## Table of Contents

- Introduction ........................................................................................................... 4
- Scope ....................................................................................................................... 4
- Design Considerations .......................................................................................... 4
- Terminology ............................................................................................................ 4
- Implementation ...................................................................................................... 5
- Configuration Example .......................................................................................... 6
  - Chassis Configuration .......................................................................................... 7
  - Network Address and Port Translation Configuration ......................................... 7
  - Softwire Concentrator Configuration .................................................................. 7
  - Service Set Configuration .................................................................................... 8
  - Interfaces Configuration ....................................................................................... 8
- Configuration Tips .................................................................................................. 9
  - MTU-V6 ................................................................................................................ 9
  - TCP-MSS Adjustment Configuration .................................................................. 9
- Operations and Management ................................................................................ 10
  - IPv6 Address Reachability for Softwire .............................................................. 10
  - DS-Lite CLI ........................................................................................................... 12
    - Flows and Conversation ..................................................................................... 12
    - NAT .................................................................................................................... 13
  - Softwire Statistics ............................................................................................... 13
  - Application-Level Gateway (ALG) Support ........................................................ 14
  - DiffServ Code Point (DSCP) Support .................................................................. 15
  - DHCP Options for DS-Lite .................................................................................. 15
- Advanced Scenarios ............................................................................................. 16
  - Multiple Concentrators across Different Service Sets in One Services PIC .......... 16
  - Network Address and Port Translation ............................................................... 17
  - Softwire Concentrator Configuration ................................................................ 19
  - Service-Set Configuration ................................................................................... 20
  - Interfaces Configuration ....................................................................................... 20
  - Examples of ALG Support for FTP and RTSP Protocols ..................................... 21
  - Softwires without NAT ....................................................................................... 23
  - Redundancy and Load Balancing Using Anycast ................................................ 25
- Configuration ........................................................................................................ 26
  - Kay Configuration ................................................................................................ 27
  - Corleone-linux1 Configuration ........................................................................... 27
  - Vito Configuration ............................................................................................... 27
  - Tennis Configuration ............................................................................................ 28
Handball Configuration ........................................................................................................... 30
Fredo Configuration .................................................................................................................. 31
Traffic Test ................................................................................................................................ 31
Ping from Kay to Fredo ............................................................................................................. 31
Verify Flows on Tennis and Handball ...................................................................................... 31
Shutdown ge-2/1/0 on Tennis .................................................................................................... 32
Ping and Verify Flows .................................................................................................................. 32
Multiple services PICs on One AFTR .......................................................................................... 33
AFTR Configuration ................................................................................................................... 33
Verify Load Balancing ................................................................................................................ 37
Verify Redundancy ..................................................................................................................... 38
Bibliography .............................................................................................................................. 39
About Juniper Networks ............................................................................................................. 40

Table of Figures
Figure 1: Packet flow on the AFTR ............................................................................................. 5
Figure 2: DS-Lite deployment example ...................................................................................... 6
Figure 3: Advanced scenarios with DS-Lite ................................................................................ 16
Figure 4: DS-Lite deployment with multiple AFTRs ................................................................. 26
Figure 5: Anycast setup for DS-Lite .......................................................................................... 26
Figure 6: Example to illustrate the use of services PICs on a single AFTR ................................. 33
Introduction

Today’s Internet service providers (ISPs) have to address the problem of IPv4 address exhaustion and in parallel enable IPv6 services over IPv4 infrastructure. IPv6 and IPv4 are expected to coexist for a long time to come. Providers are looking at mixed approaches, from enabling IPv6 services in the least disruptive ways on legacy infrastructure to deploying IPv6 from the outset in greenfield applications. Dual-Stack Lite (DS Lite) is one promising approach to allow ISPs to transition to IPv6 and at the same time deal with IPv4 address depletion [IPv4-Exhaustion] [IANA-Allocation].

Scope

This solution implementation guide provides an overview of DS-Lite with configuration steps and basic design guidelines when implementing DS-Lite with the Multiservices PIC (MS-PIC) or the Multiservices Dense Port Concentrator (MS-DPC) blade on the Juniper Networks® MX Series 3D Universal Edge Routers. This guide also covers some operational aspects and advanced configuration scenarios with DS-Lite. For an in-depth look at IPv6 transition mechanisms and methods to address IPv4 address exhaustion, please refer to www.juniper.net/ipv6.

Design Considerations

DS-Lite is a promising approach that uses IPv6-only links between the provider and the customer. Using this approach, a service provider can deploy IPv6 and still provide an IPv4 service. It allows the ISP to migrate to an IPv6 access network without changing end user software, and the current device used by the customer to access the Internet remains as before. To do this, DS-Lite encapsulates IPv4 packets originated by the existing end hosts into IPv6 packets. When a device in the customer network sends an IPv4 packet to an external destination, the IPv4 packet is encapsulated in an IPv6 packet for transport into the provider network. These IPv4-in-IPv6 tunnels are called softwires [RFC4925]. The softwires terminate in a softwire concentrator at some point in the network where IPv4 packets are decapsulated and sent through a Carrier Grade Network Address Translation (CGNAT) device [RFC2663]. At the CGNAT, the packet is decapsulated to IPv4 and NAT44 (which translates an IPv4 address to another IPv4 address) before delivery to the public Internet. This tunneling of IPv4 packets enables IPv4 applications and IPv4 hosts to communicate with the IPv4 Internet over IPv6-only links.

The DS-Lite architecture is characterized by a combination of an IPv6-only access network and IPv4 (or dual-stack) hosts in the home network. IPv6 packets originated by hosts in the subscriber’s home network are transported natively over the access network. IPv4 packets, on the other hand, are encapsulated in IPv6 according to [RFC2473].

It is important to note that the IPv4 packets originated by the end hosts have private (and possibly overlapping) IP addresses, and therefore Network Address Port Translation (NAPT) [RFC2663] needs to be applied to them.

Juniper Networks M Series Multiservice Edge Routers, T Series Core Routers, and MX Series 3D Universal Edge Routers can all play the role of a softwire concentrator

Terminology

- B4 (Basic Bridging Broadband Element): The B4 element is a function implemented on a dual-stack capable node, either a directly connected device or a home gateway that creates a tunnel to an AFTR.
- AFTR: An AFTR (Address Family Transition Router) element is the combination of an IPv4-in-IPv6 tunnel endpoint and an IPv4-IPv4 NAT implemented on the same node.
- Encapsulation: In the context of DS-Lite, encapsulation refers to a multipoint-to-point IPv4-in-IPv6 tunnel ending on a service provider AFTR.
- MS-PIC/MS-DPC: Multiservice PIC/Multiservice DPC. MS-PIC is also referred to as services PIC in this document.

1Notations in square brackets refer to the Bibliography section at the end of this guide.
Implementation

Juniper has implemented an AFTR element in its MS-DPC and services PIC. IPv6 packets coming from a B4 are steered to a MS-DPC where, according to the configuration, a softwire is created. The IPv4 packets are extracted, NAT rule lookup is done, and address translation is performed. The translated IPv4 packets are then sent to the Internet. All these functions are performed in a single pass of the services PIC (see Figure 1 below).

In the reverse path, IPv4 packets are sent to the MS-DPC where de-NAT is performed, encapsulated in an IPv6 packet corresponding to the proper softwire, and sent to the B4.

The softwires are automatically created as IPv6 packets are received. IPv4 flows created by the encapsulated packets are associated with the specific softwire that carried them in the first place. When the last IPv4 flow associated with a softwire is destroyed, the softwire itself goes away. Therefore, the configuration is simplified and there is no need to create or manage tunnel interfaces.

