Technical Guide

Transitioning IPv4 to IPv6

Feature Overview and Configuration Guide

Introduction

Due to the rapid growth of Internet users, the current IPv4 address system is not sustainable. While the Internet has grown enormously, the approximately 4 billion IPv4 addresses seemed sufficient when the commercial Internet was in its infancy.

Today, we have computers, printers, cell phones, tablets, gaming consoles, smart TVs, and numerous IoT devices like refrigerators, light bulbs, thermostats, and vehicles. Each device requires an IP address.

The fact is that IPv4 simply cannot handle the addressing needs of the modern world. Thankfully, the IETF has created the solution: the IPv6 addressing system. IPv6 uses 128-bit addresses, as opposed to IPv4s 32-bit addressing pool. This means there are approximately 340 trillion, trillion, trillion possible IPv6 addresses.

As IPv4 and IPv6 networks are not directly inter-operable, there are transition mechanisms available that permit hosts on either network type to communicate with any other host. Transition mechanisms bridge between IPv4 and IPv6 and allow the two versions to work side by side. Because IPv6 is a completely separate protocol from IPv4, it can be run in parallel with IPv4 as the transition is made from IPv4 to IPv6. Hosts and network devices can run both IPv4 and IPv6 on the same interface at the same time (dual-stacked), and each is invisible to the other; there is no interference between the two protocols. In time, IPv4 will fade away as IPv6 is often easier and more flexible than IPv4.

This guide looks at the following IPv4 to IPv6 transition mechanisms:

- Dual Stack Lite (DS-Lite)
- Light Weight 4over6
- MAP-E
- NAT64

Contents

Products and software version that apply to this guide

This guide applies to AlliedWare Plus™ router products, running version **5.4.9-0.1** or later.

For more information, see the following documents:

- The product's Datasheet
- The product's Command Reference

These documents are available from the above links on our website at alliedtelesis.com.

List of terms

The following table lists the terms and acronyms used in this guide:

Table 1: List of terms

ACRONYM	DESCRIPTION
AD	Administrative Distance
Address plus port	A method of sharing an IPv4 address between a number of hosts by allocating a specific subset of ports to each host.
AFTR	Address Family Transition Router in the service provider's network (softwire concentrator) DS-Lite implementation. This is an IPv6 tunnel (RFC 2473) endpoint which implements NAT functionality [RFC 6333].
lwAFTR	An AFTR element (Address Family Transition Router element [RFC 6333]), which supports Lightweight 4over6 extension. An lwAFTR is an IPv4-in-IPv6 tunnel endpoint which maintains per-subscriber address binding only and does not perform a NAPT44 function.
B4	Basic Bridging Broadband (softwire initiator) DS-Lite implementation. This is the CPE equipment, that creates an IPv6 tunnel (RFC 2473) to an AFTR at the service provider.
lwB4	A B4 element (Basic Bridging Broadband element [RFC 6333]), which supports Lightweight 4over6 extensions. An lwB4 is a function implemented on a dual-stack capable node, (either a directly connected device or a CPE), that supports port-restricted IPv4 address allocation, implements NAPT44 functionality and creates a tunnel to an lwAFTR.
BMR	Basic Mapping Rule. For each MAP-E tunnel there is a single BMR (but other CPE's may use the same BMR). This is used for IPv4 prefix, address or port set assignment, and for configuring MAP IPv6 address or prefix.
BR	Border Router located within the ISP network.
CGN	Carrier Grade NAT, typically performed by an ISP router.
CPE	Customer Premise Equipment
DHCP	Dynamic Host Configuration Protocol
DNS	Domain Name System
EA bits	Embedded Address bits. A set of bits embedded in the IPv6 address of the CPE that encode information about the IPv4 address/prefix. Can include the entire address or prefix, or the host portion and/or Port Set ID.
EA length	The number of EA bits
Endpoint-Independent Mapping	NAT will use, wherever possible, the source port mapping described in RFC 4787 (sometimes known as IP Masquerade).
FMR	Forwarding Mapping Rule, used to allow direct communication between MAP CPEs (mesh mode).
FQDN	Fully Qualified Domain Name
Introducer	A server on the internet that helps peer-to-peer application users find each other and facilitates the establishing of direct communication between them.

ACRONYM	DESCRIPTION
IPv6 Tunneling	Encapsulation mechanism used by MAP-E (RFC 2473). Also used by DS-Lite and lw4o6.
MAP-E	Mapping of Addresses and Ports with Encapsulation RFC 7597
MAP domain	One or more MAP CPEs and BRs connected to the same virtual link. A service provider may deploy a single MAP domain or may utilize multiple MAP domains.
MAP rule	A set of parameters describing the mapping between an IPv4 prefix, IPv4 address, or shared IPv4 address and an IPv6 prefix or address. Each domain uses a different mapping rule set.
MSS	Maximum Segment Size
RFC	Request for Comments
SNMP	Simple Network Management Protocol
NAT	Network Address Translation
MAP-E	Mapping of Address and Port with Encapsulation
MTU	Maximum Transmit Unit
NAPT	Network Address and Port Translation
NAPT44	IPv4 to IPv4 Network Address and Port Translation. Both internal to external Port and internal to external IPv4 Address translation. The port translation is used to overcome scaling problem with lots of hosts with private IPv4 addresses sharing single public IPv4 address.
PMTU	Path Maximum Transmit Unit
Port Set	A specific set of ports that assigned to a single CPE when the CPEs IPv4 address is shared with other CPEs.
PSID	Port Set ID. A number which identifies a Port Set within a particular Address plus port implementation. The PSID is embedded in all port numbers in the port set.
PSID length	The number of bits used for the PSID.
PSID offset	The offset of the PSID bits in the port number.
Softwire	An umbrella term for a range of methods that provide connections for IPv4 networks across IPv6 networks and IPv6 networks across IPv4 networks.
Upstream interface	The physical interface connected to the ISPs IPv6 network. This interface has a globally scoped IPv6 address assigned by the ISP.

Dual Stack-Lite (DS-Lite)

Introduction

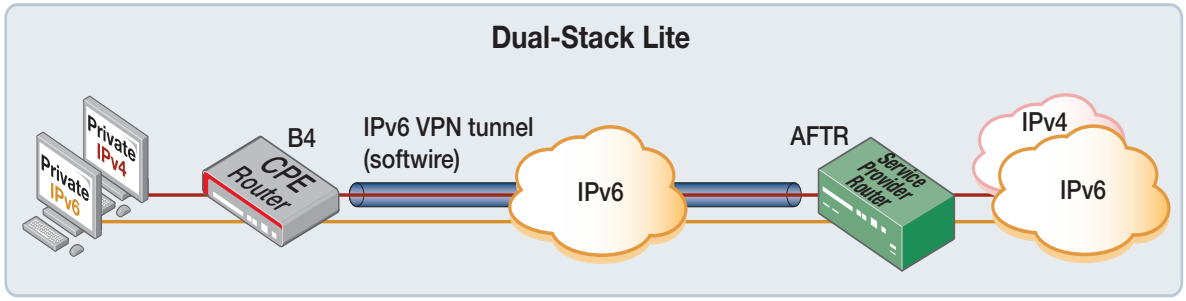
Dual stack networks are one of the many IPv4 to IPv6 transition mechanisms introduced in recent years. In a dual stack network, all nodes are enabled for both IPv4 and IPv6, which is especially important for routers as they are typically the first nodes to receive external traffic.

DS-Lite is an IPv6 transition solution for ISPs with IPv6 infrastructure to connect their IPv4 subscribers to the Internet. It uses IPv4-in-IPv6 tunneling to send a subscriber's IPv4 packet through a tunnel on the IPv6 access network to the ISP.

In simple terms, the DS-Lite architecture consists of:

- Basic Bridging Broadband (B4): a device or component at the CPE which initiates an IPv6 tunnel (encapsulating IPv4 packets within IPv6 packets) between itself and an Address Family Transition Router.
- Address Family Transition Router (AFTR): a device residing at the ISP core which terminates the IPv6 tunnel from the B4 device. In other words, decapsulates the packets and routes them to their IPv4 destination.
- Softwire: The logical IPv4-in-IPv6 tunnel created between the B4 and AFTR.

Figure 1: Dual stack-lite architecture - simple diagram



The following sections describe DS-Lite and how to configure it using AlliedWare Plus.

What is DS-Lite?

DS-Lite allows a service provider to continue support for existing IPv4 services, while also providing incentive for the deployment of IPv6. DS-Lite decouples the deployment of IPv6 in the service provider's network from the rest of the Internet, making the incremental deployment of IPv6 services within the ISP network easier.

Implementing this feature enables customers to connect to broadband services using an IPv6 only connection. Customer IPv4 packets are transported inside IPv6 VPNs and are decapsulated within the service provider's network, and routed to the IPv4 Internet from there.

How does DS-Lite work?

DS-Lite allows islands of IPv4 networks (such as within a subscriber's home), to be connected to the IPv4 Internet via an IPv6 only access line. This line is provided by the service provider.

The IPv4 connections from the subscriber's private network are tunneled inside an IPv6 VPN to the service provider's internal IPv4 network. The service provider then performs IPv4 Carrier Grade NAT, to translate internal IPv4 addresses to use public IPv4 addresses allowing access to the public IPv4 Internet. AlliedWare Plus supports the DS-Lite B4 device behavior (located at the subscriber's home), as described in the DS-Lite RFC 6333.

This allows the AlliedWare Plus router configured with DS-Lite to form an IPv6 VPN (RFC 2473), and inter-operate with ISP equipment providing DS-Lite AFTR device behavior. The IPv6 VPN tunnel destination (the address of the AFTR device within the ISP) can be statically configured with either an IPv6 FQDN or an IPv6 address. Alternatively the address of the AFTR device can be dynamically assigned via DHCPv6 option 64 (OPTION_AFTER_NAME: 64) as per RFC 6334.

IPv6 connections to remote IPv6 networks are transported across the same line seamlessly.

