an IPv6 TCP SYN to 64:ff9b::c0a8:10a.

   The aggregation router consults the routing tables and determines the packet should be sent to the HA floating IP address (2001:db8::31:100). Please refer to *Figure 2* for the topology.

2. The AX device receives the packet and the NAT64 process creates a NAT session for the client, replacing the client's IPv6 source address with an IPv4 address from the NAT pool (203.0.113.1/32). NAT64 also replaces the IPv6 destination address with the corresponding IPv4 address embedded within the IPv6 address (192.168.1.10). The AX device sends the translated packet to the server. The TCP SYN packet has source IP address 203.0.113.1 and destination address 192.168.1.10.

3. The server replies with an IPv4 SYN-ACK packet, with source IP address 192.168.1.10 and destination IP address 203.0.113.1.

4. The AX device consults the binding table, translates the IPv4 SYN-ACK into an IPv6 SYN-ACK, and forwards it towards the client. The SYN-ACK packet has source IP address 64:ff9b::c0a8:10a and destination address 2001:db8::402:2.

4 AX SERIES IPV6 CONFIGURATION

1. Configure an IPv6 IP address on the inside Virtual Ethernet (VE) port for VLAN 31:

```
AX(config)#interface ve 31
AX(config-if:ve31)#ipv6 address 2001:db8::31:2/112
```

Note: This guide builds upon the CGN configuration described in the *Base Configuration* section of the Carrier Grade NAT (CGN) Deployment Guide.

2. Add the IPv6 floating IP address to the HA configuration:

```
AX(config)#floating-ip 2001:db8::31:100 ha-group 1
```

3. Configure a static route to the access network:

```
AX(config)#ipv6 route 2001::402:0/112 2001:db8::31:1
```

4. Verify the IPv6 address settings:

```
AX#show ipv6 interfaces
Port    IPv6 Address                              Scope              Type
-------------------------------------------------------------------------------
ve31    2001:db8::31:2/112                        global             unicast
        fe80::20d:48ff:fe0a:69ca/64               link-local         unicast


AX#show ipv6 neighbor
Total IPv6 neighbor entries: 4
IPv6 Address              MAC Address     Type     Age  State       Interface    Vlan
-------------------------------------------------------------------------------
fe80::72ca:9bff:fe28:e701 70CA.9B28.E701  Dynamic  43   Reachable    ve31      31
2001:db8::31:3            000D.480A.6A6E  Dynamic  221  Reachable    ve31      31
2001:db8::31:1            70CA.9B28.E701  Dynamic  42   Reachable    ve31      31
fe80::20d:48ff:fe0a:6a6e  000D.480A.6A6E  Dynamic  237  Reachable    ve31      31
```

```
AX#show ipv6 traffic
Traffic Type    Received        Sent
-------------------------------------
Neigh Solicit   41340           16
Neigh Adverts   29993           129982
Echo Request    47              0
Echo Replies    15              47
Errors          18977           0

AX#show ipv6 route
IPv6 Routing Table
Codes: K - kernel route, C - connected, S - static, R - RIP, O - OSPF, I - IS-IS, B -
BGP
Timers: Uptime
C   2001:db8::31:0/112 via ::, ve 31, 03w0d00h
S   2001::402:0/112 [1/0] via 2001:db8::31:1, ve 31, 01w6d03h
```

4.1 DNS64 CONFIGURATION

Use the following steps to enable DNS64 operation on the AX device

1. Configure the NAT64 prefix.
2. Configure the IPv4 NAT pool, if DNS64 will be a proxy for a local IPv4 DNS server.
3. Configure a DNS template with DNS64 settings.
4. Add the configuration for the local DNS servers:
    a. Add a server configuration for each local IPv4 or IPv6 DNS, including UDP port.
    b. Add the DNS servers to a UDP service group.
    c. Add a DNS64 virtual IP address (VIP).
    d. Add virtual port type dns-udp to the VIP.
5. Bind the DNS template, service group, and NAT pools to the virtual port. When using an IPv6 NAT pool, use an IPv6 ACL to bind it to the virtual port.
6. Verify DNS Operation.