The softwire construct does not use interface resources by design; therefore the number of established softwires does not affect throughput, and scalability is independent of the number of interfaces.

Figure 1: Packet flow on the AFTR

1. Customer packet arrives on inbound PIC
2. First lookup reveals services enabled, identifies appropriate Service PIC
3. Perform any filtering, then forward to Service PIC for processing
4. Processed original packet sent to PFE for final destination lookup
5. Original packet forwarded to outbound PIC
6. Original packet transmitted
Configuration Example

In this section, we will configure an MX Series router with an MS-DPC as an AFTR in a way that replicates the example flow found in [DS-Lite]. The example flow is copied below for continuity and reference purposes.

```
+-----------+          +-----------+
|    Host   |          |    Host   |
|-----------+          |-----------+          +-----------+
| 10.0.0.1  |          | 10.0.0.2  |
|           |          |           |          +-----------+
|           |          |           |          |    Home router |
|-----------+          |-----------+          +-----------+
|           |          |           |          |        B4        |
|-----------+          |-----------+          +-----------+
|           |          |           |          | 2001:0:0:1::1 |
|           |          |           |          | <-IPv4-in-IPv6 softwire |
|-----------+          |-----------+          +-----------+
|           |          |           |          / isp core network \ |
|-----------+          |-----------+          \          /                      |
|           |          |           |          2001:0:0:2::1          |
|-----------+          |-----------+          +-----------+
|           |          |           |          |    AFTR       |
|-----------+          |-----------+          +-----------+
|           |          |           |          |    Concentrator |
|-----------+          |-----------+          +-----------+
|           |          |           |          |        NAT      |
|-----------+          |-----------+          +-----------+
|           |          |           |          | 129.0.0.1    |
|-----------+          |-----------+          +-----------+
|           |          |           |          / ipv4 host \
|-----------+          |-----------+          \          /
|           |          |           |          128.0.0.1 |
|-----------+          |-----------+          +-----------+
```

**Figure 2: DS-Lite deployment example**

In Figure 2, AFTR is an MX Series router with two gigabit interfaces and an MS-DPC. The interface facing the B4 element is ge-3/1/5 and the one facing the Internet is ge-3/1/0. The sections below illustrate configuration of the chassis, physical interface, NAT rules, softwire rule, and service set configuration.
**Chassis Configuration**

The DS-Lite feature is supported in a Layer 3 package. Assume that the services PIC is in Flexible PIC Concentrator (FPC) 0 slot 0.

```
[edit chassis]
regress@cypher# show
fpc 0 {
    pic 0 {
        adaptive-services {
            service-package layer-3;
        }
    }
}
```

**Network Address and Port Translation Configuration**

The next step is to configure NAT and Port Address Translation (PAT) rules. These rules are applied in the service set configuration section below.

```
regress@sledding# show services nat
pool pl {
    address 129.0.0.1/32;
    port {
        automatic;
    }
}
rule rl {
    match-direction input;
    term t1 {
        from {
            source-address {
                10.0.0.0/16;
            }
        }
        then {
            translated {
                source-pool pl;
                translation-type {
                    source dynamic;
                }
            }
            syslog;
        }
    }
}
```

**Softwire Concentrator Configuration**

First a softwire concentrator of type DS-Lite needs to be created with its associated IPv6 address. It is important to give the softwire concentrator a name in order to be able to refer to it in logs, in command-line interface (CLI) commands, and for general operations and management.

Then a softwire rule needs to be created. The rule below basically says that any traffic destined to the softwire concentrator ds1 will create a new softwire. Optionally, one could configure more elaborate match conditions in order to perform certain softwire initiator actions.
[edit]
regress@sledding# show services softwire
softwire-concentrator {
   ds-lite ds1 {
      softwire-address 1001::1;
      mtu-v6 1460;
   }
}
rule r1 {
   match-direction input;
   term t1 {
      then {
         ds-lite ds1;
      }
   }
}

Service Set Configuration
The NAT and softwire rules configured in the last two sections are used to create the service set as illustrated below.

service-set sset {
   syslog {
      host local {
         services any;
      }
   }
   softwire-rules r1;
   nat-rules r1;
   interface-service {
      service-interface sp-0/0/0.0;
   }
}

Interfaces Configuration
The next step is to configure the service interface and physical interface on the AFTR, and then associate the relevant service to the ingress and egress interface on the AFTR. Service sets can be configured in both interface style and next-hop VPN routing and forwarding (VRF) style. Interface style configuration is displayed below.

[edit]
regress@sledding# show interfaces sp-0/0/0
unit 0 {
   family inet;
   family inet6;
}
regress@sledding# show interfaces ge-3/1/0
description AFTR-Internet;
unit 0 {
   family inet {
      address 128.0.0.2/24;
   }
}
regress@sledding# show interfaces ge-3/1/5
description AFTR-B4;
unit 0 {
family inet;
family inet6 {
    service {
        input {
            service-set sset;
        }
        output {
            service-set sset;
        }
        address 2001:0:0:2::1/48;
    }
}

Configuration Tips

As there is no automatic detection of tunnel maximum transmission unit (MTU) on the AFTR, MTUs on the IPv6 network need to be properly configured. Alternatively, TCP maximum segment size (MSS) adjustment can be configured in order to make sure that TCP traffic works through links with different MTUs.

**MTU-V6**

This attribute sets the maximum transmission unit when encapsulating IPv4 packets into IPv6. If the final length is greater than the MTU, the IPv6 packet will be fragmented. This knob is mandatory, since it depends on other network parameters under administrator control.

**TCP-MSS Adjustment Configuration**

In order to avoid or minimize IPv6 fragmentation, TCP-MSS adjustment can be configured on a service set basis. The service set referenced above with a TCP-MSS configuration of 1,024 would look like the following:

```
service-set sset {
    syslog {
        host local {
            services any;
        }
    }
    tcp-mss 1024;
    softwire-rules r1;
    nat-rules r1;
    interface-service {
        service-interface sp-0/0/0.0;
    }
}
```
Operations and Management

There are several commands that can be used to check the workings of DS-Lite implementations.

IPv6 Address Reachability for Softwire

The softwire concentrator address need not be configured on any physical media interface. It can be any IPv6 address configured under softwire as shown in the Softwire Concentrator Configuration section below.

If configured on a physical media interface, it inherits IPv6 and the tunnel behavior of Juniper Networks® Junos operating system. If configured only under softwire, the SERVICES PIC handles the job of responding to Internet Control Message Protocol (ICMP) pings to configured IPv6 addresses and doesn’t handle ICMP translations.

The softwire address anchored on the services PIC can be pinged from any IPv6 host as long as it has a route. This is typically used by B4s to see if an obtained softwire address is configured on AFTR and is reachable.