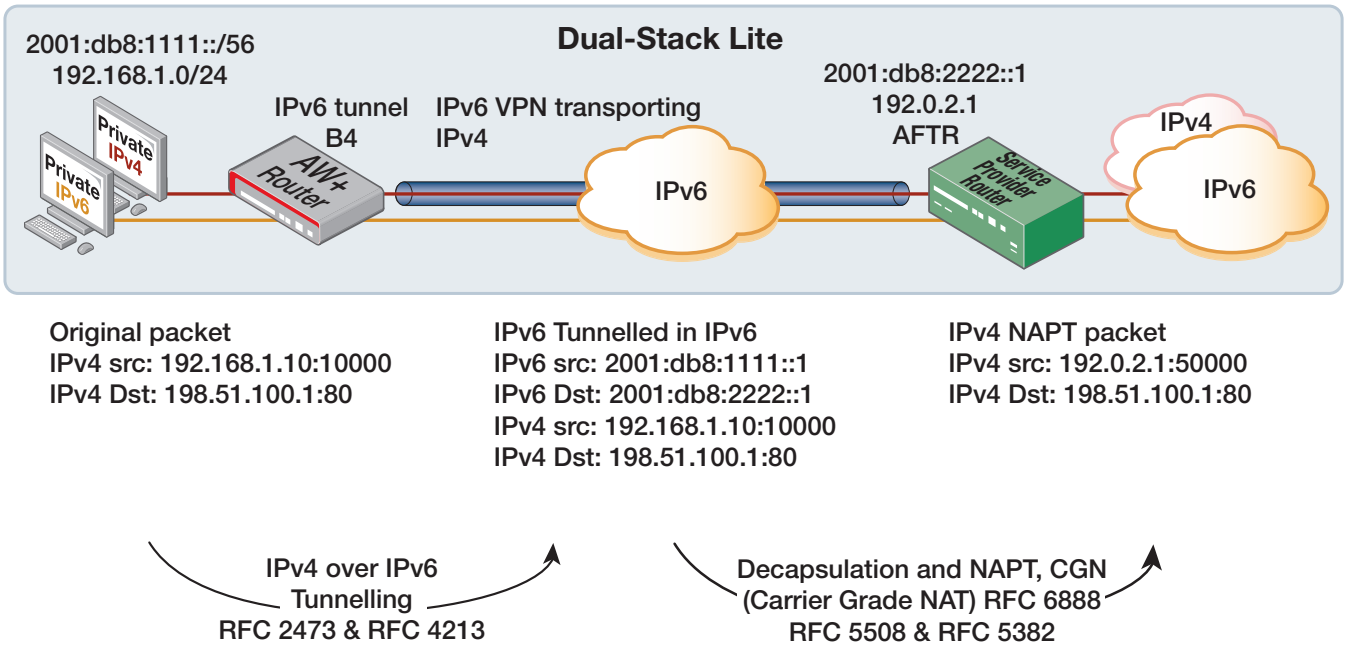
The DS-Lite process

DS-Lite tunnels packets directly to the Address Family Transition Router (AFTR) in the service provider's network. The AFTR is where the service provider allows users to share IPv4 addresses.

1. An IPv6 tunnel is created. The IPv6 tunnel source is the router WAN interface, which has a globally scoped IPv6 WAN address allocated. It is configured via the DS-Lite "upstream interface" command option. The IPv6 tunnel destination used is the IPv6 address of the AFTR device located within the ISP network. Only one DS-Lite tunnel is supported.
2. A connection is established between the router and the AFTR.
3. An IPv4 default route is configured by the network engineer to transport IPv4 traffic via the IPv6 tunnel.
4. All IPv4 traffic whose destination IP address matches the IPv4 (default) route, ingresses an internal virtual tunnel interface (or VTI) whose destination IP address matches the IPv4 default route. It is sent to the service provider encapsulated in an IPv6 RFC2473 tunnel.
5. Within the service provider network, the AFTR device performs Carrier Grade NAT, translating private IPv4 addresses to public IPv4 addresses.

The diagram below is an expansion of Figure 1 on page 6. It includes some example data to help show the DS-Lite process:

Figure 2: Dual Stack-lite example



Configuring DS-Lite

You must turn on the feature and define the upstream interface and static AFTR to create the DS-Lite tunnel.

You need to:

1. Enable DS-Lite, to turn the feature on.
2. Specify the IPv6 address of the AFTR device. This is the IPv6 tunnel destination. It can be either a statically configured host IPv6 address, or an FQDN whose address is determined via IPv6 DNS resolution. Alternatively it can be dynamically assigned via DHCPv6 option 64.
3. Specify the IPv6 DS-Lite tunnel source, (it can be either a VLAN or eth upstream interface). This interface must be configured with a globally scoped IPv6 address.

How to use DS-Lite

1. To turn on the DS-Lite feature, use the commands:

```
awplus# configure terminal
awplus(config)# interface tunnel2
awplus(config-if)# tunnel mode ds-lite
```

   To turn off the DS-Lite feature, use the **no** variant of this command.

2. Specify the IPv6 address of the AFTR device, use the command:

```
awplus(config-if)# tunnel destination <fqdn>
```

   To remove the AFTR name from DS-Lite, use the **no** variant of this command. The Fully Qualified Domain Name (FQDN), or IPv6 address of the AFTR must be known if statically configured.

   Alternatively, the domain name of the AFTR can also be learned **automatically** from a DHCPv6 server.

   - To configure the WAN interface as a DHCPv6 client, use the commands:

```
awplus(config)# interface eth1
awplus(config-if)# ipv6 address dhcp
```

   - To specify the upstream “source” interface of the IPv6 DS-Lite VPN, use the commands:

```
awplus(config)# interface tunnel2
awplus(config-if)# tunnel destination dhcp interface eth1
```

   To remove an upstream interface from DS-Lite, use the **no** variant of this command.

   The upstream interface requires a globally scoped IPv6 address, and it can be configured as a VLAN, eth, or bridge interface. The IPv6 address on this interface serves as the source for the DS-Lite tunnel, encapsulating all IPv4 traffic that matches the default route.

3. To add a route out of the tunnel for all IPv4 traffic, use the following command:

```
awplus(config)# ip route 0.0.0.0/0 tunnel2
```

How to display current DS-Lite configurations

The following show commands display the current DS-Lite configurations and status:

From Global Configuration mode or Privileged Exec mode, the **show interface tunnel** command displays the current state of DS-Lite running on your device:

```
awplus# show interface tunnel2
```

From Privileged Exec mode, the **show running-config interface tunnel** command displays the running system information specific to DS-Lite:

```
awplus# show running-config interface tunnel'x'
```

Configuration examples

The following section provides four configuration examples. The level of complexity increases as they progress from the relatively easy Dynamic AFTR and Static AFTR configuration through to the more involved IPv6 ND Proxy and firewall configurations.

Example1: Dynamic AFTR

The following configuration shows how to configure DS-Lite when the WAN interface is acting as a DHCPv6 client. In this example static AFTR is not configured, and instead the DHCPv6 client automatically requests and learns the AFTR FQDN from the ISP DHCPv6 server.

The command **static-aftr-name <fqdn>** is not required.

Config 1: Dynamic AFTR

```
!
ip name-server <address>
 ip domain-lookup
!
interface eth1
 ipv6 address dhcp
!
interface vlan1
 ip address 192.168.100.1/24
!
interface tunnel2
 tunnel source eth1
 tunnel destination dhcp interface eth1
 tunnel mode ds-lite
 ip address 192.0.0.2/29
!
ip dns forwarding
ip dns forwarding cache size 10 timeout 3600
!
```

Use the **show interface tunnel** command to confirm the dynamic AFTR name was installed correctly via DHCPv6 option 64.

Output 1: Example output from **show interface tunnel** command (DHCPv6 acquired AFTR name)

```
awplus#show interface tunnel2
Interface tunnel2
  Link is UP, administrative state is UP
  Hardware is Tunnel
  IPv4 address 192.0.0.2/29 point-to-point 192.0.0.1
  index 30 metric 1 mtu 1460
  <UP,POINT-TO-POINT,RUNNING,MULTICAST>
  VRF Binding: Not bound
  SNMP link-status traps: Disabled
  Bandwidth 1g
  Tunnel source eth1 (2001:db8::1), destination dhcp (2001:db8:acc3:55::1) AFTR
aftr.com.
  Tunnel name local 2001:db8::1, remote 2001:db8:acc3:55::1
  Tunnel protocol/transport ds-lite, key disabled, sequencing disabled
  Tunnel TTL 64
  Checksumming of packets disabled, path MTU discovery disabled
  Router Advertisement is disabled
  Router Advertisement default routes are accepted
  Router Advertisement prefix info is accepted input packets 0, bytes 0,
  dropped 0,multicast packets 0 output packets 0, bytes 0, multicast packets 0,
  broadcast packets 0 input average rate : 30 seconds 0 bps, 5 minutes 0 bps
  output average rate: 30 seconds 0 bps, 5 minutes 0 bps
  Time since last state change: 0 days 00:01:38
```

Example 2: Static AFTR

The following example shows how to configure a tunnel interface for DS-lite. In this example static AFTR is configured, as it is known.

Config 2: Static AFTR

```
awplus#show run interface tunnel2
interface tunnel2
tunnel source eth1
tunnel destination 2001:db8:acc3:55::1
tunnel mode ds-lite
ip address 192.0.0.2/29
```

Use the **show interface tunnel** command to confirm the static AFTR has been created.