4.1.1 NAT64 PREFIX

The AX Series' NAT64 implementation supports both the NAT64 well-known prefix (64:ff9b::/32) and operator-configured prefix values. The HA group for which the prefix is assigned must be included for HA implementations. To configure NAT64 to use the well-known-prefix, use the following command:

```
AX(config)#nat64 prefix well-known ha-group-id 1
```

The well-known prefix is not globally routable, and an interface configured with the well-known prefix cannot participate in IPv6 path MTU discovery. For deployments that require a routable prefix, the operator may choose an address from the service provider's public IPv6 allocation. To configure the NAT64 prefix as 2001::face/96, for example, use the following command:

```
AX(config)#nat64 prefix 2001::face/96 ha-group-id 1
```

RFC 6052 defines prefix lengths of 32, 40, 48, 56, 64, and 96. The service provider should ensure the selected prefix length is in accordance with the RFC. *Figure 5* illustrates the prefix length and subsequent IPv4 placement.

Network Specific Prefix	IPv4 Address	IPv6 Address
2001:db8::/32	192.0.2.33	2001:db8:c000:221::
2001:db8:100::/40	192.0.2.33	2001:db8:1c0:2:21::
2001:db8:122::/48	192.0.2.33	2001:db8:122:c000:2:2100::
2001:db8:122:300::/56	192.0.2.33	2001:db8:122:3c0:0:221::
2001:db8:122:344::/64	192.0.2.33	2001:db8:122:344:c0:2:2100::
2001:db8:122:344::/96	192.0.2.33	2001:db8:122:344::192.0.2.33

Figure 5: NAT64 prefix (RFC 6052)

4.1.2 DNS64 CONFIGURATION

1. To configure the IPv4 NAT pool, use the following command at the global configuration level. This NAT pool is not for NAT64 functionality, but provides an IPv4 source address for the DNS request generated by DNS64 on behalf of the IPv6 client. You must define the beginning and ending addresses of the pool and include the appropriate HA group.

   ```
   AX(config)#ip nat pool ipv4-pool1 100.64.1.101 100.64.1.101 netmask /24
   ha-group-id 1
   ```

2. Configure a DNS template with DNS64 settings. Declare a name for the template at the global configuration level, for example “dns64-temp”.

```
AX(config)#slb template dns dns64-temp
```

3. At the config-dns configuration level, enter the **dns64** command:

```
AX(config-dns)#dns64
```

4. Create a local DNS server configuration and define the server name, IP address, and port number:

```
AX(config)#slb server localdns-rs 100.64.1.100
AX(config-real server)#port 53 udp
```

5. Add the local DNS server to the service group:

```
AX(config)#slb service-group dns53 udp
AX(config-slb svc group)#member localdns-rs:53
```

6. Create a virtual server and configure its IPv6 address. This is the IP address used by the clients to reach the DNS service. Bind the service group, template, and source NAT pool to the virtual server instance:

```
AX(config)#slb virtual-server vs1 2001:db8::31:101
AX(config-slb vserver)#ha-group 1
AX(config-slb vserver)#port 53 dns-udp
AX(config-slb vserver-vport)#source-nat pool ipv4-pool1
AX(config-slb vserver-vport)#service-group dns53
AX(config-slb vserver-vport)#template dns dns64-temp
AX(config-slb vserver-vport)#exit
AX(config-slb vserver)#exit
```

If the **ha-group 1** command is omitted, the active and standby AX devices perform duplicate address detection (DAD) using the same virtual IP address (VIP), and indicate a duplicate address problem in the log.

7. Use the commands shown in the following section to verify proper DNS64 operation.

4.1.3 VERIFY DNS OPERATION

The following examples show how to verify DNS operation.

Verify Local DNS Server Operation

```
AX(config)#show slb server
Total Number of Services configured: 3
                  Current = Current Connections, Total = Total Connections
                  Fwd-pkt = Forward packets, Rev-pkt = Reverse packets
Service                   Current    Total      Fwd-pkt    Rev-pkt    Peak-conn  State
-------------------------------------------------------------------------------
localdns-rs:53/udp        0          8726       12208      19081      0          Up
localdns-rs: Total        0          8726       12208      19081      0          Up
```

Verify DNS Query Send/Receive and Translation

```
AX(config)#show dns64 statistics
DNS Service Type: dns64
Query      Q-Parallel Q-Passive  Q-Changed  Q-Bad
Response   Translated Cache      Dropped    R-Bad      R-Error    R-Empty
-------------------------------------------------------------------------------
5078       0          0          2620       0
4553       2106       0          15         0          15         0
```

Use AXdebug To Verify DNS Query Send/Receive

```
AX(config)#axdebug
AX(config-axdebug)#filter 1
AX(config-axdebug-filter:1)#port 53 53
AX(config-axdebug-filter:1)#exit
(config-axdebug)#capture brief
```

This trace illustrates the proper operation of DNS64. **Sample Output:**

1. `@456755512 i( 2,  31, f7a6b)> ipv6 2001:db8::402:2 > 2001:db8::31:101 udp 64314  > 53 length 41`

   This is the initial AAAA DNS query from the client to the DNS64 virtual IP address owned by the AX NAT64 device.