Here is an example where 1001::1 is configured as a softwire address and is pinging it from B4 with address 3000::2.

```
regress@sledding# show services softwire
softwire-concentrator {
    ds-lite ds1 {
        softwire-address 1001::1;
        mtu-v6 1460;
    }
}
rule r1 {
    match-direction input;
    term t1 {
        then {
            ds-lite ds1;
        }
    }
}
regress@sledding# show interfaces ge-3/1/5
description AFTR-B4;
unit 0 {
    family inet;
    family inet6 {
        service {
            input {
                service-set sset;
            }
            output {
                service-set sset;
            }
            address 3000::1/48;
        }
    }
    description AFTR-B4;

regress@sledding# show services softwire
softwire-concentrator {
    ds-lite ds1 {
        softwire-address 1001::1;
        mtu-v6 1460;
    }
}
rule r1 {
    match-direction input;
    term t1 {
        then {
            ds-lite ds1;
        }
    }
}
regress@sledding# show interfaces ge-3/1/5
description AFTR-B4;
unit 0 {
    family inet;
    family inet6 {
        service {
            input {
                service-set sset;
            }
            output {
                service-set sset;
            }
            address 3000::1/48;
        }
    }
    description AFTR-B4;
```
IPv4 Address Softwire Reachability

192.0.0.1 is reserved for AFTR. B4s can usually ping this address to see if AFTR is alive and IPv4-in-IPv6 tunneling is working properly.

The following example shows output of ping from B4 to 192.0.0.1.

regress@sledding# run show services stateful-firewall flows

Interface: sp-0/0/0, Service set: sset

<table>
<thead>
<tr>
<th>Flow</th>
<th>State</th>
<th>Direction</th>
<th>Frame</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPIP</td>
<td></td>
<td>Forward</td>
<td>0</td>
</tr>
<tr>
<td>ICMP</td>
<td></td>
<td>Watch</td>
<td>1</td>
</tr>
<tr>
<td>NAT Source</td>
<td></td>
<td>I</td>
<td>5</td>
</tr>
<tr>
<td>Softwire</td>
<td></td>
<td>I</td>
<td>10</td>
</tr>
<tr>
<td>DS-LITE</td>
<td></td>
<td>Forward</td>
<td>0</td>
</tr>
<tr>
<td>ICMP</td>
<td></td>
<td>Watch</td>
<td>5</td>
</tr>
<tr>
<td>NAT Dest</td>
<td></td>
<td>O</td>
<td>5</td>
</tr>
<tr>
<td>Softwire</td>
<td></td>
<td>I</td>
<td>10</td>
</tr>
</tbody>
</table>
regress@sledding# run show services softwire flows

Interface: sp-O/0/0, Service set: sset

<table>
<thead>
<tr>
<th>Flow</th>
<th>State</th>
<th>Direction</th>
<th>Frame Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICMP</td>
<td>Watch</td>
<td>I</td>
<td>10</td>
</tr>
<tr>
<td>NAT Source</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Softwire</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DS-LITE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ICMP</td>
<td>Watch</td>
<td>O</td>
<td>10</td>
</tr>
<tr>
<td>NAT Dest</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Softwire</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

DS-Lite CLI

The administrator has a large number of commands available to perform operations and management. In the next sections, we discuss a few of them. The user is referred to the appropriate Junos OS manual for a more complete set.

Flows and Conversation

The administrator can use the command “show services stateful-firewall flows” and “show services softwire flows” to check the creation of softwires, as well as pre-NAT and post-NAT flows within it. An example output for the configuration discussed above is shown below.

regress@sledding> show services stateful-firewall flows

Interface: sp-O/0/0, Service set: sset

<table>
<thead>
<tr>
<th>Flow</th>
<th>State</th>
<th>Direction</th>
<th>Frame Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>DS-LITE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TCP</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NAT Dest</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Softwire</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

regress@sledding> show services softwire flows

Interface: sp-O/0/0, Service set: sset

<table>
<thead>
<tr>
<th>Flow</th>
<th>State</th>
<th>Direction</th>
<th>Frame Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>DS-LITE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TCP</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NAT Dest</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Softwire</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

regress@sledding> show services softwire flows

Interface: sp-O/0/0, Service set: sset

<table>
<thead>
<tr>
<th>Flow</th>
<th>State</th>
<th>Direction</th>
<th>Frame Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>DS-LITE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TCP</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NAT Dest</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Softwire</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Output can be filtered using more specific fields like AFTR and/or B4 address. Another important command to show conversations (collection of related flows) is below.

```
regress@sledding> show services stateful-firewall conversations
Interface: sp-0/0/0, Service set: sset
Conversation: ALG protocol: tcp
Number of initiators: 1, Number of responders: 1
```

<table>
<thead>
<tr>
<th>Flow</th>
<th>State</th>
<th>Direction</th>
<th>Frame Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP 10.0.0.1:1025 -&gt; 128.0.0.1:80</td>
<td>Forward</td>
<td>I</td>
<td>372755</td>
</tr>
<tr>
<td>NAT Source 10.0.0.1:1025 -&gt; 129.0.0.1:1024</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Softwire 2001:0:0:1::1 -&gt; 1001::1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TCP 128.0.0.1:80 -&gt; 129.0.0.1:1024</td>
<td>Forward</td>
<td>O</td>
<td>794083</td>
</tr>
<tr>
<td>NAT Dest 129.0.0.1:1024 -&gt; 10.0.0.1:1025</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Softwire 2001:0:0:1::1 -&gt; 1001::1</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**NAT**

In order to find global NAT statistics related to pool usage, the following “show services nat pool detail” commands could be used. These are normally used in conjunction with “show service stateful-firewall flows.” See example below.

```
regress@sledding> show services nat pool detail
Interface: sp-0/0/0, Service set: sset
NAT pool: p1, Translation type: dynamic
Address range: 129.0.0.1-129.0.0.1
Port range: 512-65535, Ports in use: 1, Out of port errors: 0, Max ports used: 1
```

**Softwire Statistics**

This section illustrates various commands used to display softwire statistics, associated flow count, and track error counts.

```
regress@sledding> show services softwire statistics
DS-Lite Statistics:

Services PIC Name: :sp-0/0/0

Statistics
   ------------
   Softwires Created : 2
   Softwires Deleted : 1
   Softwires Flows Created : 2
   Softwires Flows Deleted : 1
   Slow Path Packets Processed : 2
   Fast Path Packets Processed : 274240
   Fast Path Packets Encapsulated : 583337
   Rule Match Failed : 0
   Rule Match Succeeded : 2
   IPv6 Packets Fragmented : 0
```
Transient Errors
----------------
Flow Creation Failed - Retry : 0
Slow Path Failed - Retry : 0

Errors
------
Softwire Creation Failed : 0
Flow Creation Failed : 0
Slow Path Failed : 0
Packet not IPv4-in-IPv6 : 0
IPv6 Fragmentation Error : 0
Slow Path Failed - IPv6 Next Header Offset : 0
Decapsulated Packet not IPv4 : 0
Fast Path Failed - IPv6 Next Header Offset : 0
No Softwire ID : 0
No Flow Extension : 0
Flow Limit Exceeded : 0

regress@sledding> show services softwire
Interface: sp-0/0/0, Service set: sset
Softwire                                   Direction     Flow count
2001:0:0:1::1   ->        1001::1          I             3

regress@sledding> show services softwire count
Interface   Service set                    DS-Lite       6RD
sp-0/0/0    sset                           1             0

regress@sledding> show services softwire flows count
Interface   Service set                                  Flow count
sp-0/0/0    sset                                         3

Application-Level Gateway (ALG) Support

Currently ICMP, HTTP, FTP, and Real-Time Streaming Protocol (RTSP) are supported with DS-Lite.

In the case of traceroute originated from the home host to an IPv4 host in the Internet, the softwire concentrator will not send an ICMP error back and therefore will not be properly identified as an intermediate hop. But traceroute will work properly nonetheless.

See output below of traceroute flows from the client to the home host to the IPv4 host on the Internet based on the previous example.
DiffServ Code Point (DSCP) Support

Today, the type of service (ToS) bits of the IPv4 packet are mapped to the priority field in the IPv6 packet during encapsulation. During decapsulation, the IPv6 priority field is copied to the IPv4 header if configured to do so. Otherwise it is not copied.