Output 2: Example output for **show interface tunnel** command - static configuration

```
awplus#show interface tunnel2
Interface tunnel2
Link is UP, administrative state is UP
Hardware is Tunnel
IPv4 address 192.0.0.2/29 point-to-point 192.0.0.1
index 14 metric 1 mtu 1460
<UP,POINT-TO-POINT,RUNNING,MULTICAST>
VRF Binding: Not bound
SNMP link-status traps: Disabled
Bandwidth 1g
Tunnel source eth1 (2001::db8::1), destination 2001:db8:acc3:55::1
Tunnel name local 2001:db8::1, remote 2001:db8:acc3:55::1
Tunnel protocol/transport ds-lite, key disabled, sequencing disabled
Tunnel TTL 64
Checksumming of packets disabled, path MTU discovery disabled
Router Advertisement is disabled
Router Advertisement default routes are accepted
Router Advertisement prefix info is accepted
input packets 0, bytes 0, dropped 0, multicast packets 0
output packets 0, bytes 0, multicast packets 0, broadcast packets 0
input average rate : 30 seconds 0 bps, 5 minutes 0 bps
output average rate: 30 seconds 0 bps, 5 minutes 0 bps
Time since last state change: 0 days 00:00:05
```

Example 3: IPv6 ND proxy

The configuration below demonstrates how to set up DS-Lite with IPv6 Neighbor Discovery (ND) Proxy. In this setup, the globally scoped IPv6 address is assigned to the VLAN interface rather than the Ethernet WAN interface. Therefore, the DS-Lite upstream interface is the VLAN, not the Ethernet WAN. DNS services are provided using stateless DHCPv6 and DNS forwarding cache. Additionally, the tunnel destination is dynamically learned via DHCP

Config 3: DS-Lite with IPv6 ND proxy

```
!
ipv6 dhcp pool IPoE-vlan1

dns-server interface vlan1
!
interface eth1
 ipv6 enable
 no ipv6 nd accept-ra-pinfo
 ipv6 nd proxy interface vlan1
!
interface vlan1
 ip address 192.168.10.1/24
 ipv6 address autoconfig eth1
 no ipv6 nd suppress-ra
 ipv6 nd other-config-flag
 ipv6 nd proxy interface eth1
 ipv6 dhcp server IPoE-vlan1
!
ipv6 forwarding
!
interface tunnel2
 tunnel destination dhcp interface vlan1
 tunnel mode ds-lite
 ip address 192.0.0.2/29
!
ip dns forwarding
ip route 0.0.0.0/0 tunnel2
```

Example 4: Firewall

The following configuration shows how to configure DS-Lite in conjunction with the firewall:

Config 4: DS-Lite with Firewall

```
!
zone ipv4
 network all
  ip subnet 0.0.0.0/0
!
zone ngn
 network wan_ipv6
  ipv6 subnet ::/0 interface eth1
  host eth1
   ipv6 address dynamic interface eth1
!
zone private_ipv6
 network lan
  ipv6 subnet ::/0 interface vlan1
  host vlan1
   ipv6 address dynamic interface vlan1
!
application dhcpv6-r
 protocol udp
 dport 546
!
application dhcpv6-s
 protocol udp
 sport 546
!
application icmpv6
 protocol ipv6-icmp
!
firewall
 rule 10 permit any from private_ipv6 to private_ipv6
 rule 20 permit any from private_ipv6 to ngn
 rule 30 permit any from private_ipv6.lan.vlan1 to ngn
 rule 40 permit dhcpv6-s from ngn.wan_ipv6.eth1 to ngn
 rule 50 permit icmpv6 from ngn to private_ipv6.lan.vlan1
 rule 60 permit dhcpv6-r from ngn to ngn.wan_ipv6.eth1
 rule 70 permit any from ipv4.all to ipv4.all
 protect
!
ip domain-lookup
!
ipv6 dhcp pool IPoE-vlan1
 dns-server interface vlan1
!
[continued on next page...]
```

```
interface eth1
 ipv6 enable
 no ipv6 nd accept-ra-pinfo
 ipv6 nd proxy interface vlan1
!
interface vlan1
 ip address 192.168.10.1/24
 ipv6 address autoconfig eth1
 no ipv6 nd suppress-ra
 ipv6 nd other-config-flag
 ipv6 nd proxy interface eth1
 ipv6 dhcp server IPoE-vlan1
!
ipv6 forwarding
!
interface tunnel2
 tunnel destination dhcp interface vlan1
 tunnel mode ds-lite
 ip address 192.0.0.2/29
!
ip dns forwarding
```

To check the configuration, use the **show running-config interface tunnel** command. There are a number of other useful steps and commands available to confirm the configuration is correct. These steps are shown in bold:

Output 3: Useful **steps** and **show** commands

```
Check the DS-Lite specific configuration:
awplus#show running-config interface tunnel2
 tunnel destination <fqdn>
 tunnel mode ds-lite
 ip address 192.0.0.2/29
!

Check that the IPv4 tunnel interface is up once DS-Lite is ready:
awplus#show ip int brief
Interface             IP-Address         Status          Protocol
eth1                  unassigned         admin up        running
eth2                  unassigned         admin up        down
lo                    unassigned         admin up        running
vlan1                 192.168.10.1/24    admin up        running
tunnel2               192.0.0.2/29       admin up        running

Check the state of the dynamically created DS-Lite tunnel:
awplus#show int tunnel2
Interface tunnel2
  Link is UP, administrative state is UP
  Hardware is Tunnel
  IPv4 address 192.0.0.2/29 point-to-point 192.0.0.7
  index 14 metric 1 mtu 1460
  <UP,POINT-TO-POINT,RUNNING,MULTICAST>
  VRF Binding: Not bound
  SNMP link-status traps: Disabled
  Tunnel source vlan1 (2409:10:2160:c00:200:cdff:fe38:ad), destination <fqdn>
```

```
2001:db8::1
  Tunnel name local 2001:db8::2, remote 2001:db8::1
  Tunnel protocol/transport ipv6, key disabled, sequencing disabled
  Tunnel TTL 64
  Checksumming of packets disabled, path MTU discovery disabled
  Router Advertisement is disabled
  Router Advertisement default routes are accepted
  Router Advertisement prefix info is accepted
    input packets 0, bytes 0, dropped 0, multicast packets 0
    output packets 0, bytes 0, multicast packets 0, broadcast packets 0
    input average rate : 30 seconds 0 bps, 5 minutes 0 bps
    output average rate: 30 seconds 0 bps, 5 minutes 0 bps
  Time since last state change: 0 days 10:41:59
```

Use 'show ip route' to view the dynamically created default route via the tunnel. (Note that if other static default routes via other interfaces need to be configured, you can adjust the administrative distance of the dynamically created IPv4 default route via the DS-Lite VPN.)

```
awplus#show ip route
Codes: C - connected, S - static, R - RIP, B - BGP
       O - OSPF, D - DHCP, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       * - candidate default

Gateway of last resort is 0.0.0.0 to network 0.0.0.0

S*      0.0.0.0/0 [1/0] is directly connected, tunnel2 <==The configured default
route
C       192.0.0.0/29 is directly connected, tunnel2
C       192.168.10.0/24 is directly connected, vlan1
```

Perform a ping test of IPv4 data transported within the DS-Lite tunnel:

```
awplus#ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=57 time=57.1 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=57 time=54.0 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=57 time=54.5 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=57 time=53.8 ms
64 bytes from 8.8.8.8: icmp_seq=5 ttl=57 time=53.9 ms
— 8.8.8.8 ping statistics —
5 packets transmitted, 5 received, 0% packet loss, time 4003ms
rtt min/avg/max/mdev = 53.817/54.705/57.113/1.254 ms
```

Check current firewall rules and entities (configured with dynamic IPv6 self entities:

```
awplus#show firewall rule

[* = Rule is not valid - see "show firewall rule config-check"]
  ID     Action  App               From             To               Hits
------------------------------------------------------------------------------
  10     permit  any               private_ipv6     private_ipv6     2
  20     permit  any               private_ipv6     ngn              2
  30     permit  any               private_ipv6.lan.vlan1
                                                    ngn              0
  40     permit  dhcpv6-s          ngn.wan_ipv6.eth1
                                                    ngn              1
  50     permit  icmpv6            ngn              private_ipv6.lan.vlan1
                                                                     0
  60     permit  dhcpv6-r          ngn              ngn.wan_ipv6.eth1
                                                                     1
  70     permit  any               ipv4.all         ipv4.all         1
```

```
awplus#show entity
Zone:         ipv4
 Network:     ipv4.all
  Subnet:     0.0.0.0/0

Zone:         ngn
 Network:     ngn.wan_ipv6
  Subnet:     ::/0 via eth1
  Host:       ngn.wan_ipv6.eth1
   Address:   fe80::200:cdff:fe38:ab (dynamic) <==Link Local

Zone:         private_ipv6
 Network:     private_ipv6.lan
  Subnet:     ::/0 via vlan1
  Host:       private_ipv6.lan.vlan1
   Address:   2001:db8::2 (dynamic)
   Address:   fe80::200:cdff:fe38:ad (dynamic) <==Link Local
```

The tunnel interface is explicitly configured with 192.0.0.2. Therefore IPv4 traffic originating from the tunnel(e.g. ping for testing connectivity) uses the IPv4 source IP of 192.0.0.2. This ensures that IPv4 traffic originating from the tunnel interface uses an appropriate router B4 device source IP address within the RFC defined IPv4 address range. This specific source IPv4 address is used because in RFC 6333 DS-Lite directs IANA to reserve 192.0.0.0/29 for the Basic Bridging BroadBand (B4) element:

Note that the ISP 'AFTR' device (located within ISP network) will be performing its own Carrier Grade NAT before traffic is forwarded to the target IPv4 server address located in the Internet.

```
awplus#show ip int tunnel2
Interface            IP-Address         Status          Protocol
tunnel2              192.0.0.2/29       admin up        running
```

Check the state of all interfaces:

```
awplus#show ip int brief
Interface             IP-Address         Status          Protocol
eth1                  unassigned         admin up        running
eth2                  unassigned         admin up        down
lo                    unassigned         admin up        running
vlan1                 192.168.10.1/24    admin up        running
tunnel2               192.0.0.2/29       admin up        running
```

Lightweight 4over6

Introduction

Lightweight 4over6 (lw4o6) is a method of tunneling IPv4 packets over an IPv6 network. It is an IPv6 transition technology that provides a way for ISPs that operate over a pure IPv6 network to continue to offer IPv4 Internet services to customers. Lw4o6 is defined in RFC 7596, which in turn refers to a wide range of other related RFC's. The tunneling mechanism is defined in RFC 2473 (Generic Packet Tunneling in IPv6) and is common to a number of other IPv6 transition technologies.

- Lw4o6 is an extension to the Dual-Stack Lite (DS-Lite, RFC 6333) architecture.
- Lw4o6 moves the Network Address and Port Translation (NAPT44) function from the service provider to the CPE.

  This improves the scalability of the translation infrastructure as it removes the requirement for a Carrier Grade NAT function in the tunnel concentrator and reduces the amount of centralized state that must be held to a per-subscriber level. In order to delegate the NAPT44 function and make IPv4 address sharing possible, port-restricted IPv4 addresses are allocated to the CPEs.

