



APPLICATION DELIVERY

Deploying NAT64 and DNS 64 with the Brocade ServerIron ADX and Secure64 DNS Cache Platforms

Provides reference architecture and procedures for deployment of NAT64 and DNS64 in support of **IPv6-only client environments**



SECURE64

BROCADE

CONTENTS

Introduction	3
Documentation Note	3
Overview	4
Secure64 DNS Cache	4
Brocade ServerIron ADX	5
Solution Architecture	6
Solution Topology 1: In-line	6
Solution Topology 2: Routed	7
Configuration Tasks for In-Line Topology	8
Configuration for the Brocade ServerIron ADX	8
Configuration for the Secure64 DNS Cache Server	9
Configuring IP Addresses and Routing for the Secure64 DNS Cache Server	9
Editing the Configuration File	9
Configuration Tasks for Routed Topology	11
Configuration for the Brocade ServerIron ADX	11
Configuration for the Secure64 DNS Cache Server	11
Additional DNS64 Functionality Options with Secure64 DNS Cache	12
Configuration of distinct IPv6 prefixes with Views	12
Configuration of multiple “sticky” IPv6 prefixes	12
Conclusion	13
Brocade Sales:	13
Secure64 Sales:	13
Brocade Support:	13
Secure64 Support:	13
Appendix A: Running Configurations	14
Example Brocade Configuration for In-Line Topology	14
Example Brocade Configuration for Routed Topology	15
Example Secure64 Configuration for Both Topologies	15

INTRODUCTION

As part of a pragmatic approach to deploying IPv6, Network Address Translation can be used as a mechanism to bridge connectivity gaps between IPv4-only and IPv6-only endpoints. The various use-cases and topologies included therein are collectively referred to as “NAT64”.

This document focuses on one general NAT64 use-case: v6-only clients that require access to both v6 and v4 resources. Common examples of this use-case include:

- Broadband ISP deployments (DOCSIS 3.0, xDSL, FTTx) conserving limited IPv4 resources by deploying an IPv6-only access tier
- Mobile Smartphone providers wishing to widely deploy IPv6 to customer devices
- Utility device networks, such as “smart grid” devices requiring access to existing networks
- IPv6-capable Set-Top Box (STB) networks requiring access to legacy resources

Two topology examples are included: The “In-line” topology is a great starting point for lab trials, whereas the “Routed” topology is ideal for large-scale deployments.

The minimum hardware required to test these examples includes:

- One ADX 1000, 4000, or 10000 with the –PREM routing code license (which includes NAT64 Gateway functionality), running code release 12.3.1 or later
- One Secure64 DNS Cache appliance
- Two or more commodity PCs to serve as test clients and/or servers
- Your own dual-stacked IPv4 & IPv6-enabled router, preferably connected to routable IPv4 & IPv6 networks

Documentation Note

This document does not attempt to replicate content already available in the formal product documentation; rather, this document focuses on Best Practices and common implementation concepts. Many other use-cases and deployment scenarios are possible with this equipment.

The content in this document assumes general familiarity with basic Layer 2 and Layer 3 network concepts, such as IPv4, IPv6, VRRP/VRRP-e, Spanning Tree, VLANs, trunks, and Link Aggregation Groups. Comprehensive instructions on these topics can be found in the primary product documentation.

For more information on the Brocade® ServerIron®, visit <http://Brocade.com/adx>

For more information on Secure64 DNS Cache, visit <http://www.Secure64.com>

For product documentation, community forums, and other helpful information, visit <http://my.brocade.com>

OVERVIEW

NAT64 in this case requires the use of DNS64, which is the synthesis of usable IPv6 DNS records for IPv4-only hostnames.