Here is a sample configuration to copy the IPv6 priority field to IPv4 during decapsulation.

```
regress@sledding# show services softwire
softwire-concentrator {
   ds-lite ds1 { 
      softwire-address 1001::1;
      mtu-v6 1460;
      +       copy-dscp;
   }
}
```

DHCP Options for DS-Lite

If the MX Series router is acting as the Dynamic Host Configuration Protocol (DHCP) v6 server, it can be used to convey the softwire concentrator addresses.

We do support configuring options within the DHCP-attributes associated with an address pool. From below, you can specify an option and populate the contents with any of the data types shown or an array of the same data types.

```
{master}[edit access address-assignment pool pool1 family inet6]
regress@alkaid# set dhcp-attributes option 1 ?
Possible completions:
  <[Enter]> Execute this command
> array  Array of values
  byte   Unsigned 8-bit value
  flag   Boolean flag value
  integer Signed 32-bit numeric value
  ipv6-address IPV6 address value
  short  Signed 16-bit numeric value
  string Character string value
  unsigned integer Unsigned 32-bit numeric value
  unsigned short Unsigned 16-bit numeric value
  |       Pipe through a command

Example 1: IPv6 address in option 33
------------------------------------
{master}[edit access address-assignment pool pool1 family inet6]
regress@alkaid# set dhcp-attributes option 33 ipv6-address 2000::1
{master}[edit access address-assignment pool pool1 family inet6]
regress@alkaid# show dhcp-attributes
option 33 ipv6-address 2000::1;

Example 2: array of bytes in option 44
--------------------------------------
{master}[edit access address-assignment pool pool1 family inet6]
regress@alkaid# set dhcp-attributes option 44 array byte [1 2 3 4 5]
{master}[edit access address-assignment pool pool1 family inet6]
regress@alkaid# show dhcp-attributes
option 44 array byte [ 1 2 3 4 5 ];
```
NOT WORKING PROPERLY: duplicate values in array eliminated (PR-512081)
--------------------------------------------------------------------------
{master}[edit access address-assignment pool pool1 family inet6]
regress@alkaid# set dhcp-attributes option 44 array byte [1 2 3 4 5 1 2 3 4 5]
{master}[edit access address-assignment pool pool1 family inet6]
regress@alkaid# show dhcp-attributes
option 44 array byte [ 1 2 3 4 5 ]; <<--- NOTE: duplicate values eliminated

Advanced Scenarios

Multiple Concentrators across Different Service Sets in One Services PIC

In this scenario, we have two different service sets (sset2, sset3) hosting different softwire concentrators. Each service set has a different set of NAT rules and pools, corresponding, for example, to an enterprise and home subscriber being NATed differently. Feature rich traffic such as RTSP, FTP (active and passive), and HTTP come through different softwires and VLANs to the corresponding service sets.

![Diagram](attachment:image.png)

Figure 3: Advanced scenarios with DS-Lite
Network Address and Port Translation

First we configure the NAT rules. From Figure 3, note that softwire 3 (SI-3) uses NAT pool p2, and softwires 1 and 4 (SI-1, SI-4) use pool p1. Also, there is one term per application, which allows troubleshooting during early deployments.

```
regress@cypher> show configuration services nat
pool p1 {
    address 129.0.0.1/32;
    port automatic auto;
}
pool p2 {
    address 130.0.0.1/32;
    port automatic;
}
rule si-3 {
    match-direction input;
    term http-client-1 {
        from {
            source-address {
                10.0.3.1/32;
            }
            applications junos-http;
        }
        then {
            translated {
                source-pool p2;
                translation-type {
                    source dynamic;
                }
            }
        }
    }
    term ftp-passive-client-1 {
        from {
            source-address {
                10.0.3.2/32;
            }
            applications junos-ftp;
        }
        then {
            translated {
                source-pool p2;
                translation-type {
                    source dynamic;
                }
            }
        }
    }
}
rule si-1 {
    match-direction input;
    term rtsp-client-1 {
        from {
            source-address {
                10.0.0.4/32;
            }
        }
    }
}
```
applications junos-rtsp;
} then {
    translated {
        source-pool p1;
        translation-type {
            source dynamic;
        }
    }
    syslog;
}
}

rule si-4 {
    match-direction input;
    term http-client-1 {
        from {
            source-address {
                10.0.0.1/32;
                10.0.0.3/32;
            }
            applications junos-http;
        } then {
            translated {
                source-pool p1;
                translation-type {
                    source dynamic;
                }
                syslog;
            }
        }
    }
}
source-pool pl;
translation-type {
  source dynamic;
}
}
}
}

Softwire Concentrator Configuration

This section displays the multiple softwire concentrator configurations. Also note that configurations of the MTU-v6 attribute is mandatory and commit will fail if it is not provided.

softwire {
  softwire-concentrator {
    ds-lite ds2 {
      softwire-address 2002::2:0:0:0:1;
      mtu-v6 1460;
    }
    ds-lite ds3 {
      softwire-address 2002::3:0:0:0:1;
      mtu-v6 1460;
    }
    ds-lite ds4 {
      softwire-address 2002::4:0:0:0:1;
      mtu-v6 1460;
    }
  }
  rule sc-4 {
    match-direction input;
    term t1 {
      then {
        ds-lite ds4;
      }
    }
  }
  rule sc-2 {
    match-direction input;
    term t1 {
      then {
        ds-lite ds2;
      }
    }
  }
  rule sc-3 {
    match-direction input;
    term t1 {
      then {
        ds-lite ds3;
      }
    }
  }
  rule-set rs1 {
    rule sc-4;
    rule sc-2;
  }
}
Service-Set Configuration

Once the softwire rules have been created, these and the NAT rules configured in the Network Address and Port Translation section are then applied to the relevant service sets.

```bash
regress@cypher> show configuration services
service-set sset2 {
    syslog {
        host local {
            services any;
        }
    }
    softwire-rule-sets rs1;
    nat-rules si-1;
    nat-rules si-4;
    interface-service {
        service-interface sp-5/0/0;
    }
}
service-set sset3 {
    softwire-rules sc-3;
    nat-rules si-3;
    interface-service {
        service-interface sp-5/0/0;
    }
}
```

Interfaces Configuration

This section shows the configuration for all relevant interfaces on AFTR implemented on a Juniper Networks MX480 3D Universal Edge Router as shown in Figure 3.

```bash
ge-1/3/5 {
    description “AFTR-B4”;
    vlan-tagging;
    unit 2 {
        vlan-id 2;
        family inet;
        family inet6 {
            service {
                input {
                    service-set sset2;
                }
                output {
                    service-set sset2;
                }
                address 2001:0:0:2::1/48;
            }
        }
        unit 3 {
            vlan-id 3;
            family inet;
            family inet6 {
                service {
                    input {
                        service-set sset3;
                    }
                    output {
                        service-set sset3;
                    }
                }
            }
        }
    }
}
```
Examples of ALG Support for FTP and RTSP Protocols

Real-Time Streaming Protocol (RTSP) ALG

This demonstrates the RTSP ALG relevant commands with Juniper’s DS-Lite solution.

regress@cypher> show services stateful-firewall conversations application-protocol rtsp