Important concepts

The terms B4 and AFTR have already been described in previous sections of this document. However, it helps to see them again and how they relate to the lw406 technology:

- **B4**: Basic Bridging Broadband, as defined in RFC 6333. This is the CPE equipment, that creates an IPv6 tunnel to an AFTR at the service provider.
- **AFTR**: Address Family Transition Router, as defined in RFC 6333. This is an IPv6 tunnel endpoint which implements NAT functionality. This endpoint is located at the ISP.
- **lwB4**: A B4 that supports port-restricted IPv4 address allocation, and implements NAPT44 functionality.
- **lwAFTR**: An AFTR that maintains per-subscriber address binding only and does not perform a NAPT44 function.
- **Softwire**: An umbrella term for a range of methods that provide connections for IPv4 networks across IPv6 networks and IPv6 networks across IPv4 networks

How it works

The three main components in the Lightweight 4over6 architecture are the:

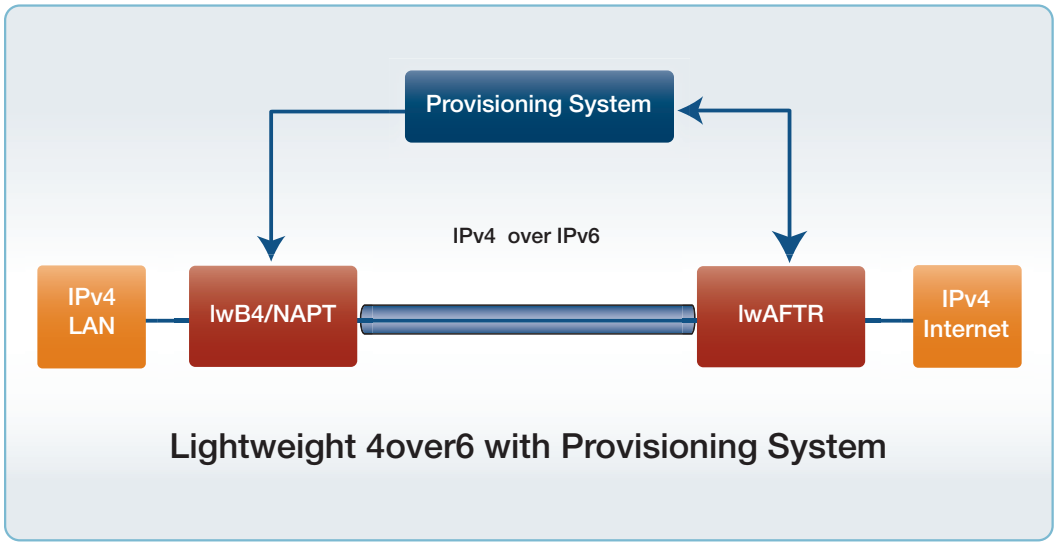
- lwB4, which performs the NAPT function and encapsulation/decapsulation IPv4/IPv6.
- lwAFTR, which performs the encapsulation/decapsulation IPv4/IPv6.
- provisioning system, which tells the lwB4 which IPv4 address and port set to use.

LwB4 differs from a regular B4 in that it now performs the NAPT functionality. This means that it needs to be provisioned with the public IPv4 address and port set it is allowed to use. The ISP provides this information through a provisioning mechanism such as DHCP

LwAFTR needs to know the binding between the IPv6 address of each subscriber and the IPv4 address and port set allocated to that subscriber. This information is used to perform ingress filtering upstream and encapsulation downstream. Note that this is per-subscriber state as opposed to per-flow state in the regular AFTR case.

The consequence of this architecture is that the information maintained by the provisioning mechanism and the one maintained by the lwAFTR MUST be synchronized.

Figure 3: lw4o6 - Provisioning system



The CPE (lwB4) retrieves its configuration parameters from the provisioning system.

The configuration parameters are:

- lwAFTR IPv6 address (tunnel destination address)
- IPv4 external public address for NAPT44
- Port parameters to determine the restricted port set for NAPT44 use
- IPv6 binding prefix

The IPv6 tunnel source address can then be derived from these configuration parameters:

- The CPE receives an IPv4 packet from the LAN side and performs NAPT44 functionality to the packet, using the IPv4 public address and a port number from its assigned port set.
- The NATed packet is encapsulated with an IPv6 header and transmitted via the IPv6 tunnel to the service provider lwAFTR.
- When the CPE receives an IPv6 packet from the lwAFTR at its IPv6 tunnel endpoint, it decapsulates the packet, then forwards it to its IPv4 LAN destination.

lwB4 provisioning using DHCPv6

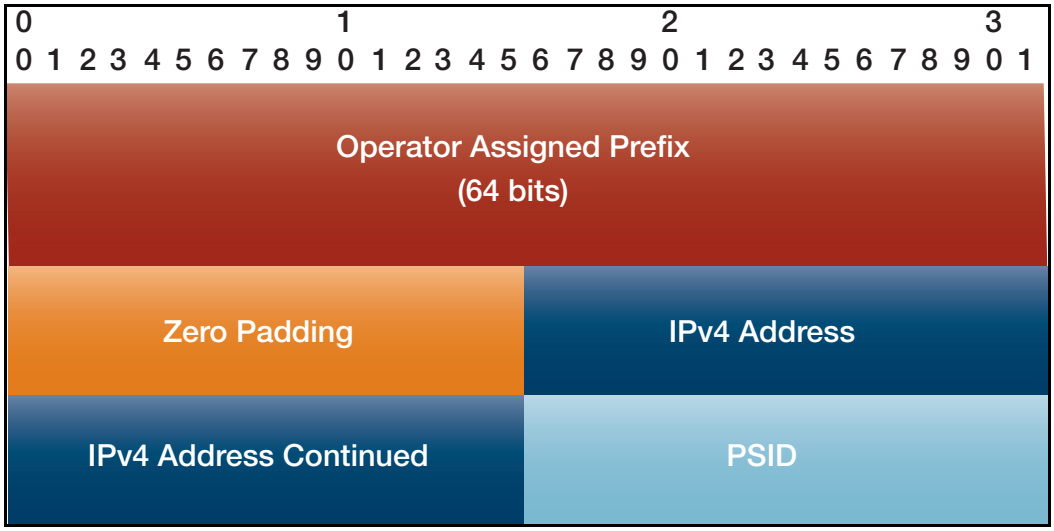
lw4o6 follows the provisioning mechanism as described in RFC 3315 and RFC 7598. The DHCPv6 Options used for lw4o6 are:

- OPTION_S46_CONT_LW: Lightweight 4o6 Container option. This is a container that encapsulates all the options needed for lw4o6, namely, OPTION_S46_BR, OPTION_S46_V4V6BIND, and OPTION_S46_PORTPARAMS. There can be more than one OPTION_S46_BR and at most one OPTION_S46_V4V6BIND.
- OPTION_S46_BR: specifies the lwAFTR IPv6 address.
- OPTION_S46_V4V6BIND: specifies the IPv4 public address for NAPT44, IPv6 binding prefix length, and IPv6 binding prefix.
- OPTION_S46_PORTPARAMS: specifies the PSID offset, PSID length, and PSID.

THE CPE DHCPv6 client always includes the OPTION_S46_CONT_LW option under the Option Request option (ORO) in all the SOLICIT, REQUEST, RENEW, REBIND, and INFORMATION-REQUEST messages. In response, the DHCP server includes inside their messages the OPTION_S46_CONT_LW options parameters.

Deriving the IPv6 tunnel source address

After the CPE is provisioned with its configuration parameters, it performs a longest-prefix match between the IPv6 binding prefix and its active IPv6 prefixes. The result forms the subnet to be used for deriving the IPv6 tunnel source address.



- **Operator Assigned Prefix** - provisioned IPv6 prefix. If the length is less than 64 bits, it is right-padded with zeros to 64 bits.
- **Zero Padding** - all zeros padding.
- **IPv4 Address** - provisioned IPv4 public address.
- **PSID** - Port Set ID, the value is left-padded with zeros to 16 bits.

Conditions:

- When the CPE's IPv6 tunnel source address is changed, it re-initiates its dynamic provisioning process.
- When the CPE's IPv4 public address and PSID is changed, its NAT table is flushed.

Deriving the restricted port set

The restricted port set is derived using the port-set algorithm as described in Section 5.1 of RFC 7597. This is also described in "Port mapping algorithm and port set" on page 30. The section below uses terms that are explained here.

lw4o6-specific requirements:

- The number of a-bits SHOULD be 0, thus allocating a single contiguous port set to each CPE.
- PSIDs containing the system ports (0-1023) should not be allocated to CPEs.

Handling ICMP errors

In general, lw4o6 handles ICMPv6 as described in RFC 2473.

For inbound ICMPv6 error message (Type 1, Code 5: Destination Unreachable, source address failed ingress/egress policy) originating from the lwAFTR, this means:

- that no matching entry in the lwAFTR's binding table has been found, so the IPv4 payload is not being forwarded by the lwAFTR.
- the CPE re-initiates the dynamic port-restricted provisioning process.
- for VALIDATION: The source address of the ICMPv6 error message must match the lwAFTR. If it doesn't, the packet will be discarded.

How to use LW4o6

The router must be connected to an ISP's IPv6 network that is set up for lw4o6. The LAN interfaces of the router must be configured as an IPv4 network. A tunnel must be configured and put into lw4o6 mode. Typically a default route would be configured via the tunnel. The softwire settings for the tunnel need to be specified, including:

- which interface is connected to the lw4o6 network.
- the method by which the rule is to be acquired (DHCP is the only supported option for lw4o6).

Once the lwAFTR address and rule has been acquired, AlliedWare Plus automatically configures the IPv4 address on the tunnel and a NAT rule to translate all the traffic going through it.

With these settings in place, traffic routed through the tunnel will now be translated and encapsulated. The reverse process will happen with traffic received over the tunnel.