2. `@456755512 o( 2,  31, f7a6b)> ip 100.64.1.101 > 100.64.1.100 udp  10849  > 53 length 41`

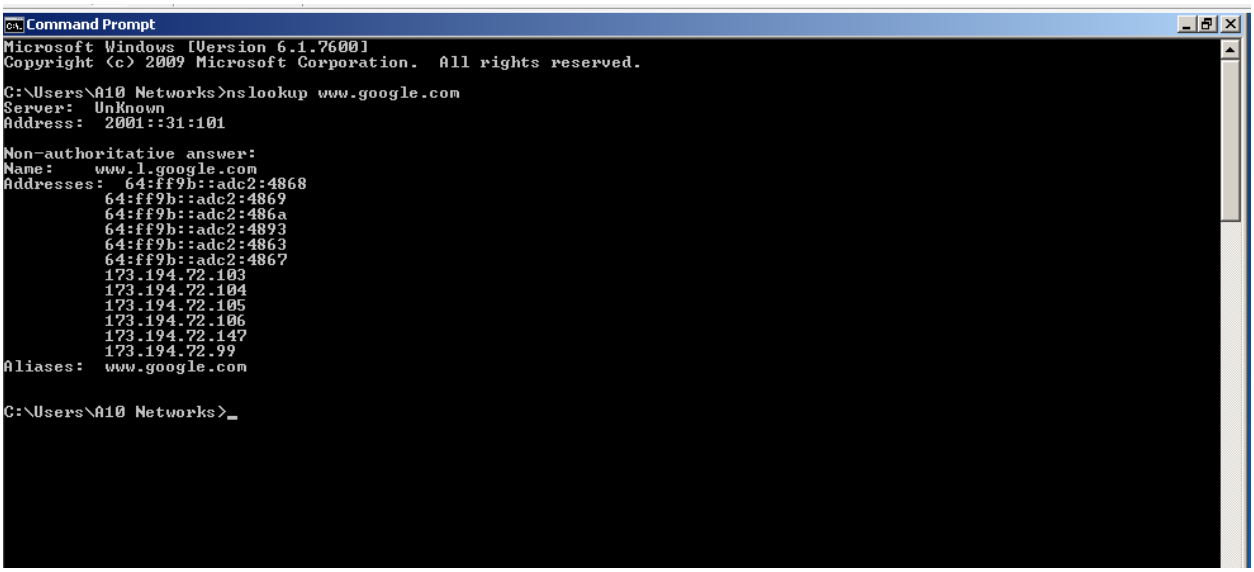
This is the request from the AX device using the translated source address (taken from the IPv4 NAT pool) to the local IPv4 DNS server.

3. `@456755786 i( 2,  31, bcfa9)> ip 100.64.1.100 > 100.64.1.101 udp  53  > 10849 length 415`
   This is the reply from the local DNS server.

4. `@456755786 o( 1,  31, bcfa9)> ipv6 2001:db8::31:101 > 2001:db8::402:2 udp 53  > 64314 length 415`
   This is the relayed reply on the inside IPv6 network towards the client. If R-errors are increasing (use **show dns64 statistics**) and this packet trace is completing as illustrated, the DNS server log should be consulted to determine the issue. This situation can arise if the proper subnets are not allowed in the BIND configuration.

Use client DNS tools to Verify DNS Resolution Is Working Properly

In the following example, DNS64 is returning the synthesized response constructed from the NAT64 prefix, 64:ff9b::/96 and the hexadecimal representation of the IPv4 address of the FQDN.



4.2 DNS64 CONFIGURATION OPTIONS

The following options are available for configuring DNS64.

4.2.1 DNS64 FOR LOCAL IPV6 DNS SERVER

The configuration above assumes that the DNS server is using an IPv4 IP stack. The AX device also provides the option to use IPv6 DNS servers, or both IPv4 and IPv6 DNS servers, concurrently. To utilize IPv6 DNS servers, create an IPv6 NAT pool to enable the DNS VIP to reach the local IPv6 DNS server, and declare a real server configuration for the local IPv6 DNS server.