Interface: sp-5/0/0, Service set: sset2

Conversation: ALG, Protocol: rtsp

Number of initiators: 3, Number of responders: 3

<table>
<thead>
<tr>
<th>Flow</th>
<th>NAT Source</th>
<th>NAT Dest</th>
<th>State</th>
<th>Direction</th>
<th>Frame Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP</td>
<td>10.0.0.4:7398 -&gt; 128.0.0.4:554</td>
<td>Watch</td>
<td>I</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>NAT Source</td>
<td>10.0.0.4:7398 -&gt; 129.0.0.1:1708</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Softwire</td>
<td>2001:0:0:1::1 -&gt; 2002:0:0:2::1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UDP</td>
<td>10.0.0.4:18054 -&gt; 128.0.0.4:12470</td>
<td>Forward</td>
<td>I</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>NAT Source</td>
<td>10.0.0.4:18054 -&gt; 129.0.0.1:1710</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Softwire</td>
<td>2001:0:0:1::1 -&gt; 2002:0:0:2::1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UDP</td>
<td>10.0.0.4:18055 -&gt; 128.0.0.4:12471</td>
<td>Forward</td>
<td>I</td>
<td>23</td>
<td></td>
</tr>
<tr>
<td>NAT Source</td>
<td>10.0.0.4:18055 -&gt; 129.0.0.1:1711</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Softwire</td>
<td>2001:0:0:1::1 -&gt; 2002:0:0:2::1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TCP</td>
<td>128.0.0.4:554 -&gt; 129.0.0.1:1708</td>
<td>Watch</td>
<td>O</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>NAT Dest</td>
<td>129.0.0.1:1708</td>
<td>10.0.0.4:7398</td>
<td>Softwire</td>
<td>2001:0:0:1::1</td>
<td>2002:0:0:2::1</td>
</tr>
<tr>
<td>UDP</td>
<td>128.0.0.4:12470 -&gt; 129.0.0.1:1710</td>
<td>Forward</td>
<td>O</td>
<td>5827</td>
<td></td>
</tr>
<tr>
<td>NAT Dest</td>
<td>129.0.0.1:1710</td>
<td>10.0.0.4:18054</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
FTP ALG (Passive and Active)

This section illustrates FTP ALG commands with the DS-Lite solution.

regress@cypher> show services stateful-firewall conversations application-protocol ftp

Interface: sp-5/0/0, Service set: sset2
Conversation: ALG, Protocol: ftp

Number of initiators: 2, Number of responders: 2

<table>
<thead>
<tr>
<th>Flow</th>
<th>Source Address</th>
<th>Destination Address</th>
<th>State</th>
<th>Direction</th>
<th>Frame Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP</td>
<td>10.0.0.2:16175</td>
<td>128.0.0.3:21</td>
<td>Watch</td>
<td>I</td>
<td>15</td>
</tr>
<tr>
<td>NAT Source</td>
<td>10.0.0.2:16175</td>
<td>129.0.0.1:1491</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Software</td>
<td>2001:0:0:1::1</td>
<td>2002:0:0:2::1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TCP</td>
<td>10.0.0.2:16176</td>
<td>128.0.0.3:16186</td>
<td>Forward</td>
<td>I</td>
<td>5</td>
</tr>
<tr>
<td>NAT Source</td>
<td>10.0.0.2:16176</td>
<td>129.0.0.1:1495</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Software</td>
<td>2001:0:0:1::1</td>
<td>2002:0:0:2::1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TCP</td>
<td>128.0.0.3:21</td>
<td>129.0.0.1:1491</td>
<td>Watch</td>
<td>O</td>
<td>12</td>
</tr>
<tr>
<td>NAT Dest</td>
<td>129.0.0.1:1491</td>
<td>10.0.0.2:16175</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Software</td>
<td>2001:0:0:1::1</td>
<td>2002:0:0:2::1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TCP</td>
<td>128.0.0.3:16186</td>
<td>129.0.0.1:1495</td>
<td>Forward</td>
<td>O</td>
<td>12</td>
</tr>
<tr>
<td>NAT Dest</td>
<td>129.0.0.1:1495</td>
<td>10.0.0.2:16176</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Software</td>
<td>2001:0:0:1::1</td>
<td>2002:0:0:2::1</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Interface: sp-5/0/0, Service set: sset3
Conversation: ALG, Protocol: ftp

Number of initiators: 2, Number of responders: 2

<table>
<thead>
<tr>
<th>Flow</th>
<th>Source Address</th>
<th>Destination Address</th>
<th>State</th>
<th>Direction</th>
<th>Frame Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP</td>
<td>10.0.3.2:16162</td>
<td>128.0.0.3:21</td>
<td>Watch</td>
<td>I</td>
<td>17</td>
</tr>
<tr>
<td>NAT Source</td>
<td>10.0.3.2:16162</td>
<td>130.0.0.1:1540</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Software</td>
<td>2001:0:0:3::1</td>
<td>2002:0:0:3::1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TCP</td>
<td>10.0.3.2:16163</td>
<td>128.0.0.3:16156</td>
<td>Forward</td>
<td>I</td>
<td>11</td>
</tr>
<tr>
<td>NAT Source</td>
<td>10.0.3.2:16163</td>
<td>130.0.0.1:1542</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Software</td>
<td>2001:0:0:3::1</td>
<td>2002:0:0:3::1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TCP</td>
<td>128.0.0.3:21</td>
<td>130.0.0.1:1540</td>
<td>Watch</td>
<td>O</td>
<td>12</td>
</tr>
<tr>
<td>NAT Dest</td>
<td>130.0.0.1:1540</td>
<td>10.0.3.2:16162</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Software</td>
<td>2001:0:0:3::1</td>
<td>2002:0:0:3::1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TCP</td>
<td>128.0.0.3:16156</td>
<td>130.0.0.1:1542</td>
<td>Forward</td>
<td>O</td>
<td>11</td>
</tr>
<tr>
<td>NAT Dest</td>
<td>130.0.0.1:1542</td>
<td>10.0.3.2:16163</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Software</td>
<td>2001:0:0:3::1</td>
<td>2002:0:0:3::1</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Softwires without NAT

Some business ISP providers can give public IP addresses to their customers, using softwires as a means to transition to IPv6 and not only an IPv4 address depletion mitigation technology. In this case, no NAT is needed at the end of the tunnel. This can be achieved with the configuration of softwire concentrators with next-hop style service sets. The example below also uses RSP (redundant interfaces) for a feature rich example.

```plaintext
egress@clownfish1# show interfaces

ge-0/2/5 {
  flexible-vlan-tagging;
  unit 0 {
    vlan-id 100;
    family inet {
      address 100.100.1.1/16;
    }
    family inet6 {
      address 8000::1/126;
    }
  }
}
ge-0/2/6 {
  flexible-vlan-tagging;
  unit 0 {
    vlan-id 100;
    family inet {
      address 110.110.1.1/24;
    }
    family inet6;
  }
  unit 1 {
    vlan-id 101;
    family inet {
      address 110.110.2.1/24;
    }
  }
}
rsp0 {
  traceoptions {
    flag all;
  }
  redundancy-options {
    primary sp-1/0/0;
    secondary sp-1/1/0;
  }
  services-options {
    syslog {
      host local {
        services any;
      }
    }
  }
  unit 3 {
    family inet;
    family inet6;
    service-domain inside;
  }
  unit 4 {
```
family inet;
family inet6;
service-domain outside;
}