Configuration

Configuration example using Prefix Delegation. DHCPv6 must be enabled on the upstream interface, either PD or Stateful DHCPv6:

Config 5: Example using prefix delegation

```
!
softwire-configuration swconfig
 method dhcp
 upstream-interface eth1
!
interface eth1
 ipv6 address ipv6_pd ::1/64
 ipv6 dhcp client pd ipv6_pd
!
interface vlan1
 ip address 192.168.2.1/24
!
interface tunnel2
 tunnel softwire swconfig
 tunnel mode lw4o6
!
ip route 0.0.0.0/0 tunnel2
```

Monitoring lw4o6

The key commands to monitor lw4o6 are:

- show interface tunnel
- show softwire-configuration
- show ip interface
- show ipv6 interface

The example key command outputs below use the static configuration above:

```
awplus#show interface tunnel2
Interface tunnel0
  Link is UP, administrative state is UP
  Hardware is Tunnel
  IPv4 address 192.0.2.23/32 point-to-point 192.0.2.23
  index 19 metric 1 mtu 1460
  <UP,POINT-TO-POINT,RUNNING,MULTICAST>
  VRF Binding: Not bound
  SNMP link-status traps: Disabled
  Bandwidth 1g
  Tunnel protocol/transport lw4o6, key disabled, sequencing disabled
  Tunnel TTL 64
  Checksumming of packets disabled, path MTU discovery disabled
  Router Advertisement is disabled
  Router Advertisement default routes are accepted
  Router Advertisement prefix info is accepted
    input packets 0, bytes 0, dropped 0, multicast packets 0
    output packets 0, bytes 0, multicast packets 0, broadcast packets 0
    input average rate : 30 seconds 0 bps, 5 minutes 0 bps
    output average rate: 30 seconds 0 bps, 5 minutes 0 bps
  Time since last state change: 0 days 00:00:25
```

```
awplus#show softwire-configuration
Softwire Configuration: lw4o6

 Configuration Source: dhcp
 Upstream Interface: eth1
 MAP-E Version: rfc
 lwAFTR Address: 2001:0db8:acc3:0055:0000:0000:0000:0001
 lw4o6 Rule:
  IPv4-Address: 192.0.2.23
  IPv6-Prefix: 2001:0db8::/32
   PSID offset: 0
   PSID length: 16
   PSID: 129 (0x81)

 Border Relay Device: Not Set
```

```
awplus#show ip interface
Interface          IP-Address         Status          Protocol
eth1               unassigned         admin up        running
eth2               unassigned         admin up        down
lo                 unassigned         admin up        running
vlan1              192.168.2.1/24     admin up        running
tunnel2            192.0.2.23/24      admin up        running <--IPv4 address auto generated
```

```
awplus#show ipv6 interface
  * = Autoconfigured Address
Interface     IPv6-Address                      State        Status      Protocol
eth1          2001:db8:1:1781::1111/64          preferred    admin up    running
              2001:db8:1:1781:0:c000:217:81/64  preferred     <--This is the lwB4 IPv6 address
              fe80::200:cdff:fe38:93/64         preferred
eth2          unassigned                        N/A          admin up    down
lo            unassigned                        N/A          admin up    running
vlan1         unassigned                        N/A          admin up    running
tunnel0       unassigned                        N/A          admin up    running
```

If the softwire configuration referenced by the tunnel is incomplete or absent, the following output for **show int tunnel2** with be displayed:

```
awplus#show int tunnel2
Interface tunnel2
  Link is DOWN, administrative state is UP
  Hardware is Tunnel
  Tunnel interface is inactive:
    Specified softwire configuration is incomplete
```

If the softwire configuration has not been reference from the tunnel configuration, the following output for **show int tunnel2** with be displayed:

```
awplus#show int tunnel2
Interface tunnel2
  Link is DOWN, administrative state is UP
  Hardware is Tunnel
  Tunnel interface is inactive:
    Tunnel is not yet fully configured
```

MAP-E

Introduction

There are a number of ways to deal with IPv4 exhaustion and IPv6 transition. One method is called Mapping of Address and Port (MAP). The real advantage with MAP is that it's stateless and doesn't require additional hardware as traffic grows.

There are two modes of MAP, they are MAP-E and MAP-T. The difference between the two options is evident in their names:

- In MAP-**E**, IPv4 traffic is **encapsulated** into IPv6 using a v6 header before it is sent over the v6 network. At the network operator's boundary router, the IPv6 header is then stripped, and the IPv4 traffic is forwarded to the IPv4 Internet.
- In MAP-**T**, the IPv4 packet header is **translated** or mapped to the IPv6 header and back.

Feature overview

MAP-E is a method of tunneling IPv4 packets over an IPv6 network. It is an IPv6 transition technology that provides a way for ISPs that operate over a pure IPv6 network to continue to offer IPv4 Internet services to customers. MAP-E is defined in RFC 7597, which in turn refers to a wide range of other related RFC's. The tunneling mechanism is defined in RFC 2473 (Generic Packet Tunneling in IPv6) and is common to a number of other IPv6 transition technologies.

Many ISPs have migrated from IPv4 to IPv6 networks. However, many customers are still using IPv4 facilities. IPv6 transition technologies, such as MAP-E, allow them to continue to provide IPv4 Internet services to customers over their IPv6 networks.

MAP-E consists of the following methods:

- an encapsulation method RFC 2473 (IPv6 Tunneling) to transport IPv4 packets over an IPv6 network.
- a mechanism for mapping between an IPv4 prefix or IPv4 address or IPv4 shared address and an IPv6 address
- rules that inform a CPE of its IPv4 addressing and how to reach other IPv4 prefixes and the wider Internet.

The key components of a MAP-E network are:

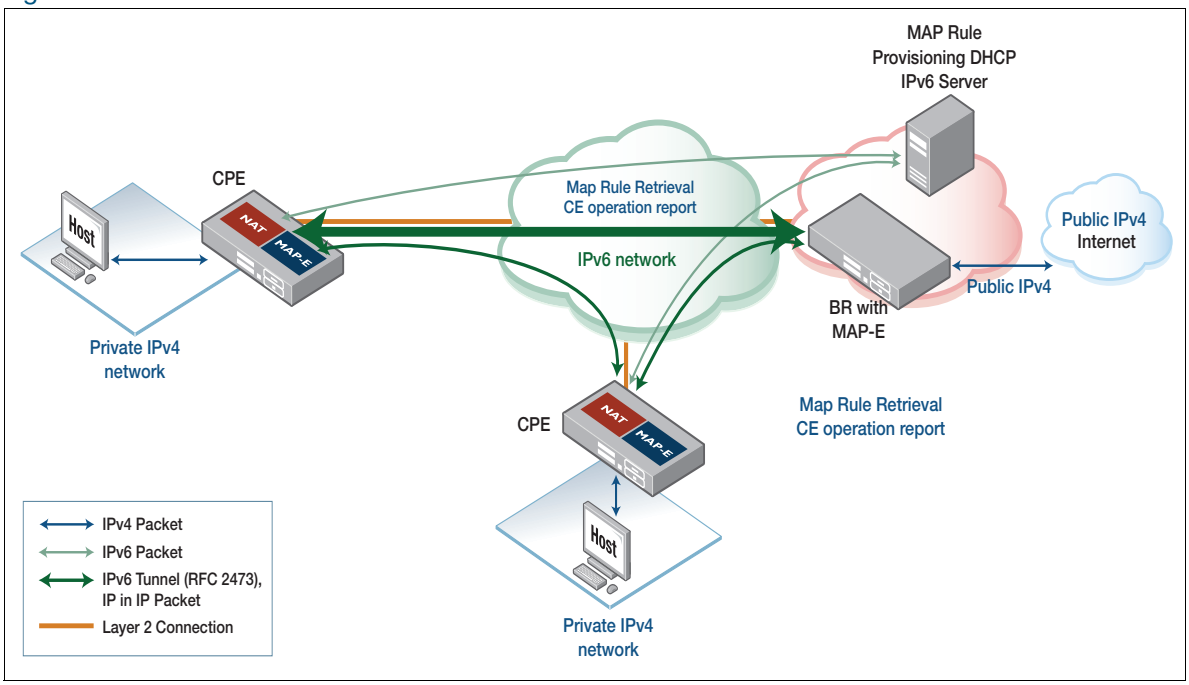
- the underlying IPv6 network
- the Customer Premise Equipment (CPE) situated at the boundary of the customers IPv4 network and ISP's IPv6 network.
- the Boarder Router (BR) situated at the boundary of the ISP's IPv6 network and the entry point to the IPv4 Internet.
- a server for provisioning the MAP rules.

AlliedWare Plus implements the CPE component only. However, it does allow manual configuration of the MAP rules, meaning a rule provisioning server is not required.

How does MAP-E work?

The diagram below shows a MAP-E network with several IPv4 customers networks connected to a service.

Figure 4: MAP-E network



The operation is as follows:

- The CPE obtains an IPv6 prefix from the underlying IPv6 network using one of the usual mechanisms.
- The CPE obtains MAP rules from the Rule Provisioning server via special DHCPv6 options as follows:
  - **OPTION_S46_CONT_MAPE**: MAP-E Container option. This is a container that encapsulates all the options needed for MAP-E, namely, OPTION_S46_BR, OPTION_S46_RULE, and OPTION_S46_PORTPARAMS. There can be at least one OPTION_S46_BR and at least one OPTION_S46_RULE.

- **OPTION_S46_BR**: specifies the BR IPv6 address.
- **OPTION_S46_RULE**: specifies the flags (indicate FMR or BMR), EA length, IPv4 prefix length, IPv4 prefix, IPv6 prefix length, and IPv6 prefix
- **OPTION_S46_PORTPARAMS**: specifies the PSID offset, PSID length, and PSID.

- From the information in the MAP rules, the CPE derives:
  - Its own IPv4 prefix and length. Traffic from the LAN will be NATed to this address.
  - If the network uses IPv4 address sharing, the PSID which defines the set of ports the CPE is permitted to use with that IP address.
  - The IPv6 address of the BR.
  - Information about how to reach other IPv4 prefixes that are present within the Map Domain.

Note that in some cases the MAP rules are obtained through some method that is proprietary to the ISP, such as via a special web server.

At this point the CPE is fully configured. When a host on the private IPv4 network wants to send a packet (e.g. when a user wants to browse a web page on the Internet) the host will send the packet to the CPE as it's gateway device.

- The CPE will consult it's routing table finding a route over the MAP-E tunnel (assuming the Internet connection is being provided through the MAP-E tunnel).
- The source address of the packet will be NATed to the IPv4 address assigned to the CPE and, in the case of IP address sharing, the source port will also be translated to an available port within the assigned set of ports.

The MAP rules are now consulted to find the correct tunnel end point to use, based on the IPv4 destination address.