Following are examples of normal DNS behavior under various addressing schemes:

- In a typical IPv4 client-to-server transaction, an IPv4-only client requests its local IPv4 DNS server to resolve a hostname to an IPv4 address, which is mapped via an **A record** in DNS
- Similarly, in a typical IPv6 client-to-server transaction, an IPv6-only client requests its local IPv6 DNS server to resolve a hostname to an IPv6 address which is mapped via an **AAAA record** in DNS
- A client with both IPv4 and IPv6 addresses (called 'dual-stack') will request both, and **will prefer the AAAA record** response if provided with both
- But when an IPv6-only client requests a hostname that only has an **A record** (effectively an IPv4-only server), the client connection fails

DNS64 solves this v6-to-v4 issue by synthesizing temporary IPv6 AAAA records for IPv4-only requests.

The basic DNS64 process flow is described here:

- In the DNS server, administrators specify a 96-bit IPv6 prefix
- This 96-bit prefix is then routed to a NAT64 Gateway (either via static or dynamic routing)
- The NAT64 Gateway is configured to accept any traffic destined for that prefix
- When a v6-only client requests name resolution for a site that only answers with an IPv4 **A record**, the prefix is added to the IPv4 address response to form a **temporary AAAA record** that is usable only on the local network
- Thus the address format is: **<IPv6 DNS64 prefix, 96 bits>:<IPv4 address, 32 bits, converted to hex>**
That is, the IPv4 address is effectively embedded within a complete IPv6 address
- Following routing, the client sends its request to the NAT64 Gateway, which creates a dynamic NAT session and strips off the 96-bit prefix to reveal the original IPv4 destination
- The NAT64 Gateway then creates a session on behalf of the client from a designated set of routed IPv4 addresses (called a dynamic NAT pool), thus completing the connection
- Ultimately, two "conversations" are established at this point: Client-to-NAT64 Gateway (v6-side), and NAT64 Gateway (v4 side)-to-Server

Additional notes on the example configurations:

- Do not announce the NAT64 prefix used in these examples (64:ff9b:/96) to external routing peers; this "well-known" prefix is for use within your own network and is similarly used by others
- Note that synthesized AAAA records, due to their dynamic nature, are not cached. The Secure64 DNS Cache does cache the authoritative paths in order to accelerate future lookups

Secure64 DNS Cache

Secure64 DNS Cache is a caching and validating DNS resolver that runs on the HP Integrity server hardware platform. The application is specifically designed for environments where security, performance, DNSSEC validation, and attack resistance are highly important. The Secure64 DNS Cache application runs on the specialized Secure64 SourceT micro operating system. The micro OS employs a secure architecture that makes applications immune to compromise from rootkits and malware, and resistant to network attacks. Additionally, its high-speed network I/O stack and parallel processing capabilities accelerate application performance.

DNS plays a central role in the migration to IPv6. The Secure64 DNS Cache server includes configurable mechanisms such as DNS64 to facilitate the transition to IPv6 addresses. DNS64 works in conjunction with

NAT64 to translate IPv4 addresses into IPv6 addresses. It applies to requests from an IPv6-only client for a name that does not have an IPv6 address (AAAA record). DNS64 synthesizes an AAAA resource record from the A resource record via an algorithm and a configured IPv6 prefix.

Brocade ServerIron ADX

Brocade ServerIron ADX is a high performance NAT64 gateway that eases the migration to IPv6 by enabling service providers and enterprises to maximize their existing IPv4-based investments while communicating with the growing IPv6-based world, without the need for “rip-and-replace” upgrades. As an IPv6 gateway, the Brocade ADX enables IPv4 networks to interoperate with IPv6 networks via a simple, standards-based Network Address Translation 64 (NAT64) gateway. This capability enables IPv4 clients to communicate with new IPv6 services, as well as new IPv6-based clients to communicate with the traditional IPv4 networks, all without requiring forklift upgrades to existing infrastructure.