{master}[edit]

regress@clownfish1# show services
service-set ss1 {
  syslog {
    host local {
      services any;
      log-prefix DSLITE_ss0;
    }
  }
  softwire-rules ss1_sc_rule_0;
  stateful-firewall-rules ss1_sfw_rule_0;
  next-hop-service {
    inside-service-interface rsp0.3;
    outside-service-interface rsp0.4;
  }
}

stateful-firewall {
  rule ss1_sfw_rule_0 {
    match-direction input-output;
    term 1 {
      from {
        applications [ junos-ftp junos-http junos-https junos-tftp junos-
        rtsp junos-icmp-all junos-icmp-ping junos-sip ];
      }
      then {
        accept;
        syslog;
      }
    }
  }
}

softwire {
  softwire-concentrator {
    ds-lite ss1_sc_name_0 {
      softwire-address abcd::1;
    }
  }
}

rule ss1_sc_rule_0 {
  match-direction input;
  term 1 {
    then {
      ds-lite ss1_sc_name_0;
    }
  }
}

{master}[edit]
regress@clownfish1# show routing-instances
ssl_rinst0 {
  instance-type vrf;
  interface ge-0/2/5.0;
Redundancy and Load Balancing Using Anycast

An IPv6 Anycast address is like any unicast address that is assigned to more than one interface (typically belonging to different nodes), with the property that a packet sent to an Anycast address is routed using the preferred route from the routing table. This is commonly used to load balance between geographically dispersed servers.

In case of DS-Lite, redundancy and load balancing can be provided using the same Anycast address for multiple AFTRs. This can also be achieved between multiple services PICs in one AFTR using the Anycast address for the softwire address.
Multiple AFTR Setup

Figure 4: DS-Lite deployment with multiple AFTRs

In the setup that is shown in Figure 4, we configure the same Anycast address on two or more AFTRs (softwire concentrators) as the softwire address. B4s only need to know this Anycast address for the softwire endpoint, and the least cost AFTR per the routing updates, will be used for the other softwire endpoint. If the least cost AFTR goes down or the cost to get to this AFTR becomes higher than another AFTR, packets will be redirected to the other AFTR. This is automatically taken care by routing updates in the IPv6 cloud. We can also configure different NAT pools at AFTRs and provide continuous service between IPv4 nodes in different domains.

Configuration

Figure 5: Anycast setup for DS-Lite

In the setup shown in Figure 5, Corleone-linux1 is B4 and Tennis and Handball are AFTRs. Softwire is created between Corleone-linux1, and Anycast address b001::1/128. B001::1/128 is configured on both AFTRs (Tennis and Handball). Initially only one of the links (let’s say through Tennis) will be used for traffic between Kay and Fredo. Bring down the interface on Tennis, and traffic will continue to flow through Handball.
Kay Configuration

This is usually the IPv4 node/host in home networks. We will configure the IPv4 interface to ISP and a static route to Fredo. Usually it will be the default route pointing to the customer premises equipment (CPE).

```
regress@kay# show interfaces ge-0/1/0
description "to corleone-linux1";
unit 0 {
    family inet {
        address 11.11.1.1/24;
    }
}
regress@kay# run show route 88.88.1.2
inet.0: 33 destinations, 33 routes (32 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both
88.88.1.0/24 *[Static/5] lw0d 17:54:09
   > to 11.11.1.2 via ge-0/1/0.0
```

Corleone-linux1 Configuration

This is CPE with both IPv4 and IPv6 stacks. It acts as the B4 device for DS-Lite purposes. Configure the IPv4 interface, IPv6 interface, and IPv4-in-v6 tunnel to the Anycast address.

```
[root@corleone-linux1 ~]# ip route get 88.88.1.2
88.88.1.2 dev dsltun3 src 10.209.15.39
    cache mtu 1460 advmss 1420 hoplimit 64
[root@corleone-linux1 ~]# ip -6 route get b001::1
b001::1 from :: via abcd::30 dev eth2 src abcd::29 metric 1024 mtu 1500 advmss 1440 hoplimit 0
[root@corleone-linux1 ~]# history | grep "add dsltun3"
231  ip -6 tunnel add dsltun3 mode ipip6 remote b001::1 local abcd::29 dev eth2
```

Vito Configuration

This is a pure IPv6 node in a V6 cloud. Configure IPv6 interfaces and OSPFv3 for route updates.

```
regress@vito# show interfaces fe-1/2/0
description "to corleone-linux1";
unit 0 {
    family inet6 {
        address abcd::30/120;
    }
}
regress@vito# show interfaces ge-2/0/0
description "V6 cloud to Tennis";
unit 0 {
    family inet6 {
        address 8001::1/120;
    }
}
regress@vito# show interfaces ge-2/0/1
description "v6 cloud to handball";
```
Enable ospf3 for route advertisements.
regress@vito# show protocols
ospf3 {
  area 0.0.0.0 {
    interface ge-2/0/1.0;
    interface ge-2/0/0.0;
    interface fe-1/2/0.0 {
      passive;
    }
  }
}

regress@vito# run show route table inet6 b001::1

inet6.0: 13 destinations, 15 routes (13 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

b001::1/128          *[OSPF/10] 00:21:45, metric 1
                      > to fe80::8271:1fff:fe10:4e0f via ge-2/0/0.0
                      to fe80::8271:1fff:fe11:171d via ge-2/0/1.0

**Tennis Configuration**

This is the AFTR with an IPv6 interface to the ISP network (V6 cloud) and an IPv4 interface to the Internet. Configure IPv4, V6 interfaces, softwire endpoint, and NAT.

regress@tennis# show interfaces
sp-1/1/0 {
  unit 0 {
    family inet;
    family inet6;
  }
}
ge-2/1/0 {
  description “V6 cloud to vito ge-2/0/0”;
  unit 0 {
    family inet;
    family inet6 {
      service {
        input {
          service-set dsl-ss;
        }
        output {
          service-set dsl-ss;
        }
      }
      address 8001::2/120;
    }
  }
}
ge-2/1/6 {
    description "tofredo ge-0/0/0";
    unit 0 {
        family inet {
            address 88.88.1.1/24;
        }
    }
}
lo0 {
    unit 0 {
        family inet;
        family inet6 {
            address b001::1/128;
        }
    }
}
regress@tennis# show services
service-set dsl-ss {
    softwire-rules dsl-sw;
    nat-rules dsl-nat1;
    interface-service {
        service-interface sp-1/1/0;
    }
}
softwire {
    softwire-concentrator {
        ds-lite dsl1 {
            softwire-address b001::1;
            mtu-v6 9192;
        }
    }
    rule dsl-sw {
        match-direction input;
        term t1 {
            then {
                ds-lite dsl1;
            }
        }
    }
}
nat {
    pool dsl-p1 {
        address 7.7.7.0/24;
    }
    rule dsl-nat1 {
        match-direction input;
        term t1 {
            from {
                source-address {
                    11.11.1.0/24;
                }
            }
            then {
                translated {
                    source-pool dsl-p1;
                    translation-type {
source dynamic;
syslog;
}
}

global-config:

regress@tennis# show protocols
ospf3 {
area 0.0.0.0 {
  interface lo0.0;
  interface ge-2/1/0.0;
}
}

Handball Configuration
This is AFTR with an IPv6 interface to the ISP network (V6 cloud) and an IPv4 interface to the Internet. Configure IPv4, IPv6 interfaces, softwire endpoint, and NAT.