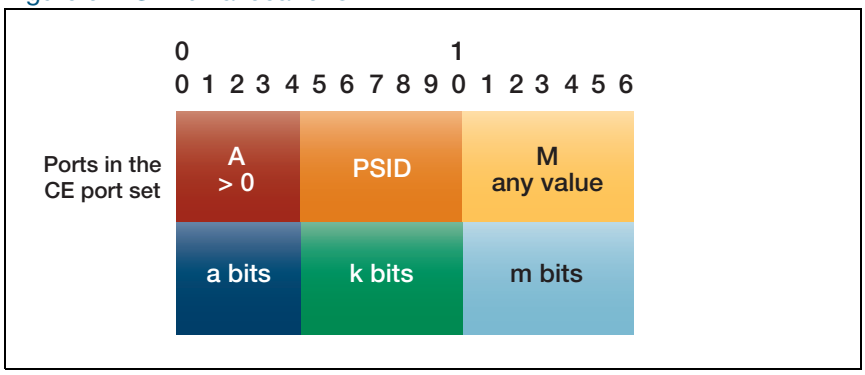
- In the case of a packet destined for a host on the Internet, there will be no matching Forwarding Mapping Rule (FMR) so the BR will be selected as the tunnel endpoint.
- In the case of a packet destined to a host in the same MAP Domain, the associated Forwarding Mapping Rule (FMR) will be found. This will be used to derive the correct tunnel end point corresponding to another CPE.
- The IPv4 packet is now encapsulated in an IPv6 header, where the source address is the IPv6 address assigned to the CPE and destination address is the appropriate tunnel end point.
- When the CPE receives IPv6 packets encapsulating an IPv4 packet it validates the source address to confirm it corresponds to an appropriate MAP rule. The IPv4 packet is then decapsulated. The destination IPv4 should match the CPE's IPv4 address and the destination port should match the CPE's assigned port set, in the case of IP sharing. The destination address and port are translated to the corresponding private values and the packet is routed through to the private network.

Port mapping algorithm and port set

MAP-E uses shared IPv4 addresses (address + port) for stateless mapping of IPv4 addresses into IPv6 addresses, and the 16-bit port number is logically split into three parts as shown in Figure 5.

- **A** selects the range of the port number. If **a bits** is greater than zero, then A must also be greater than zero.
- The **a bits** are the PSID offset and used to reserve the system ports. In RFC 7597, the default length is 6, representing the port range 0-1023 (calculated as 2^(16-6)-1) is reserved.
- The **PSID** should uniquely identify the range of port sets for a CE.
- **k bits** is the length of the PSID field in bits. This is derived by the EA bits minus the suffix length of the IPv4 address in the Basic Mapping Rule.
- **M** represents the contiguous ports.
- The **m bits** represents the contiguous ports in a port set.

Figure 5: PSID bit allocations



The example below shows how the mapping algorithm works:

- The PSID offset has a value of a = 6 bits
- IPv4 address suffix length is 8 (IPv4 address length (32) - prefix length (24))
- The PSID has a length of k = 8 bits (k bits = EA bits 16 - the IPv4 address suffix length 8)
- Each IPv4 Global address is shared between 256 customers (sharing ratio)
- The M length is m = 2 bits (m bits = 16 bits of port number - 6 bits of PSID offset - 8 bits of PSID)
- There are 63 port sets and each set can have 4 ports. In total there are 252 ports CE.

```
Shared Ipv4 address: 192.168.1.0/24
EA bits: 16
PSID: 94 (0x5E, 01011110 in binary)
PSID offset: 6

Port sets: 1400-1403, 2424-2427, 3448-3451, ....., 64888-64891
port   a bits    PSID       m bits
1400   000001    01011110   00
1401   000001    01011110   01
1402   000001    01011110   10
1403   000001    01011110   11
2424   000010    01011110   00
2425   000010    01011110   01
2426   000010    01011110   10
2427   000010    01011110   11

.....
64888  111111    01011110   00
64889  111111    01011110   01
64890  111111    01011110   10
64891  111111    01011110   11
```

Using MAP-E

To use MAP-E, the router must be connected to an ISP's IPv6 network that is set up for MAP-E. The LAN interfaces of the router must be configured as an IPv4 network. You also need to configure a tunnel and put it into MAP-E mode. Typically, a default route is configured via the tunnel. The softwire settings for the tunnel need to be specified, including:

- which version of MAP-E is being used
- which interface is connected to the MAP-E network
- the method by which the MAP rules are to be acquired.

Once the MAP rules have been acquired AlliedWare Plus automatically configures the IPv4 prefix on the tunnel and a NAT rule to translate all the traffic going through it.

It is possible to manually configure the MAP rules in the softwire settings. However, this is primarily intended for testing environments. It should not be attempted in a live network unless very specific direction has been supplied by the ISP.

With these settings in place, traffic routed through the tunnel will now be translated and encapsulated. The reverse process will happen with traffic received over the tunnel.

Communication between hosts in the MAP domain

In addition to providing access to the wider IPv4 Internet, MAP-E supports direct communication amongst hosts in a MAP domain via the forwarding mapping rules. This is known as **mesh-mode**. MAP-E is generally not suitable for server hosting within the MAP domain, but it is suitable for peer to peer communications where exchanges are mediated via an **introducer**.

One special case in mesh-mode is where the network employs IP address sharing using the Address plus Port architecture (RFC 6346). This creates the possibility of peer to peer communication between two hosts each using the same IP address (but with a distinct set of ports). In ordinary routing this would be problematic but the AlliedWare Plus MAP-E implementation has special provision to support this use-case.

By default, mesh-mode between CPE's with different IPv4 addresses will work with a suitable rule in place. If the network employs address sharing the optional command **mesh-mode** must be added to the softwire configuration to support mesh-mode between CPE's that share the same address. This command is optional as it adds a small processing overhead. Therefore it is recommended it only be configured if IP address sharing is in use.

Without this command all traffic that is sent to the shared IP address will be processed locally.

The mesh-mode command syntax is as follows:

```
awplus# configure terminal
awplus(config)# softwire-configuration demo
awplus(config-softwire)# mesh-mode
```

Configuration examples

Basic configuration with MAP rule provisioning via DHCP

Below is a MAP-E configuration where the MAP rules are received via DHCP. The MAP-E tunnel is used to communicate with the ISP and Internet. An IPv4 address will automatically be assigned to the tunnel based on the MAP rules and NAT masquerading will automatically be configured to translate IPv4 egressing via the tunnel. An IPv6 address will automatically be generated on eth1 via a router advertisement from the ISP.

```
!
softwire-configuration swconfig
 method dhcp
 map-version rfc
 upstream-interface eth1
!
interface eth1
 ipv6 enable
!
interface vlan1
 ip address 192.168.2.1/24
!
interface tunnel2
 tunnel softwire swconfig
 tunnel mode map-e
!
ip route 0.0.0.0/0 tunnel2
```

Advanced configuration with static MAP rule definition

AlliedWare Plus supports static MAP rule configuration. However, this should only be used in a live production network where the ISP has provided the exact parameters that are needed. Static MAP rule may also be used in cases where a user is performing testing in an isolated test environment.

Example 1 The following static configuration contains a Basic Mapping Rule (BMR) only. This defines the IPv4 address for the tunnel and the IPv6 prefix. The IPv4 address will automatically be assigned to the tunnel and NAT masquerading will automatically be configured to translate IPv4 egressing via the tunnel. It also specifies the Port Set ID (PSID) and length used for IP address sharing. All IPv4 traffic will be sent via the Border Router (BR) - giving a hub and spoke network architecture.

```
softwire-configuration softwire1
 map-version rfc
 method static
 br-address 2001:db8::1
 rule 1 ipv4-prefix 192.0.2.23/32 ipv6-prefix 2001:db8:1:1781::/64 psid-length 8
psid 129
 upstream-interface eth1
!
!
interface eth1
 ipv6 address 2001:db8:1:1781::1111/64
 ipv6 enable
!
interface vlan1
 ip address 192.168.2.1/24
!
interface tunnel0
 tunnel mode map-e
 tunnel softwire softwire1
!
ip route 0.0.0.0/0 tunnel0
```

Example 2 The following static configuration contains a BMR that also acts as a Forwarding Mapping Rule. It achieves the same BMR result as the previous example but also defines the parameters needed to forward IPv4 traffic directly to other hosts with the 192.0.2.0/24 subnet without going via the BR (mesh-mode).

The rule needs to be interpreted in conjunction with the End-user IPv6 prefix assigned to the upstream interface - 2001:db8:1:1781::/64. The rule specifies that there are 16 bits of Embedded Address information (EA bits). Based on the Rule IPv6 prefix length these 16 bits can be found following the first 48 bits in the End-user IPv6 prefix - “0x1781”.

The IPv4 prefix length of 24 indicates that the first 8 bits of the EA bits give the host portion of the IPv4 address of the tunnel (0x17 => 23 decimal).

The remaining 8 bits of the EA bits are the PSID (0x81 => 129 decimal). The rule also defines how the end-user IPv6 address can be derived for other hosts within the 192.0.2.0/24 subnet.

```
!
softwire-configuration softwire1
 map-version rfc
 method static
 br-address 2001:db8::1
 rule 1 ipv4-prefix 192.0.2.0/24 ipv6-prefix 2001:db8:1::/48 ea-length 16
forwarding
 upstream-interface eth1
!
!
interface eth1
 ipv6 address 2001:db8:1:1781::1111/64
 ipv6 enable
!
interface vlan1
 ip address 192.168.2.1/24
!
!
interface tunnel0
 tunnel mode map-e
 tunnel softwire softwire1
!
ip route 0.0.0.0/0 tunnel0
```

The full explanation of how MAP rules work can be found in RFC 7597.