In addition to NAT64 gateway capability, Brocade ADX is a low-latency, purpose-built L4-7 application delivery switch that offers a high performance IPv6 SLB (Server Load Balancing) functionality and runs on series of fixed-form and modular-chassis configurable platforms (Brocade ADX 1000 Series - 1 U, Brocade ADX 4000 Series- 4 U, and Brocade ADX 10000 Series - 10 U). Brocade ADX series offers the industry’s leading “capacity on-demand” software license-based upgrade to facilitate the pay-as-you-grow deployment model based on the performance specification needs. Brocade ADX hardware features a multi-chip, multi-core, high-density architecture designed to provide the industry’s highest performance for application delivery.

SOLUTION ARCHITECTURE

Solution Topology 1: In-line

This simple topology is useful for small deployments, or gaining familiarity with the technology in a lab environment.

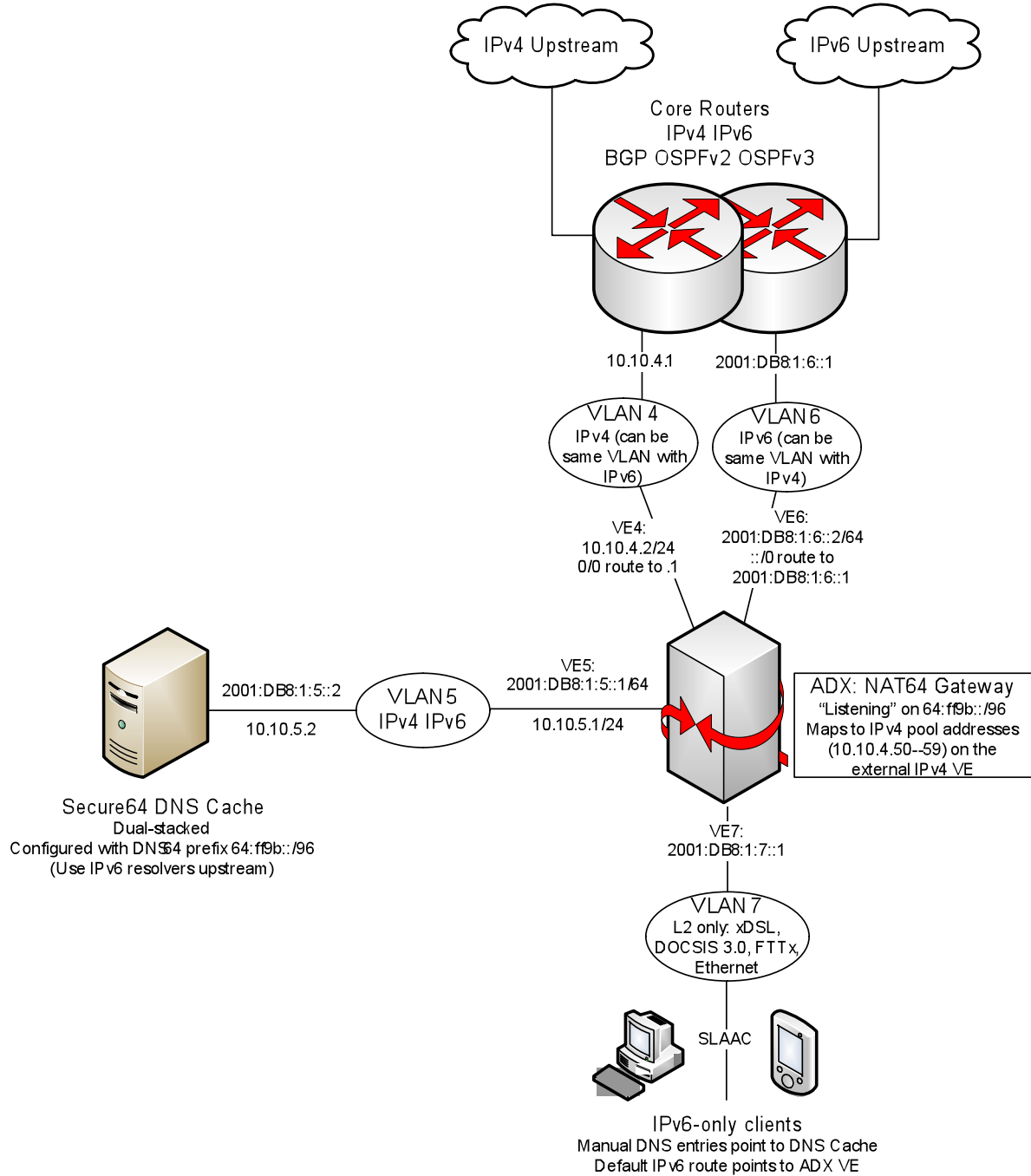


Figure 01. In-line NAT64+DNS64 topology

Solution Topology 2: Routed

This topology is useful in environments where only the translated v4 traffic is intended to traverse through the ADX.