regress@handball# show interfaces sp-1/1/0
unit 0 {
  family inet;
  family inet6;
}

regress@handball# show interfaces ge-2/3/0
description “to fredo ge-0/1/0”;
unit 0 {
  family inet {
    address 89.89.1.1/24;
  }
}

regress@handball# show interfaces ge-2/3/4
description “V6 cloud to vito”;
unit 0 {
  family inet;
  family inet6 {
    service {
      input {
        service-set dsl-ss;
      }
      output {
        service-set dsl-ss;
      }
    }
    address 9001::2/120;
  }
}

regress@handball# show interfaces lo0
unit 0 {
  family inet;
  family inet6 {
    address b001::1/128;
  }
}

regress@handball# show protocols
ospf3 {

area 0.0.0.0 {
    interface ge-2/3/4.0;
    interface lo0.0;
}
}
regress@handball# show routing-options
static {
    route 88.88.1.0/24 next-hop 89.89.1.2;
}

Fredo Configuration
This is an IPv4 node on the Internet. Configure the IPv4 interface and routes needed for reverse traffic.

regress@fredo# show interfaces
ge-0/0/0 {
    description “to tennis ge-2/1/6”;
    unit 0 {
        family inet {
            address 88.88.1.2/24;
        }
    }
}
ge-0/1/0 {
    description “to handball ge-2/3/0”;
    unit 0 {
        family inet {
            address 89.89.1.2/24;
        }
    }
}
regress@fredo# show routing-options
static {
    route 7.7.7.0/24 next-hop 88.88.1.1;
    route 8.8.8.0/24 next-hop 89.89.1.1;
}

Traffic Test

Ping from Kay to Fredo
regress@kay# run ping 88.88.1.2 count 10

PING 88.88.1.2 (88.88.1.2): 56 data bytes
... 88.88.1.2 ping statistics ...
10 packets transmitted, 10 packets received, 0% packet loss
round-trip min/avg/max/stddev = 1.370/1.518/1.839/0.132 ms

Verify Flows on Tennis and Handball
regress@handball# run show services stateful-firewall flows
regress@tennis# run show services stateful-firewall flows
Interface: sp-1/1/0, Service set: dal-ss
Flow
State Direction Frame Count
ICMP 11.11.1.1 -> 88.88.1.2 Watch I 10
NAT Source 11.11.1.1 -> 7.7.7.1
Softwire abcd::29 -> b001::1
Shut down ge-2/1/0 on Tennis

regress@tennis# deactivate interfaces ge-2/1/0

[edit]
regress@tennis# commit
commit complete
regress@vito# run show route table inet6 b001::1

inet6.0: 13 destinations, 15 routes (13 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
b001::1/128 *[OSPF/10] 00:00:01, metric 1
> to fe80::8271:1fff:fe11:171d via ge-2/0/1.0

Ping and Verify Flows

regress@kay# run ping 88.88.1.2 count 10
PING 88.88.1.2 (88.88.1.2): 56 data bytes
64 bytes from 88.88.1.2: icmp_seq=0 ttl=63 time=1.814 ms
... 88.88.1.2 ping statistics ---
10 packets transmitted, 10 packets received, 0% packet loss
round-trip min/avg/max/stddev = 1.367/1.527/1.814/0.114 ms

regress@handball# run show services stateful-firewall flows
Interface: sp-1/1/0, Service set: dsl-ss

<table>
<thead>
<tr>
<th>Flow</th>
<th>State</th>
<th>Direction</th>
<th>Frame Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICMP</td>
<td>Watch</td>
<td>O</td>
<td>10</td>
</tr>
<tr>
<td>Nat Source</td>
<td>Watch</td>
<td>I</td>
<td>10</td>
</tr>
<tr>
<td>Softwire</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ICMP</td>
<td>Watch</td>
<td>O</td>
<td>10</td>
</tr>
<tr>
<td>Nat Dest</td>
<td>Watch</td>
<td>I</td>
<td>10</td>
</tr>
<tr>
<td>Softwire</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DS-LITE</td>
<td>Forward</td>
<td>I</td>
<td>20</td>
</tr>
</tbody>
</table>

[edit]
regress@tennis# run show services stateful-firewall flows

[edit]
Multiple services PICs on One AFTR

In Figure 6, three B4s are connected to AFTR’s softwire id 1001::1 using different tunnels. AFTR has two services for load balancing and redundancy. When HTTP clients connect to the server, we can see traffic being load-balanced between the services PICs, and also when one of the services PICs is down, all three of them go through the same services PIC.

AFTR Configuration

Configure next-hop style service sets dslite-svc-set1 and dslite-svc-set2:

```
root@sledding# show services service-set dslite-svc-set1
software-rules dslite-rule;
stateful-firewall-rules sfw-r1;
nat-rules dslite-nat-rule1;
next-hop-service {
    inside-service-interface sp-0/1/0.1;
    outside-service-interface sp-0/1/0.2;
}
```

```
root@sledding# show services service-set dslite-svc-set2
software-rules dslite-rule;
stateful-firewall-rules sfw-r1;
nat-rules dslite-nat-rule2;
next-hop-service {
    inside-service-interface sp-0/0/0.1;
    outside-service-interface sp-0/0/0.2;
}
```

Configure stateful firewall and softwire rules:

```
root@sledding# show services stateful-firewall
rule sfw-r1 {
    match-direction input-output;
    term t1 {
        from {
            applications junos-http;
        }
        then {
            accept;
        }
    }
}
```

```
root@sledding# show services softwire
softwire-concentrator {
```
ds-lite ds1 {
    softwire-address 1001::1;
    mtu-v6 9192;
}

rule ds-lite-rule {
    match-direction input;
    term t1 {
        then {
            ds-lite ds1;
        }
    }
}

Configure two different NAT pools and NAPT:

root@sledding# show services nat
pool ds-lite-pool1 {
    address-range low 33.33.33.1 high 33.33.33.32;
    port {
        automatic;
    }
}
pool ds-lite-pool2 {
    address-range low 44.44.44.1 high 44.44.44.32;
    port {
        automatic;
    }
}
rule ds-lite-nat-rule1 {
    match-direction input;
    term t1 {
        from {
            source-address {
                20.20.0.0/16;
            }
        }
        then {
            translated {
                source-pool ds-lite-pool1;
                translation-type {
                    source dynamic;
                }
            }
        }
    }
}
rule ds-lite-nat-rule2 {
    match-direction input;
    term t1 {
        from {
            source-address {
                20.20.0.0/16;
            }
        }
        then {
            translated {
                source-pool ds-lite-pool2;
            }
        }
    }
}
Configure interfaces:

```bash
root@sledding# show interfaces
sp-0/0/0 {
    unit 0 {
        family inet;
        family inet6;
    }
    unit 1 {
        family inet;
        family inet6;
        service-domain inside;
    }
    unit 2 {
        family inet;
        family inet6;
        service-domain outside;
    }
}
sp-0/1/0 {
    services-options {
        syslog {
            host local {
                services any;
            }
        }
    }
    unit 0 {
        family inet;
        family inet6;
    }
    unit 1 {
        family inet;
        family inet6;
        service-domain inside;
    }
    unit 2 {
        family inet;
        family inet6;
        service-domain outside;
    }
}
ge-3/1/0 {
    unit 0 {
        family inet {
            address 200.200.200.1/24;
        }
    }
}
ge-3/1/5 {
```
unit 0 {
    family inet;
    family inet6 {
        address 2001::1/16;
    }
}

Configure routing options to install route to Anycast address pointing at both services PICs with higher priority:

    root@sledding# show routing-options
    rib inet6.0 {
        static {
            route 1001::1/128 {
                next-hop [ sp-0/1/0.1 sp-0/0/0.1 ];
                preference 0;
            }
        }
    }