For example, to configure a local IPv6 DNS server with IP address 2001:db8::1000 and VIP 2001:db8::200, and use a source NAT address of 2001:db8::50, enter the following configuration. Note that changes from the IPv4 DNS local server configuration are emphasized in bold Italics.

```
AX(config)#ipv6 nat pool ipv6-pool1 2001:db8::50 4629::50 netmask 64
AX(config)#slb template dns dns64-temp
AX(config-dns)#dns64
AX(config-dns)#exit
AX(config)#slb server localdns-rs 2001:db8::1000
AX(config-real server)#port 53 udp
AX(config-real server-node port)#exit
AX(config-real server)#exit
AX(config)#slb service-group dns53 udp
AX(config-slb svc group)#member localdns-rs:53
AX(config-slb svc group)#exit
AX(config)#slb virtual-server vs1 2001:db8::200
AX(config-slb vserver)#port 53 dns-udp
AX(config-slb vserver-vport)#source-nat pool ipv6-pool1
AX(config-slb vserver-vport)#service-group dns53
AX(config-slb vserver-vport)#template dns dns64-temp
AX(config-slb vserver-vport)#exit
AX(config-slb vserver)#exit
```

4.2.2 DNS64 FOR BOTH IPV6 AND IPV4 LOCAL DNS SERVERS

The AX Series DNS64 implementation allows for simultaneous IPv4 and IPv6 local DNS server deployment. The operator must define NAT pools for both IPv4 and IPv6, define the real servers and add them to the DNS service group, and configure an IPv6 access control list (ACL) to redirect traffic to the proper server.

For example, to configure simultaneous access to both IPv4 and IPv6 local DNS servers addressed as 100.64.1.100 and 2001:db8::1000, and with translated source addresses 100.64.1.101 and 2001:db8::50 respectively, use the following commands:

```
AX(config)#ipv6 access-list dnslist
AX(config-access-list:dnslist)#permit tcp any any
AX(config-access-list:dnslist)#permit udp any any
AX(config-access-list:dnslist)#permit icmp any any
AX(config-access-list:dnslist)#permit ipv6 any any
AX(config-access-list:dnslist)#exit
AX(config)#ipv6 nat pool ipv6-pool1 2001:db8::50 2001:db8::50 netmask 64
AX(config)#ip nat pool ipv4-pool1 100.64.1.101 100.64.1.101 netmask /24 ha-group-id 1
AX(config)#slb template dns dns64-temp
AX(config-dns)#dns64
AX(config-dns)#exit
AX(config)#slb server localdns-rs1 2001:db8::1000
AX(config-real server)#port 53 udp
AX(config-real server-node port)#exit
AX(config-real server)#exit
AX(config)#slb server localdns-rs2 100.64.1.100
AX(config-real server)#port 53 udp
AX(config-real server-node port)#exit
AX(config-real server)#exit
AX(config)#slb service-group dns53 udp
AX(config-slb svc group)#member localdns-rs1:53
AX(config-slb svc group)#member localdns-rs2:53
AX(config-slb svc group)#exit
AX(config)#slb virtual-server vs1 2001:db8::31:101
AX(config-slb vserver)#port 53 dns-udp
AX(config-slb vserver-vport)#source-nat pool ipv4-pool1
AX(config-slb vserver-vport)#access-list name dnslist source-nat-pool ipv6-pool1
AX(config-slb vserver-vport)#service-group dns53
AX(config-slb vserver-vport)#template dns dns64-temp
AX(config-slb vserver-vport)#exit
AX(config-slb vserver)#exit
```

4.3 NAT64 CONFIGURATION

Use the following steps to configure NAT64:

1. Configure a NAT pool (or group of pools) containing the IPv4 addresses to use for translating traffic from IPv6 clients to IPv4 servers.
2. Configure a Limit ID (LID) and add the pool or pool group.
3. Configure a class list that matches on IPv6 client addresses and maps to the LID.
4. Configure the NAT64 prefix.
5. Bind the class list to the NAT64 feature. The class list applies to packets from the inside NAT interface to the outside NAT interface.

   Note: *Only a single class list can be used for this purpose.*
6. Enable IPv6 inside NAT on the interface connected to the IPv6 clients.
7. Enable IPv4 outside NAT on the interface connected to the IPv4 Internet.

Detailed Steps

1. To configure the NAT pool for IPv6 to IPv4 translation, include the pool name, addresses, and HA group. Enter the following command at the global configuration level.