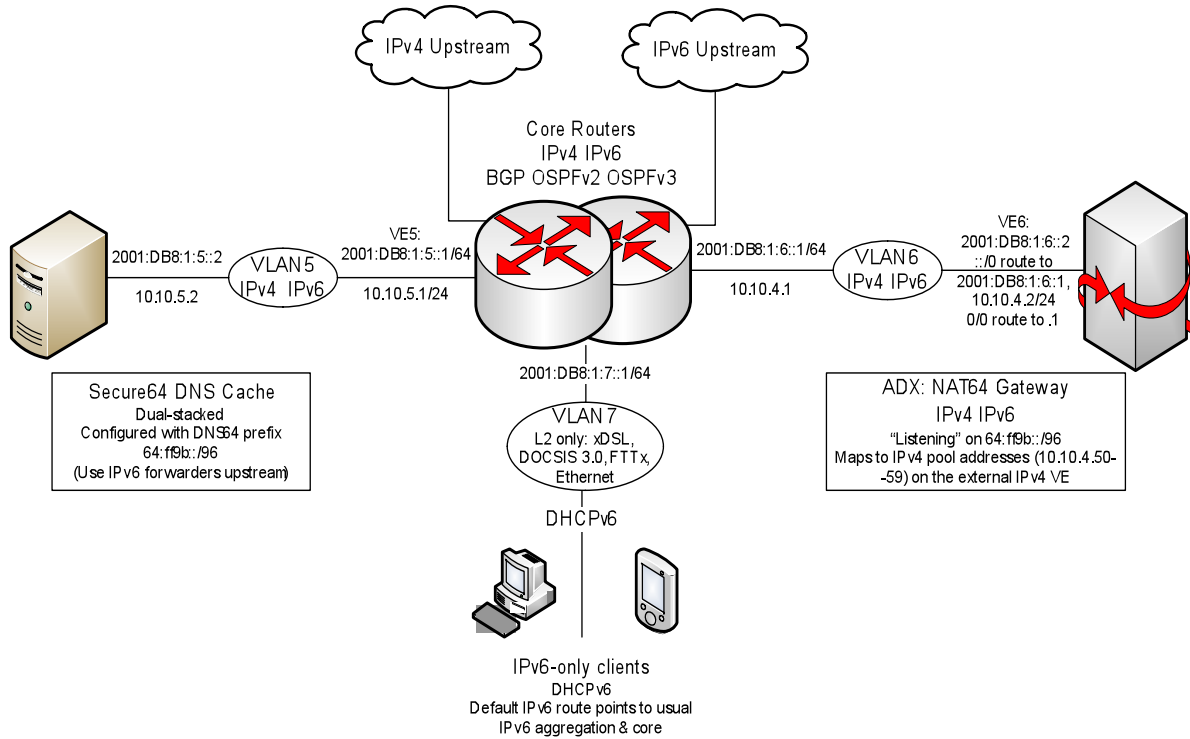


Figure 02. Routed (out-of-critical path) NAT64+DNS64 topology

CONFIGURATION TASKS FOR IN-LINE TOPOLOGY

Configuration for the Brocade ServerIron ADX

1. Attach to the serial console port using the included cable and your favorite terminal program. At the opening CLI prompt, enter:
ServerIron> enable
2. Access the configuration level of the CLI, enter:
ServerIron# configure terminal
ServerIron (config)#
3. Create basic telnet CLI accessibility:
(NOTE: This is not a secure access method, but can be used in labs or internal networks. Please consult the product documentation for configuration of secure access methods such as SSH with AAA and ACLs.)
ServerIron (config)# no enable aaa console
ServerIron (config)# telnet server
ServerIron (config)# username admin password brocade
4. Define the VLANs and network interfaces:
(NOTE: For the sake of convenience, the physical interface numbers used here have been matched with the logical interface numbers and VLANs. This is obviously not a requirement.)
ServerIron (config)# vlan 4 name v4-upstream by port
ServerIron (config)# untagged ethe 4
ServerIron (config)# router-interface ve 4
ServerIron (config)# vlan 5 name DNS by port
ServerIron (config)# untagged ethe 5
ServerIron (config)# router-interface ve 5
ServerIron (config)# vlan 6 name v6-upstream by port
ServerIron (config)# untagged ethe 6
ServerIron (config)# router-interface ve 6
ServerIron (config)# vlan 7 name v6-clients by port
ServerIron (config)# untagged ethe 7
ServerIron (config)# router-interface ve 7
ServerIron (config)# ip route 0.0.0.0 0.0.0.0 10.10.4.1
ServerIron (config)# ipv6 route ::/0 2001:db8:1:6::1
ServerIron (config)# interface ve 4
ServerIron (config)# ip address 10.10.4.2 255.255.255.0
ServerIron (config)# interface ve 5
ServerIron (config)# ipv6 address 2001:db8:1:5::1/64
ServerIron (config)# ipv6 enable