Configure load balancing options for Packet Forwarding Engine (PFE):

    root@sledding# show routing-options
    forwarding-table {
        export load-balancing-policy;
    }
    root@sledding# show policy-options
    policy-statement load-balancing-policy {
        then {
            load-balance per-packet;
        }
    }
    root@sledding# show forwarding-options
    hash-key {
        family inet6 {
            layer-3 {
                destination-address;
                next-header;
                source-address;
            }
        }
    }
Verify Load Balancing

Send traffic from all three clients. Client 2’s traffic is going through services PIC sp-0/0/0, and the other two clients’ traffic is going through services PIC sp-0/1/0.

root@sledding> show services stateful-firewall conversations brief

Interface: sp-0/0/0, Service set: dslite-svc-set2

Conversation: ALG, Protocol: tcp

Number of initiators: 1, Number of responders: 1

<table>
<thead>
<tr>
<th>Flow</th>
<th>State</th>
<th>Direction</th>
<th>Frame Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NAT Source</td>
<td>20.20.1.3:1025</td>
<td>200.200.200.2:80</td>
<td>Forward I 664568</td>
</tr>
<tr>
<td>Software</td>
<td>2001::3</td>
<td>1001::1</td>
<td></td>
</tr>
<tr>
<td>TCP</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NAT Dest</td>
<td>44.44.44.1:1024</td>
<td>20.20.1.3:1025</td>
<td>Forward O 1514200</td>
</tr>
<tr>
<td>Software</td>
<td>2001::3</td>
<td>1001::1</td>
<td></td>
</tr>
</tbody>
</table>

Interface: sp-0/1/0, Service set: dslite-svc-set1

Conversation: ALG, Protocol: tcp

Number of initiators: 1, Number of responders: 1

<table>
<thead>
<tr>
<th>Flow</th>
<th>State</th>
<th>Direction</th>
<th>Frame Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NAT Source</td>
<td>20.20.1.2:1025</td>
<td>200.200.200.2:80</td>
<td>Forward I 660553</td>
</tr>
<tr>
<td>Software</td>
<td>2001::2</td>
<td>1001::1</td>
<td></td>
</tr>
<tr>
<td>TCP</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NAT Dest</td>
<td>33.33.33.1:1024</td>
<td>20.20.1.2:1025</td>
<td>Forward O 1520337</td>
</tr>
<tr>
<td>Software</td>
<td>2001::2</td>
<td>1001::1</td>
<td></td>
</tr>
</tbody>
</table>

Conversation: ALG protocol tcp

Number of initiators: 1, Number of responders: 1

<table>
<thead>
<tr>
<th>Flow</th>
<th>State</th>
<th>Direction</th>
<th>Frame Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NAT Source</td>
<td>20.20.1.4:1025</td>
<td>200.200.200.2:80</td>
<td>Forward I 662114</td>
</tr>
<tr>
<td>Software</td>
<td>2001::4</td>
<td>1001::1</td>
<td></td>
</tr>
<tr>
<td>TCP</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NAT Dest</td>
<td>33.33.33.1:1024</td>
<td>20.20.1.4:1025</td>
<td>Forward O 1522189</td>
</tr>
<tr>
<td>Software</td>
<td>2001::4</td>
<td>1001::1</td>
<td></td>
</tr>
</tbody>
</table>
root@sledding> show services stateful-firewall flows count
Interface   Service set                                              Flow count
sp-0/0/0    dslite-svc-set2                                                 3
sp-0/1/0    dslite-svc-set1                                                   6

root@sledding> show services stateful-firewall statistics
Interface   Service set          Accept      Discard       Reject       Errors
sp-0/0/0    dslite-svc-set2     118991296            0            0            0
sp-0/1/0    dslite-svc-set1     237615050            0            0            0

**Verify Redundancy**

Bring down services PIC sp-0/0/0. Client 2's traffic switches to services PIC sp-0/1/0 for any new flows.

root@sledding> request chassis pic fpc-slot 0 pic-slot 0 offline
fpc 0 pic 0 offline initiated, use “show chassis fpc pic-status” to verify

root@sledding> show services stateful-firewall conversations brief
Interface:  sp-0/1/0, Service set:  dslite-svc-set1
Conversation:  ALG, Protocol:  tcp
Number of initiators:  1, Number of responders:  1

<table>
<thead>
<tr>
<th>Flow</th>
<th>State</th>
<th>Direction</th>
<th>Frame Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP 20.20.1.2:1025 -&gt; 200.200.200.2:80</td>
<td>Forward</td>
<td>I</td>
<td>127553</td>
</tr>
<tr>
<td>NAT Source 20.20.1.2:1025 -&gt; 33.33.33.1:1030</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Softwire 2001::2 -&gt; 1001::1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TCP 200.200.200.2:80 -&gt; 33.33.33.1:1030</td>
<td>Forward</td>
<td>O</td>
<td>296667</td>
</tr>
<tr>
<td>NAT Dest 33.33.33.1:1030 -&gt; 20.20.1.2:1025</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Softwire 2001::2 -&gt; 1001::1</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Conversation:  ALG, Protocol:  tcp
Number of initiators:  1, Number of responders:  1

<table>
<thead>
<tr>
<th>Flow</th>
<th>State</th>
<th>Direction</th>
<th>Frame Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP 20.20.1.3:1025 -&gt; 200.200.200.2:80</td>
<td>Forward</td>
<td>I</td>
<td>129493</td>
</tr>
<tr>
<td>NAT Source 20.20.1.3:1025 -&gt; 33.33.33.1:1029</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Softwire 2001::3 -&gt; 1001::1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TCP 200.200.200.2:80 -&gt; 33.33.33.1:1029</td>
<td>Forward</td>
<td>O</td>
<td>298710</td>
</tr>
<tr>
<td>NAT Dest 33.33.33.1:1029 -&gt; 20.20.1.3:1025</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Softwire 2001::3 -&gt; 1001::1</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Conversation: ALG, Protocol: tcp

Number of initiators: 1, Number of responders: 1

<table>
<thead>
<tr>
<th>Flow</th>
<th>Source Address:Port</th>
<th>Destination Address:Port</th>
<th>State</th>
<th>Direction</th>
<th>Frame Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP</td>
<td>20.20.1.4:1025</td>
<td>200.200.200.2:80</td>
<td>NAT Source</td>
<td>Forward</td>
<td></td>
</tr>
<tr>
<td></td>
<td>20.20.1.4:1025</td>
<td>33.33.33.1:1031</td>
<td>Softwire</td>
<td>I</td>
<td>127170</td>
</tr>
<tr>
<td>TCP</td>
<td>200.200.200.2:80</td>
<td>33.33.33.1:1031</td>
<td>NAT Dest</td>
<td>Forward</td>
<td>298831</td>
</tr>
<tr>
<td></td>
<td>33.33.33.1:1031</td>
<td>20.20.1.4:1025</td>
<td>Softwire</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

root@sledding> show services stateful-firewall statistics

Interface   Service set          Accept      Discard       Reject       Errors
sp-0/1/0    dslite-svc-set1       3591368            0            0                 0

root@sledding> show services stateful-firewall flows count

Interface   Service set                                              Flow count
sp-0/1/0    dslite-svc-set1                                                   9

Bibliography


[IANA-Allocation] http://www.iana.org/assignments/ipv4-address-space/ipv4-address-space.xhtml
About Juniper Networks

Juniper Networks is in the business of network innovation. From devices to data centers, from consumers to cloud providers, Juniper Networks delivers the software, silicon and systems that transform the experience and economics of networking. The company serves customers and partners worldwide. Additional information can be found at [www.juniper.net](http://www.juniper.net).