Monitoring MAP-E

The key commands to monitor MAP-E are:

- show interface tunnel
- show softwire-configuration
- show ip interface
- show ipv6 interface

An example of each command follows, note that some of these examples are annotated:

```
awplus#show interface tunnel0
Interface tunnel0
  Link is UP, administrative state is UP
  Hardware is Tunnel
  IPv4 address 192.0.2.23/24 point-to-point 192.0.2.255  <--IPv4 address
automatically generated on the tunnel
  index 15 metric 1 mtu 1460
  <UP,POINT-TO-POINT,RUNNING,MULTICAST>
  VRF Binding: Not bound
  SNMP link-status traps: Disabled
  Bandwidth 1g
 Tunnel protocol/transport map-e, key disabled, sequencing disabled <-- Indicates
that the tunnel is in MAP-E mode
  Tunnel TTL 64
  Checksumming of packets disabled, path MTU discovery disabled
  Router Advertisement is disabled
  Router Advertisement default routes are accepted
  Router Advertisement prefix info is accepted
    input packets 0, bytes 0, dropped 0, multicast packets 0
    output packets 0, bytes 0, multicast packets 0, broadcast packets 0
    input average rate : 30 seconds 0 bps, 5 minutes 0 bps
    output average rate: 30 seconds 0 bps, 5 minutes 0 bps
  Time since last state change: 0 days 00:09:44
```

```
awplus#show softwire-configuration

Softwire Configuration: softwire1

 Configuration Source: static
 Upstream Interface: eth1
 MAP-E Version: rfc
 No LW4o6 Configuration

 Border Relay Device: 2001:db8::1
  Rule 1
   IPv4-prefix: 192.0.2.0/24
   IPv6-prefix: 2001:db8:1::/48
   Embedded address length: 16
   Forwarding: enabled
   PSID offset: default
   PSID length: default
   PSID: default (0x0)
```

```
awplus#show ip interface
Interface          IP-Address          Status          Protocol
eth1               unassigned          admin up        running
eth2               unassigned          admin up        down
lo                 unassigned          admin up        running
vlan1              192.168.2.1/24      admin up        running
tunnel0            192.0.2.23/24       admin up        running  <-- IPv4 address
automatically generated on the tunnel
```

```
awplus#show interface tunnel0
23:06:08 awplus IMISH[3462]: [manager@ttyS0]show interface tunnel0
Interface tunnel0
  Link is UP, administrative state is UP
  Hardware is Tunnel
  IPv4 address 192.0.2.23/24 point-to-point 192.0.2.255  <-- IPv4 address
automatically generated on the tunnel
  index 14 metric 1 mtu 1460
  IPv4 mss 1420
  <UP,POINT-TO-POINT,RUNNING,MULTICAST>
  VRF Binding: Not bound
  SNMP link-status traps: Disabled
  Bandwidth 1g
  Tunnel protocol/transport map-e, key disabled, sequencing disabled <-- Indicates
that the tunnel is in MAP-E mode
  Tunnel TTL 64
  Checksumming of packets disabled, path MTU discovery disabled
  Router Advertisement is disabled
  Router Advertisement default routes are accepted
  Router Advertisement prefix info is accepted
    input packets 4191, bytes 5161783, dropped 0, multicast packets 0
    output packets 2908, bytes 422758, multicast packets 0, broadcast packets 0
    input average rate : 30 seconds 581.58 Kbps, 5 minutes 128.42 Kbps
    output average rate: 30 seconds 43.65 Kbps, 5 minutes 11.54 Kbps
    input peak rate 2.71 Mbps at 2018/09/27 23:05:49
    output peak rate 233.60 Kbps at 2018/09/27 23:05:24
  Time since last state change: 0 days 00:01:06
```

```
awplus#show ipv6 interface
  * = Autoconfigured Address
Interface     IPv6-Address                    State        Status      Protocol
eth1          2001:db8:1:1781::1111/64        preferred    admin up    running
<--This is the MAP-E IPv6 address
              fe80::21a:ebff:fe98:3b0b/64     preferred
eth2          unassigned                      N/A          admin up    running
lo            unassigned                      N/A          admin up    running
vlan1         unassigned                      N/A          admin up    running
tunnel0       unassigned                      N/A          admin up    running
```

If the softwire configuration referenced by the tunnel is incomplete or absent, the following output for the command **show int tunnel0** will be displayed:

```
awplus#show int tunnel0
Interface tunnel0
  Link is DOWN, administrative state is UP
  Hardware is Tunnel
  Tunnel interface is inactive:
    Specified softwire configuration is incomplete
```

The BR is a mandatory item that needs to be specified in a static configuration. If this hasn't been configured the **show softwire-configuration** command will indicate it is missing.

```
awplus#show softwire-configuration

Softwire Configuration: softwire1

 Configuration Source: static
 Upstream Interface: eth1
 MAP-E Version: rfc
 No LW4o6 Configuration

 Border Relay Device: Not Set    <-- HERE
  Rule 1
   IPv4-prefix: 192.0.2.0/24
   IPv6-prefix: 2001:db8:1::/48
   Embedded address length: 16
   Forwarding: enabled
   PSID offset: default
   PSID length: default
   PSID: default (0x0)
```

If the softwire configuration has not been referenced from the tunnel configuration the following output for the command **show int tunnel0** with be displayed:

```
awplus#show int tunnel0
Interface tunnel0
  Link is DOWN, administrative state is UP
  Hardware is Tunnel
  Tunnel interface is inactive:
    Tunnel is not yet fully configured
```

NAT64 and DNS64

Introduction

As the world moves to IPv6, network administrators need to manage both IPv4 and IPv6. They still have to support IPv4 because many servers only use IPv4, both on the Internet and within their own networks. NAT64 is a technology that helps IPv6 devices communicate with IPv4 servers.

The main idea behind NAT64 is it allows providers and customers connected to that provider to be running IPv6 only, yet still be able to connect to IPv4 only services out on other networks such as, of course, the Internet.

DNS64 translates DNS queries and NAT64 translates IP packets, enabling IPv6 devices to connect with IPv4 servers.

RFC 6146 defines NAT64 and DNS64 as methods to help network administrators transition their networks to using only IPv6. NAT64 and DNS64 use existing DNS server technologies, eliminating the need for special-purpose servers for host-to-address mappings.

How DNS64 facilitates IPv6 hosts accessing IPv4-only websites

If you have an IPv6-only host trying to access a website that only runs on IPv4, you need to use an IPv6 address that translates to the IPv4-only destination.

This is where the DNS64 comes in. DNS64 gets that IPv4 address, translates that into an IPv6 address, and provides that back into the client. The way it does this is it goes out and actually queries the DNS route servers, and finds the IPv4 address of that destination.

Then that IPv4 address of the destination, is embed into the NAT64 address. The beginning part, the prefix, is put at the beginning of the address, and the IPv4 address is put in the host portion of the address. There is a more detailed explanation of this communication process next.

How does NAT64 work?

The key components of NAT64 are:

- NAT64, which performs the translation of IPv6 header to IPv4 header and back.
- DNS64, which translates DNS queries and their answers between AAAA and A.

What does ‘AAAA’ and ‘A’ mean?

- The 'AAAA' stands for **IPv6 address record**. Specifically, it refers to the type of DNS record that maps a domain name to an IPv6 address.
- This is analogous to the 'A' record, which maps a domain name to an **IPv4 address record**, but the 'AAAA' record is used for the larger, 128-bit IPv6 addresses.

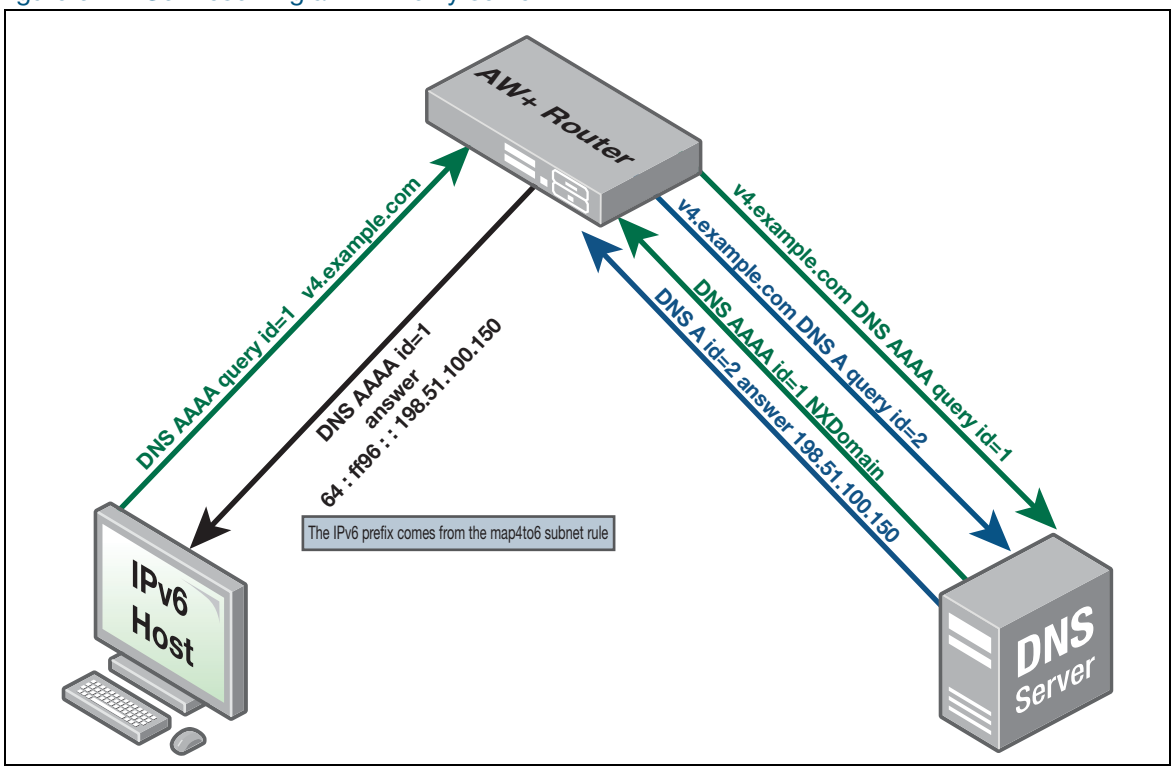
Breakdown:

1. The client sends a DNS query containing one or more AAAA (IPv6) RR questions.
2. DNS64 records the DNS query transaction ID and adds A (IPv4) RR questions for each of the AAAA RR questions, then forwards the DNS query as normal.
3. DNS64 receives a DNS response matching the original transaction ID.
4. For each name lookup with an A answer but no corresponding AAAA answer, DNS64 translates the A answer to AAAA. The IPv4 addresses in the answers become the suffix on DNS64's IPv6 prefix.
5. For each name lookup that has both A and AAAA answers, DNS64 removes the A answer and uses the AAAA answer.
6. The response is then forwarded to the client as normal.
7. The client sends a packet to the server's IPv6 address.
8. NAT64 replaces the IPv6 header with an IPv4 header. The destination IPv4 address is derived from the last four octets of the IPv6 address.
9. The server responds to the client using the mapped IPv4 address.
10. The device reverses the process, replacing the IPv4 header with an IPv6 header.
11. The client receives an IPv6 reply from the IPv4 server.