ServerIron (config)# interface ve 6
ServerIron (config)# ipv6 address 2001:db8:1:6::2/64
ServerIron (config)# ipv6 enable
ServerIron (config)# interface ve 7
ServerIron (config)# ipv6 address 2001:db8:1:7::1/64
ServerIron (config)# ipv6 enable
ServerIron (config)# exit
5. Configure the NAT64 IPv6 prefix:
ServerIron (config)# nat64 ipv6-prefix 64:ff9b::/96
6. Configure the NAT64 pool of IPv4 addresses to be used, and tune the session cleanup setting:
ServerIron (config)# nat64 pool test1 10.10.4.50 10.10.4.59 prefix-len 24
ServerIron (config)# server msl 2

7. To exit from the configuration level of the CLI, enter:
 ServerIron (config)# end

8. To save the configuration to NVRAM, enter:
 ServerIron# write memory

NOTE: Be sure to *configure* your dual-stack router as shown in the diagram above.

Configuration for the Secure64 DNS Cache Server

Configuring IP Addresses and Routing for the Secure64 DNS Cache Server

1. Login as a user assigned to the sysadmin role.
2. Enable the sysdnsadmin role at the view prompt:

```
[view@Secure64]#> enable sysadmin
```
3. Define routing and network interfaces:
4.

```
[sysadmin@Secure64]#> route default 10.10.5.1
```
5.

```
[sysadmin@Secure64]#> route default 2001:DB8:1:5::1
```
6.

```
[sysadmin@Secure64]#> route sym
```
7.

```
[sysadmin@Secure64]#> ifconfig eth1 10.10.5.2 255.255.255.0
[sysadmin@Secure64]#> ifconfig eth2 2001:DB8:1:5::2/64
[sysadmin@Secure64]#> activate
[sysadmin@Secure64]#> save
[sysadmin@Secure64]#> show config
```

Editing the Configuration File

It is assumed that the Secure64 DNS Cache server is installed with users configured and assigned to system roles and that a basic server configuration is in place. For detailed user and configuration information, see *Chapter 2 Getting Started* and *Chapter 3 Configuring Secure64 DNS Cache* in the *Secure64 DNS Cache Administrator's Guide*.