   ```
   AX(config)#ip nat pool 6to4_1 203.0.113.1 203.0.113.1 netmask /28  ha-group-id
   1 lsn
   ```

   Note: *The **lsn** option is required for this pool group.*

   Alternatively, NAT pools can be combined into pool groups. This simplifies future changes and allows non-contiguous address bundling. Use the command below to create a pool group. Declare a pool-group name (“example”) and list NAT pools to be included in the group (“6to4_1”).

   ```
   AX(config)#ip nat pool-group example 6to4_1 ha-group-id 1
   ```

2. Create the NAT64 LID. The LID associates the NAT64 pool or pool groups with specific configuration options including user quota, override, and rule lists. The operator can specify up to 1024 LIDs. Begin the configuration by assigning a LID number. This enters the LSN-LID configuration level.

   ```
   AX(config)#lsn-lid 2
   ```

3. Specify the NAT pool or pool groups to assign to this LID:

   ```
   AX(config-lsn-lid)#source-nat-pool 6to4_1
   ```

4. Specify optional parameters for this NAT pool. (See *NAT64 Configuration Options* for more details.)

```
AX(config)#user-quota icmp 50
AX(config)#user-quota udp 250 reserve 0
AX(config)#user-quota tcp 250 reserve 0
```

5. Create the class list and specify the internal subnets and hosts that will be associated with a specific LID. In this example, the class list named “6to4” matches on all IPv6 addresses and is tied to the configuration in LID 2.

```
AX(config)#class-list 6to4
AX(config)#::/0 lsn-lid 2
```

Note: *The wildcard IPv6 address (::/0) is supported. Any IPv6 subnet or host also can be configured within the class list.*

6. The NAT64 prefix is configured as part of the DNS64 configuration. (Please refer to DNS64 Configuration step 1 for more details.)

7. Bind the class list to the NAT64 process:

```
AX(config)#nat64 inside source class-list 6to4
```

8. Declare the interfaces for NAT operation. Inside NAT is configured for client-side interfaces, and outside NAT is configured for public, Internet-facing interfaces.

```
AX(config)#interface ve 20
AX(config-if:ve20)#ip nat outside
AX(config)#interface ve 31
AX(config-if:ve31)#ipv6 nat inside
```

9. Use the commands shown in the following section to verify proper NAT64 operation.

4.3.1 VERIFY NAT64 OPERATION

Display the Session Table

```
AX#show session
Traffic Type                        Total
-------------------------------------------
TCP Established                     10
TCP Half Open                       0
UDP                                 0
Non TCP/UDP IP sessions             0
```

```
Other                                0
Reverse NAT TCP                      0
Reverse NAT UDP                      0
Curr Free Conn                       66845614
Conn Count                           28422
Conn Freed                           28412
TCP SYN Half Open                    0
Conn SMP Alloc                       0
Conn SMP Free                        0
Conn SMP Aged                        0
Conn Type 0 Available                133038080
Conn Type 1 Available                66845614
Conn Type 2 Available                33337761
Conn Type 3 Available                16629760
Conn SMP Type 0 Available            133038080
Conn SMP Type 1 Available            66519040
Conn SMP Type 2 Available            33275902
Conn SMP Type 3 Available            16637928
```

Prot	Forward Source	Forward Dest	Reverse Source	Reverse Dest	Age	Hash	Flags
Tcp	[2001:db8::402:2]:57165	[64:ff9b::3f97]:80	63.151.118.161:80	203.0.113.1203.0.113.1:1460	300	1	NS
Tcp	[2001:db8::402:2]:57155	[64:ff9b::adc2]:80	173.194.72.100:80	203.0.113.1203.0.113.1:1315	300	1	NS
Tcp	[2001:db8::402:2]:57140	[64:ff9b::adc2]:80	173.194.72.99:80	203.0.113.1203.0.113.1:1185	300	1	NS
Tcp	[2001:db8::402:2]:57156	[64:ff9b::adc2]:80	173.194.72.120:80	203.0.113.1203.0.113.1:1316	300	2	NS
Tcp	[2001:db8::402:2]:57157	[64:ff9b::adc2]:80	173.194.72.120:80	203.0.113.1203.0.113.1:1497	300	3	NS
Tcp	[2001:db8::402:2]:57162	[64:ff9b::adc2]:80	173.194.72.120:80	203.0.113.1203.0.113.1:1502	300	3	NS
Tcp	[2001:db8::402:2]:57163	[64:ff9b::adc2]:80	173.194.72.138:80	203.0.113.1203.0.113.1:1138	300	4	NS
Tcp	[2001:db8::402:2]:57148	[64:ff9b::adc2]:80	173.194.72.120:80	203.0.113.1:1273	300	4	NS
Tcp	[2001:db8::402:2]:57158	[64:ff9b::adc2]:80	173.194.72.94:80	203.0.113.1:1368	300	4	NS
Tcp	[2001:db8::402:2]:57154	[64:ff9b::adc2]:80	173.194.72.100:80	203.0.113.1:1244	300	5	NS

```
Total Sessions: 10
```