DNS64 resolving an IPv4-only server

For a map6to4 prefix of 64:ff9b::/96 and an IPv4 server of 198.51.100.150, the IPv6 host behind NAT64 would address the IPv4 server using the IPv6 address 64:ff9b::c633:6496, often written as 64:ff9b::198.51.100.150.

Figure 6: DNS64 resolving an IPv4-only server



The communication steps are as follows:

1. **IPv6 Host to AW+ Router**: The IPv6 host sends a DNS AAAA query for v4.example.com to the AW+ router.
2. **AW+ Router to DNS Server**: The AW+ router forwards two DNS queries in response to the IPv6 host's DNS query:
   - The original DNS AAAA query
   - And a DNS A query, translated from the original DNS AAAA query
3. **DNS Server to AW+ Router**:
   - The DNS server responds to the DNS AAAA query with NXDOMAIN
   - The DNS server responds to the DNS A query with the address 198.51.100.150
4. **AW+ Router to IPv6 Host**: The AW+ router translates the DNS response from A to AAAA and sends it back to the IPv6 host. (answer 64:ff9b::198.51.100.150)
5. **IPv6 Host to AW+ Router**: Now that the AW+ router has received both DNS responses, it translates the DNS A response into a DNS AAAA response. The IPv6 prefix (64:ff96::) is the configured map4to6 IPv6 prefix.
6. **AW+ Router to IPv6 Host**: The IPv6 host receives the DNS AAAA response and can now communicate with the v4.example.com server.

This flow ensures the IPv6 host can communicate with an IPv4-only host through the AlliedWare Plus router that handles translation between the protocols.

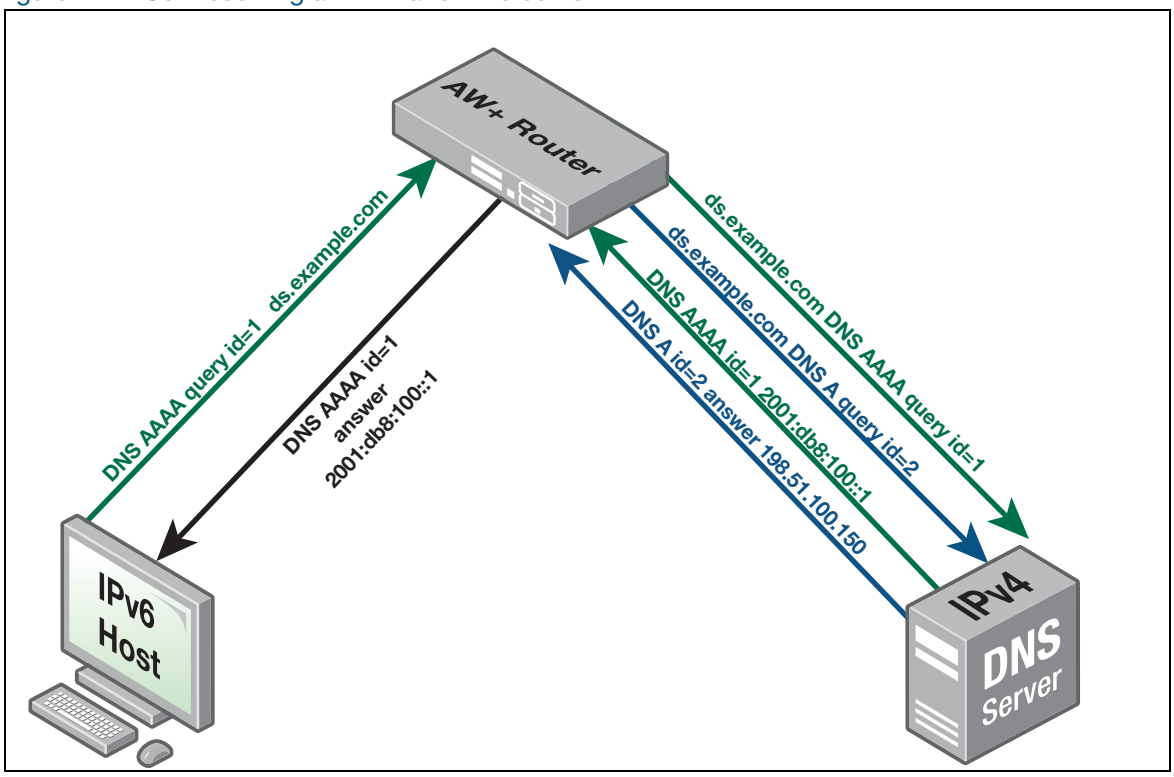
The other important thing to note here is where the division is in the IPv6 address. Notice where the double colons are that separate the beginning and end of the address, which means that the IPv4 portion is put into the host area of the IPv6 address.

For an IPv6 host, the AlliedWare Plus router may masquerade using its IPv4 upstream interface, so the IPv4 server will see the IPv6 host as the AlliedWare Plus router IPv4 address.

DNS64 resolving an IPv4 and IPv6 server

An example of DNS64's behavior if the queried hostname resolves to an IPv4 and IPv6 address (both the A query and AAAA query are answered). This would occur for a dual stack (IPv4 and IPv6) server.

Figure 7: DNS64 resolving an IPv4 and IPv6 server



The communication steps are as follows:

1. **IPv6 Host to AW+ Router**: The IPv6 host sends a DNS AAAA query for ds.example.com to the AW+ router.
2. **AW+ Router to DNS Server**: The AW+ router forwards two DNS queries in response to the IPv6 host's DNS query.
   - The original DNS AAAA query
   - DNS A query, translated from the original DNS AAAA query
3. **DNS Server to AW+ Router**: The DNS server responds to the DNS AAAA query with the address 2001:db8:100::1
4. **AW+ Router to IPv6 Host**: The DNS server responds to the DNS A query with the address 198.51.100.150.
5. **IPv6 Host to AW+ Router**: Now that the AW+ router has received both DNS responses, it ignores the DNS A query and forwards the DNS AAAA response to the IPv6 host. In this case, no translation was needed as both the IPv6 host and ds.example.com support IPv6.
6. **AW+ Router to IPv6 Host**: The IPv6 host receives the DNS AAAA response and can now communicate with the ds.example.com server.

Note that the above diagrams focus on the DNS transactions. Any HTTP traffic as a result of the DNS query would be handled by NAT64.

NAT64 translation

NAT64 translation will only be performed when necessary (when the server is IPv4 only). NAT64 communicates in a stateless manner, meaning the translation process does not keep track of the state of each communication session.

The communication steps are:

1. Translate matching IPv6 packets to IPv4 packets, using the configured map6to4 and map4to6 subnets.
2. Translate matching IPv4 packets to IPv6 packets, by stripping off the top 96 bits of the IPv6 address.

Configuring NAT64

There are three parts to configuring NAT64:

1. Configure the map4to6 instance with map6to4 and map4to6 subnets.
2. Set up the tunnel interface in map4to6 mode.
3. Configure the DNS resolver to use the map4to6 instance.

Simple configuration example

The following configuration example reflects the configuration used in the above Figure 7 on page 43.

1. **Configure the map4to6 instance**

```
awplus# configure terminal
awplus(config)# 4to6-mapping example6to4
awplus(config-4to6-mapping)# map4to6 subnet 0.0.0.0/0 64:ff9b::/96
awplus(config-4to6-mapping)# map6to4 subnet 2001:db8:20::/96 0.0.0.0/0
```

This is how the address is configured, the device then translates that address into 2001:db8:20::c000:0296/96, which is the Hex form of 2001:db8:20::192.0.2.150/96.

No limits are needed for NAT64 as the mapping is static. This is because the entire IPv4 address space can be represented within a single /96 IPv6 address block.

2. **Set up the tunnel interface in map4to6 mode**.

```
awplus#configure terminal
awplus(config)#interface tunnel0
awplus(config-if)#tunnel mode map4to6
```

3. **Configure the DNS resolver to use the map4to6 instance.**

```
awplus#configure terminal
awplus(config)# ip dns forward
awplus(config)# ip dns forward dns64 example6to4
```

Full configuration example

```
zone inet
 network all
  ip subnet 0.0.0.0/0 interface eth2
!
zone lan
 network all
  ip subnet 0.0.0.0/0
!
nat
 rule masq any from lan to inet
 enable
!
4to6-mapping example6to4
  map6to4 subnet 0.0.0.0/0 2001:db8:20::/96
  map4to6 subnet 0.0.0.0/0 64:ff9b::/96
!
ipv6 dhcp pool lanv6
 address range 2001:db8:20::192.0.2.100 2001:db8:20::192.0.2.150
 dns-server 2001:db8:20::192.168.0.2.1
!
service dhcp-server
!
interface eth1
 ipv6 address 2001:db8:20::192.0.2.1/64
 no ipv6 nd suppress-ra
 ipv6 nd managed-config-flag
 ipv6 nd other-config-flag
 ipv6 nd dns-server 2001:db8:20::192.0.2.1
 ipv6 dhcp server lanv6
!
interface eth2
 ip address 203.0.113.1/24
!
interface tunnel0
 tunnel 4to6-mapping example6to4
 tunnel mode map4to6
 ip address 198.51.100.1/32
 ipv6 enable
!
ipv6 forwarding
!
ip route 192.0.2.0/24 tunnel0
!
ipv6 route 64:ff9b::/96 tunnel0
!
ip dns forwarding
ip dns forwarding dns64 example6to4
```

With the above configuration, the system will use the AlliedWare Plus router's upstream interface for the translated IPv4 addresses.

Although the IPv6 subnet is 2001:db8:20::/64, the DHCP range for assigning IPv6 addresses should be within 2001:db8:200::192.0.2.100 to 2001:db8:200::192.0.2.150.

This ensures that the addresses match the NAT64 IPv4 range (192.0.2.0/24). If the DHCP assigns addresses outside this range, those addresses will be sent to a different IPv4 network after translation, which can cause routing issues.

Show command

This show command displays IPv4 to IPv6 mapping details:

```
awplus#show 4to6-mapping
IPv4 to IPv6 Instances:

 Mapping Instance: example6to4
  map4to6 Mappings:
   user subnet mapping between 0.0.0.0/0 and 64:ff9b::/96
  map6to4 Mappings:
   user subnet mapping between 0.0.0.0/0 and 2001:db8:20::/96
```