To edit the `cache.conf` file using the basic text editor included in Secure64 DNS Cache (designed for small editing tasks):

1. Login as a user assigned to the cachednsadmin role
2. Enable the cachednsadmin role at the view prompt:

```
[view@Secure64]#> enable cachednsadmin
```
3. Open the configuration file in the editor:

```
[cachednsadmin@Secure64]# edit cache.conf
```
4. The editor screen displays 24 lines of the file contents at one time, and it supports 80 characters per line. Use the arrow keys and control commands to navigate the file contents. The keyboard control commands for the editor are listed at the top of the editor screen.
5. Define the interface for listening to queries from clients and provide answers back to clients, the interface for outgoing DNS resolver queries, and the clients allowed to access the DNS server in the `server: section` of the Secure64 DNS Cache `cache.conf` configuration file:

```
interface: 10.10.5.2
interface: 2001:DB8:1:5::2
outgoing-interface: 10.10.5.2
outgoing-interface: 2001:DB8:1:5::2
access-control: 0.0.0.0/0 allow
access-control: ::0/0 allow
```

6. To enable DNS64, add the following information to the server: section of the cache.conf configuration file :
`dns64-prefix: 64:ff9b::/96`
7. Use CTRL-X to save and exit the configuration file on the Secure64 DNS Cache server.
8. Start and stop the Secure64 DNS Cache server to enable the configuration changes:
`[cachednsadmin@Secure64]# stop cachedns`
`[cachednsadmin@Secure64]# start cachedns`

CONFIGURATION TASKS FOR ROUTED TOPOLOGY

Configuration for the Brocade ServerIron ADX

1. Attach to the serial console port using the included cable and your favorite terminal program. At the opening CLI prompt, enter:
ServerIron> enable
2. Access the configuration level of the CLI, enter:
ServerIron# configure terminal
ServerIron (config)#
3. Create basic telnet CLI accessibility:
(NOTE: This is not a secure access method, but can be used in labs or internal networks. Please consult the product documentation for configuration of secure access methods such as SSH with AAA and ACLs.)
ServerIron (config)# no enable aaa console
ServerIron (config)# telnet server
ServerIron (config)# username admin password brocade
4. Define the VLANs and network interfaces:
(NOTE: For the sake of convenience, the physical interface numbers used here have been matched with the logical interface numbers and VLANs. This is obviously not a requirement.)
ServerIron (config)# vlan 6 name gateway64 by port
ServerIron (config)# untagged ethe 6
ServerIron (config)# router-interface ve 6
ServerIron (config)# ip route 0.0.0.0 0.0.0.0 10.10.4.1
ServerIron (config)# ipv6 route ::/0 2001:db8:1:6::1
ServerIron (config)# interface ve 6
ServerIron (config)# ip address 10.10.4.2 255.255.255.0
ServerIron (config)# ipv6 address 2001:db8:1:6::2/64
ServerIron (config)# ipv6 enable
ServerIron (config)# exit
5. Configure the NAT64 IPv6 prefix:
ServerIron (config)# nat64 ipv6-prefix 64:ff9b::/96
6. Configure the NAT64 pool of IPv4 addresses to be used, and tune the session cleanup setting:
ServerIron (config)# nat64 pool test1 10.10.4.50 10.10.4.59 prefix-len 24
ServerIron (config)# server msl 2
7. To exit from the configuration level of the CLI, enter:
ServerIron (config)# end
8. To save the configuration to NVRAM, enter:
ServerIron# write memory

NOTE: Be sure to configure your dual-stack router as shown in the diagram above.

Configuration for the Secure64 DNS Cache Server

All configuration tasks are the same as for the in-line topology.

Follow the instructions in [Configuration for the Secure64 DNS Cache Server](#) on page 9.