Display the NAT64 Statistics

```
AX#show nat64 stat
NAT64 Statistics:
-------------------
Total TCP Ports Allocated              3858
Total TCP Ports Freed                  3858
Total UDP Ports Allocated              3
```

```
Total UDP Ports Freed                    3
Total ICMP Ports Allocated               175
Total ICMP Ports Freed                   175
Data Session Created                     4034
Data Session Freed                       4034
User-Quota Created                       164
User-Quota Freed                         164
User-Quota Creation Failed               0
TCP NAT Port Unavailable                 0
UDP NAT Port Unavailable                 0
ICMP NAT Port Unavailable                0
New User NAT Resource Unavailable        0
TCP User-Quota Exceeded                  95
UDP User-Quota Exceeded                  0
ICMP User-Quota Exceeded                 0
Extended User-Quota Matched              0
Extended User-Quota Exceeded             0
Data Session User-Quota Exceeded         0
TCP Full-cone Session Created            3858
TCP Full-cone Session Freed              3858
UDP Full-cone Session Created            3
UDP Full-cone Session Freed              3
Full-cone Session Creation Failed        0
Hairpin Session Created                  0
Self-Hairpinning Drop                    0
Endpoint-Independent Mapping Matched     0
Endpoint-Independent Filtering Matched   0
Endpoint-Dependent Filtering Drop        0
NAT Pool Mismatch Drop                   0
TCP Port Overloaded                      0
UDP Port Overloaded                      0
TCP Port Overloading Session Created     0
UDP Port Overloading Session Created     0
TCP Port Overloading Session Freed       0
UDP Port Overloading Session Freed       0
Layer 3 Forwarded Packets                0
Source Address Prefix Match Drop         0
LSN LID Drop                             0
LSN LID Pass-through                     0
No Class-List Match                      0
```

4.4 FIXED-NAT CONFIGURATION

Fixed-NAT is a NAT64 feature that allocates NAT ports for each client from a predetermined ("fixed") set of ports on the NAT address. Since each client using Fixed-NAT receives a deterministic set of ports, a client can be identified without any need for logging. Each individual client can be identified based solely on the NAT IP address and the port numbers within the client's fixed, unvarying allocation of ports.

Fixed-NAT supports a simplified single command-line structure. To enable Fixed-NAT, use the following command from the global configuration level. Specify the inside address range and netmask, outside address range and netmask, ports per user, and HA group.

```
AX(config)#fixed-nat inside 2001:db8::402:2 2001:db8::402:2 netmask 128 nat
203.0.113.1 203.0.113.1 netmask /28 ports-per-user 500
```

Note: *The* ***ports-per-user*** *option allows the operator to manually configure the port block allocation per inside address. If this option is omitted, the software automatically calculates the number of ports for allocation based upon the number of inside and outside address ports available.*

Fixed-NAT configuration is covered extensively in the Carrier Grade NAT (CGN) Deployment Guide. Please see that guide for more details.

4.5 NAT64 CONFIGURATION OPTIONS

The AX Series' implementation of NAT64 includes additional configurable options.

4.5.1 NAT64 ALG SUPPORT

The AX Series NAT64 implementation provides ALG support for the following protocols:

- FTP
- RTSP
- SIP
- TFTP

FTP is supported by default. To enable additional ALG support for a protocol, use a command such as the following:

```
AX(config)#nat64 alg ftp enable
```

For SIP NAT mappings, the Session Traversal Utilities for NAT (STUN) timeout value is set to "Expires". For SIP RTP/RTCP NAT mappings, the corresponding full-cone session's STUN timeout is configurable.