ADDITIONAL DNS64 FUNCTIONALITY OPTIONS WITH SECURE64 DNS CACHE

Configuration of distinct IPv6 prefixes with Views

To customize DNS64 behavior for specific clients, use the Secure64 DNS Cache views feature. With views, you can configure different functionality and DNS64 prefixes based on characteristics of the requesting client. The methods used to identify clients for a view are:

- `match-clients: <ip_spec>` Defines the clients to use the view based on the IPv4 or IPv6 source address(es) of the incoming message, and/or
- `match-destinations: <ip_spec>` Defines the clients to use the view based on the destination IPv4 or IPv6 address(es) of the incoming message (the IP address on the Secure64 DNS server to which the query was sent).

Example:

```
view:
  view-name: dns64clients
  # applies only to the clients within the specified IPv6 addresses
  match-clients: 3ffe:801::/32
  dns64-prefix: 2001:db8:122:344::/96
```

Configuration of multiple “sticky” IPv6 prefixes

You can define multiple DNS64 prefixes to support multiple NAT64 devices and spread the load across the devices. Each NAT64 device defines different IPv6 prefixes, which are also defined in the `dns64-prefix:` attributes of the Secure64 DNS Cache `cache.conf` configuration file.

When more than one prefix is defined, Secure64 DNS Cache allocates use of the prefixes based on DNS query source. (Specifically, the system uses the last octet of the IP address of the requesting client and the modulo of that value against the number of prefixes defined to select the prefix used for the client.) As a result, the requesting client uses the same prefix, and the load is spread across the NAT64 devices.

Example:

```
dns64-prefix: 2607:fb90:beef::/96
dns64-prefix: 2001:db8:01a0::/96
dns64-prefix: 1234:db8:5678::/96
```

Using the above examples, assume the client’s IP address is `2001:abcd::1998`. The modulo of $98/3$ is 2, which causes the third `dns64-prefix:` to be selected (since the internal ordering is 0, 1, 2). For the given client, the same `dns64-prefix:` is always selected (assuming the same number of defined prefixes and the same client IP address).

CONCLUSION

The use-case and topologies described here are a small sample of what is possible with NAT64. For additional information on the solutions outlined here, or other NAT64 use-cases:

Brocade Sales:

- sales@brocade.com
- Main site: <http://www.brocade.com>
- US and Canada Toll-Free: 877-272-3232
- International: 408-333-8000
- <http://www.brocade.com/company/contacting-brocade/index.page>

Secure64 Sales:

- sales@secure64.com
- <http://www.secure64.com>
- US and Canada Toll-Free: 877-890-0064
- International: 303-242-5890

Brocade Support:

- support@brocade.com
- Portal: <http://my.brocade.com>
- US and Canada Toll-Free: 800-752-8061
- International: 408-333-6061
- <http://www.brocade.com/services-support/index.page>

Secure64 Support:

- support@secure64.com
- <http://www.secure64.com/support>
- US and Canada Toll-Free: 877-890-0064
- International: 303-242-5890

APPENDIX A: RUNNING CONFIGURATIONS

Example Brocade Configuration for In-Line Topology

```
ServerIronADX 1000(config)#sho run
!Building configuration...
!
ver 12.3.01
!
global-protocol-vlan
!
nat64 ipv6-prefix 64:ff9b::/96
!
nat64 pool test1 10.10.4.50 10.10.4.59 prefix-len 24
!
server msl 2
!
context default
!
!
!
vlan 1 name DEFAULT-VLAN by port
!
vlan 4 name v4-upstream by port
    untagged ethe 4
    router-interface ve 4
!
vlan 5 name DNS by port
    untagged ethe 5
    router-interface ve 5
!
vlan 6 name v6-upstream by port
    untagged ethe 6
    router-interface ve 6
!
vlan 7 name v6-clients by port
    untagged ethe 7
    router-interface ve 7
!
!
no enable aaa console
!
ip route 0.0.0.0 0.0.0.0 10.10.4.1
!
ipv6 route ::/0 2001:db8:1:6::1
!
telnet server
username admin password brocade
!
interface ve 4
    ip address 10.10.4.2 255.255.255.0
!
interface ve 5
    ipv6 address 2001:db8:1:5::1/64
    ipv6 enable
!
interface ve 6
    ipv6 address 2001:db8:1:6::2/64
    ipv6 enable
!
interface ve 7
    ipv6 address 2001:db8:1:7::1/64
    ipv6 enable
```