The RTP/RTSP STUN timeout can be 2-10 minutes. The default is 5 minutes. For example, to change the RTP/RTCP-STUN timeout for SIP NAT mappings to 10 minutes, use the following command at the global configuration level:

```
AX(config)#ip nat lsn alg sip rtp-stun-timeout 10
```

4.5.2 FRAGMENTATION

By default, fragmentation of inbound IPv6 packets is enabled, and fragmentation of inbound IPv4 packets is disabled. If an inbound IPv4 packet has the Don't Fragment (DF) bit set, the AX device does not fragment the packet, and instead returns an ICMP unreachable message. Fragmentation of outbound IPv4 packets is enabled.

The following fragmentation options can be configured for NAT64 operation:

- DF-bit-transparency – Adds an empty IPv6 fragmentation header if the IPv4 DF bit is zero. By default, this option is disabled.
- Nat64 fragmentation inbound – Controls fragmentation options from NAT outside to NAT inside interfaces. The following options are available:
  - Drop – Drops inbound fragmented packets.
  - IPv6 – Enables fragmentation of IPv6 packets; ***this is the default behavior.***
  - df-set – If DF bit is set, there are additional options:
    - Drop.
    - Fragment IPv6 packets.
    - Send ICMP type 3 code 4 (fragmentation needed but DF set); ***this is the default behavior.***
- Nat64 fragmentation outbound – Controls fragmentation options from NAT inside to NAT outside interfaces. The following options are available:
  - Drop – Drops outbound fragmented packets.
  - IPv4 – Enables fragmentation of outbound IPv4 packets. ***This is the default behavior.***
  - send-icmpv6 – Enables sending of ICMPv6 unreachable messages for outbound IPv6 fragmented packets.

For example, to configure NAT64 fragmentation behavior so that all fragmented packets in the outbound direction are dropped, enter the following command at the global configuration level:

```
AX(config)#nat64 fragmentation outbound drop
```

4.5.3 NAT64/DNS64 OVERRIDE OPTIONS

The AX NAT64 implementation supports specific override actions for both NAT64 and DNS64 elements. NAT64 shares CGN NAT44 override options including NAT, drop, and pass-through. These options and their configuration are discussed in detail in the Carrier Grade NAT (CGN) Deployment Guide.

In addition, the DNS64 implementation supports the following override behaviors:

- Disable – Does not perform any DNS64 processing of the client's DNS request. The client's request is forwarded to the DNS server, and the reply is sent to the client without modification.
- Different prefix – Uses a different NAT64 prefix to synthesize IPv6 addresses in the reply to the client. This option allows for NAT64 load balancing.
- Exclude answer – Drops AAAA replies that contain specific IPv6 addresses or prefixes. In this case, the AX device sends an A query on behalf of the client, then uses DNS64 to add synthesized IPv6 addresses to the reply before sending the reply to the client.

Please refer to the *IPv4-to-IPv6 Transition Solutions Guide* for additional information and for configuration examples.

4.5.4 ADDITIONAL NAT64 OPTIONS

The AX Series' NAT64 implementation shares many features in common with CGN. These features include:

- NAT session timeouts
- Port preservation
- Endpoint-Independent Mapping (EIM)
- Endpoint-Independent Filtering (EIF)
- Hairpinning
- NAT IP selection method
- Port Batching
- Port Overloading
- STUN Timeout

- SYN timeout
- User and session quotas

These configuration options can be found at the configuration levels for LSN LIDs (lsn-lid), NAT (ip nat), and CGN (ip nat lsn). Please refer to the Carrier Grade NAT (CGN) Deployment Guide for more information and configuration details.

5 LOGGING

The AX Series' NAT64 implementation supports all CGN log-file sizes and volume reduction features including Compact (hex), Binary, and RFC 5424 formats; as well as log batching and Fixed-NAT logging. Standard Syslog is supported, as well as Syslog over TCP and logging to RADIUS. Please refer to the Carrier Grade NAT (CGN) Deployment Guide for more operational details and configuration guidelines.

The following log output illustrates NAT64 session-creation and port-mapping logs in the default logging format (ASCII):

```
Aug 21 20:22:23 AX3030-T-47 NAT-TCP-N: [2001:db8::402:2]:54620<--
>[64:ff9b::6289:3517]:443, 98.137.53.23:443<-->203.0.113.1:54620#015
Aug 21 20:22:25 AX3030-T-47 NAT-TCP-D: [2001:db8::402:2]:54620<--
>[64:ff9b::6289:3517]:443, 98.137.53.23:443<-->203.0.113.1:54620#015
Aug 21 20:23:37 AX3030-T-47 NAT-TCP-F: [2001:db8::402:2]:54608 ->
203.0.113.1:54608#015
Aug 21 20:24:07 AX3030-T-47 NAT-TCP-C: [2001:db8::402:2]:54621 ->
203.0.113.1:54621#015
```

6 NAT64 NETWORK INTEGRATION

The network illustrated in *Figure 2* makes use of both static routing and dynamic routing protocols, for redistribution of IP routes for NAT pools and floating IP addresses. IPv6 static routing is used between the aggregation layer and the AX device; BGP is used between the BGP peering router and the AX device. The BGP peering router injects a default route towards the AX device. The AX device injects the NAT pool subnets, modifying the next hop to the outside floating IP address (10.200.2.1).

The AX device is configured with a static route for 2001::402:0/112 towards the aggregation router. The aggregation router contains an IPv6 default route to the floating IP interface (2001:db8::31:100), and must use policy-based-routing to redirect traffic that is intended for translation to the AX device's inside floating IP address.

The AX Series supports dynamic routing in the IPv6 access network using BGP, OSPF, or IS-IS routing protocols. Using these protocols allows basic reachability; however, by default, they do not redistribute the well-known NAT64 IP address range throughout the access network. The details and configurations

for the IPv4 “outside” network, including the BGP peering router, are covered in the Carrier Grade NAT (CGN) Deployment Guide.

The access network must have reachability to both the DNS64 VIP and the NAT64 prefix, for translation to take place. In this example, OSPF is used to illustrate the redistribution configuration task.

6.1 ENABLE OSPFV3 ROUTING

1. Enter the interface configuration level of the interface to support OSPFv3:

```
AX(config)#interface ve 31
```

2. Enable IPv6 processing:

```
AX(config-if:ve31)#ipv6 enable
```

3. Configure OSPFv3 with initial parameters:

```
AX(config-if:ve31)#ipv6 router ospf area 0 tag 1 instance-id 0
```

4. Access the OSPFv3 router instance configuration level:

```
AX(config-if:ve31)#exit
AX(config)#router ipv6 ospf 1
```

5. Declare the router ID:

```
AX(config-router)#router-id 1.1.1.1
```

6.2 REDISTRIBUTE NAT64 INFORMATION

1. Enable redistribution of the HA floating IP address into OSPFv3. The following command controls the redistribution of the HA group’s floating IP address, ensuring reachability to the HA pair:

```
AX(config-router)#redistribute floating-ip
```

2. Enable redistribution of the virtual IP address of the DNS64 instance into OSPFv3. The following command controls the redistribution of the DNS64 virtual IP address allowing clients to reach the DNS64 server:

```
AX(config-router)#redistribute vip 2001:db8::31:101 floating-IP-forward-address
2001:db8::31:100
```

6.2.1 FLOATING IP VIRTUAL MAC IN HA DEPLOYMENTS

It is considered a best practice to include the floating IP forwarding address. However, if the **ha-group** command is included in VIP configuration as demonstrated in this guide, the IPv6 neighbor entry for the VIP will be the virtual MAC address of the floating IP, allowing proper operation after HA events for directly connected aggregation routers.

Example:

```
AX(config)#slb virtual-server vs1 2001:db8::31:101
AX(config -slb vserve)#ha-group 1
```

The IPv6 neighbor in the downstream aggregation router shows that DNS64 VIP is mapped to the correct virtual MAC address.

```
Router2#show ipv6 neighbors vrf cgn
IPv6 Address                              Age Link-layer Addr State Interface
2001:db8::402:2                             1 000c.29e7.abbf  STALE Gi0/2.3
2001:db8::31:100                            3 021f.a000.0001  STALE Gi0/1.2
2001:db8::31:101                            1 021f.a000.0001  STALE Gi0/1.2
```

Redistribute the NAT64 Prefix

The NAT64 prefix must be redistributed with the next hop set as the floating IP address (2001:db8::31:100). To accomplish this, create a route map to modify the next hop and refer to the route map with the **redistribution** command:

```
AX(config)#route-map ipv6fip permit 1
AX(config-route-map)#set ipv6 next-hop 2001:db8::31:100
AX(config-route-map)#exit
AX(config)#router ipv6 ospf 1
AX(config-router)#redistribute nat64 route-map ipv6fip
```

7 CONCLUSION

This NAT64 Deployment Guide is the second in the series of IPv6 deployment guides. This deployment guide builds upon the first deployment guide, which covers Carrier-Grade NAT(CGN/CGNAT)/Large Scale NAT(LSN). This NAT64 deployment guide shows how to configure a sample NAT64 network configuration, and how to configure the AX Series.