Example Brocade Configuration for Routed Topology

```

ServerIronADX 1000(config)#sho run
!Building configuration...
!
ver 12.3.01
!
global-protocol-vlan
!
nat64 ipv6-prefix 64:ff9b::/96
!
nat64 pool test1 10.10.4.50 10.10.4.59 prefix-len 24
!
server msl 2
!
context default
!
!
!
vlan 1 name DEFAULT-VLAN by port
!
vlan 6 name gateway64 by port
    untagged ethe 6
    router-interface ve 6
!
!
no enable aaa console
!
ip route 0.0.0.0 0.0.0.0 10.10.4.1
!
ipv6 route ::/0 2001:db8:1:6::1
!
telnet server
username admin password brocade
!
interface ve 6
    ip address 10.10.4.2 255.255.255.0
    ipv6 address 2001:db8:1:6::2/64
    ipv6 enable
!
!

```

Example Secure64 Configuration for Both Topologies

Below are the applicable contents of the cache.conf DNS configuration file. (Other DNS configuration options are available for additional features and functionality. Refer to the *Secure64 DNS Cache Administrator Guide* for details):

```

server:
    #Interfaces to listen to queries from clients and send responses
        interface: 10.10.5.2
        interface: 2001:DB8:1:5::2

    #Interfaces for resolving queries
    #If not defined, server uses all interfaces in random order
        outgoing-interface: 10.10.5.2
        outgoing-interface: 2001:DB8:1:5::2

```

```
#Control clients that can access the DNS server
#0.0.0.0/0 and ::0/0 allows all
    access-control: 0.0.0.0/0 allow
    access-control: ::0/0 allow

#Define the DNS64 prefix
    dns64-prefix: 64:ff9b::/96
```

Below are the system configuration options executed at the sysadmin command line. (Includes system and networking options not discussed here. Other system configuration options are available for additional features and functionality. For details, refer to the *Secure64 DNS Cache Administrator Guide*):

```
[sysadmin@Secure64]# show config
running configuration:
ifconfig eth0 192.168.110.100 255.255.255.0
ifconfig eth1 10.10.5.2 255.255.255.0
ifconfig eth2 2001:DB8:1:5::2/64
ifconfig lo0 192.168.130.110 255.255.255.255
console eth0
route default 10.10.5.1
route default 2001:DB8:1:5::1
route sym
nameserver 0 192.168.127.1 6
nameserver 1 192.168.127.8 4
ntp 192.168.95.65
tz America/Denver
syslog 192.168.95.65
loglevel 4 50
nodename Secure64
No defense rules configured.
ruleactions on
nullchecksums allow
timeout 0
ip frag timeout 5
ip frag high 40
ip frag low 30
```

© 2011 Brocade Communications Systems, Inc. All Rights Reserved. 09/10 GA-SG-397-00 R2

Brocade, the B-wing symbol, DCX, Fabric OS, and SAN Health are registered trademarks, and Brocade Assurance, Brocade NET Health, Brocade One, CloudPlex, MLX, VCS, VDX, and When the Mission Is Critical, the Network Is Brocade are trademarks of Brocade Communications Systems, Inc., in the United States and/or in other countries. Other brands, products, or service names mentioned are or may be trademarks or service marks of their respective owners.

Notice: This document is for informational purposes only and does not set forth any warranty, expressed or implied, concerning any equipment, equipment feature, or service offered or to be offered by Brocade. Brocade reserves the right to make changes to this document at any time, without notice, and assumes no responsibility for its use. This informational document describes features that may not be currently available. Contact a Brocade sales office for information on feature and product availability. Export of technical data contained in this document may require an export license from the United States government.

Secure64®, SourceT®, and Secure64 DNS Cache™ are trademarks or registered trademarks of Secure64 Software Corporation.