Abstract

This document specifies a new Remote Authentication Dial-In User Service (RADIUS) attribute to carry the Multicast-Prefixes-64 information, aiming to delivery the Multicast and Unicast IPv6 Prefixes to be used to build multicast and unicast IPv4-Embedded IPv6 addresses. This RADIUS attribute is defined based on the equivalent DHCPv6 OPTION_v6_PREFIX64 option.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on December 5, 2015.

Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents

Table of Contents

1. Introduction .................................................. 3
2. Convention and Terminology .................................. 4
3. Multicast-Prefixes-64 Configuration with RADIUS and DHCPv6 .. 5
4. RADIUS Attribute ................................................ 8
   4.1. Multicast-Prefixes-64 ...................................... 8
5. Table of Attributes ............................................. 11
6. Security Considerations ....................................... 12
7. IANA Considerations .......................................... 13
8. Acknowledgments ................................................ 14
9. Normative References .......................................... 15
Authors’ Addresses ................................................. 16
1. Introduction

The solution specified in [I-D.ietf-softwire-dslite-multicast] relies on stateless functions to graft part of the IPv6 multicast distribution tree and IPv4 multicast distribution tree, also uses IPv4-in-IPv6 encapsulation scheme to deliver IPv4 multicast traffic over an IPv6 multicast-enabled network to IPv4 receivers.

To inform the mB4 element of the PREFIX64, a PREFIX64 option may be used. [I-D.ietf-softwire-multicast-prefix-option] defines a DHCPv6 PREFIX64 option to convey the IPv6 prefixes to be used for constructing IPv4-embedded IPv6 addresses.

In broadband environments, a customer profile may be managed by Authentication, Authorization, and Accounting (AAA) servers, together with AAA for users. The Remote Authentication Dial-In User Service (RADIUS) protocol [RFC2865] is usually used by AAA servers to communicate with network elements. Since the Multicast-Prefixes-64 information can be stored in AAA servers and the client configuration is mainly provided through DHCP running between the NAS and the requesting clients, a new RADIUS attribute is needed to send Multicast-Prefixes-64 information from the AAA server to the NAS.

This document defines a new RADIUS attribute to be used for carrying the Multicast-Prefixes-64, based on the equivalent DHCPv6 option already specified in [I-D.ietf-softwire-multicast-prefix-option].

This document makes use of the same terminology defined in [I-D.ietf-softwire-dslite-multicast].

This attribute can be in particular used in the context of DS-Lite Multicast, MAP-E Multicast and other IPv4-IPv6 Multicast techniques. However it is not limited to DS-Lite Multicast.

DS-Lite unicast RADIUS extensions are defined in [RFC6519].
2. Convention and Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

The terms DS-Lite multicast Basic Bridging BroadBand element (mB4) and the DS-Lite multicast Address Family Transition Router element (mAFTR) are defined in [I-D.ietf-softwire-dslite-multicast].
3. Multicast-Prefixes-64 Configuration with RADIUS and DHCPv6

Figure 1 illustrates in DS-Lite scenario how the RADIUS protocol and DHCPv6 work together to accomplish Multicast-Prefixes-64 configuration on the mB4 element for multicast service when an IP session is used to provide connectivity to the user.

The NAS operates as a client of RADIUS and as a DHCP Server/Relay for mB4. When the mB4 sends a DHCPv6 Solicit message to NAS (DHCP Server/Relay), the NAS sends a RADIUS Access-Request message to the RADIUS server, requesting authentication. Once the RADIUS server receives the request, it validates the sending client, and if the request is approved, the AAA server replies with an Access-Accept message including a list of attribute-value pairs that describe the parameters to be used for this session. This list MAY contain the Multicast-Prefixes-64 attribute (asm-length, ASM_PREFIX64, ssm-length, SSM_PREFIX64, unicast-length, U_PREFIX64). Then, when the NAS receives the DHCPv6 Request message containing the OPTION_V6_PREFIX64 option in its Option Request option, the NAS SHALL use the prefixes returned in the RADIUS Multicast-Prefixes-64 attribute to populate the DHCPv6 OPTION_V6_PREFIX64 option in the DHCPv6 reply message.

NAS MAY be configured to return the configured Multicast-Prefixes-64 by the AAA Server to any requesting client without relaying each received request to the AAA Server.

Figure 1: RADIUS and DHCPv6 Message Flow for an IP Session
Figure 2 describes another scenario, which accomplish DS-Lite Multicast-Prefixes-64 configuration on the mB4 element for multicast service when a PPP session is used to provide connectivity to the user. Once the NAS obtains the Multicast-Prefixes-64 attribute from the AAA server through the RADIUS protocol, the NAS MUST store the received Multicast-Prefixes-64 locally. When a user is online and sends a DHCPv6 Request message containing the OPTION_V6_PREFIX64 option in its Option Request option, the NAS retrieves the previously stored Multicast-Prefixes-64 and uses it as OPTION_V6_PREFIX64 option in DHCPv6 Reply message.

![Figure 2: RADIUS and DHCPv6 Message Flow for a PPP Session](image)

According to [RFC3315], after receiving the Multicast-Prefixes-64 attribute in the initial Access-Accept packet, the NAS MUST store the received V6_PREFIX64 locally. When the mB4 sends a DHCPv6 Renew message to request an extension of the lifetimes for the assigned address or prefix, the NAS does not have to initiate a new Access-
Request packet towards the AAA server to request the Multicast-Prefixes-64. The NAS retrieves the previously stored Multicast-Prefixes-64 and uses it in its reply.

Also, if the DHCPv6 server to which the DHCPv6 Renew message was sent at time T1 has not responded, the DHCPv6 client initiates a Rebind/Reply message exchange with any available server. In this scenario, the NAS receiving the DHCPv6 Rebind message MUST initiate a new Access-Request message towards the AAA server. The NAS MAY include the Multicast-Prefixes-64 attribute in its Access-Request message.
4. RADIUS Attribute

This section specifies the format of the new RADIUS attribute.

4.1. Multicast-Prefixes-64

The Multicast-Prefixes-64 attribute conveys the IPv6 prefixes to be used in [I-D.ietf-softwire-dslite-multicast] to synthesize IPv4-embedded IPv6 addresses. The NAS SHALL use the IPv6 prefixes returned in the RADIUS Multicast-Prefixes-64 attribute to populate the DHCPv6 PREFIX64 Option [I-D.ietf-softwire-multicast-prefix-option].

This attribute MAY be used in Access-Request packets as a hint to the RADIUS server, for example, if the NAS is pre-configured with Multicast-Prefixes-64, these prefixes MAY be inserted in the attribute. The RADIUS server MAY ignore the hint sent by the NAS, and it MAY assign a different Multicast-Prefixes-64 attribute.

If the NAS includes the Multicast-Prefixes-64 attribute, but the AAA server does not recognize this attribute, this attribute MUST be ignored by the AAA server.

NAS MAY be configured with both ASM_PREFIX64 and SSM_PREFIX64 or only one of them. Concretely, AAA server MAY return ASM_PREFIX64 or SSM_PREFIX64 based on the user profile and service policies. AAA MAY return both ASM_PREFIX64 and SSM_PREFIX64. When SSM_PREFIX64 is returned by the AAA server, U_PREFIX64 MUST also be returned by the AAA server.

If the NAS does not receive the Multicast-Prefixes-64 attribute in the Access-Accept message, it MAY fall back to a pre-configured default Multicast-Prefixes-64, if any. If the NAS does not have any pre-configured, the delivery of multicast traffic is not supported.

If the NAS is pre-provisioned with a default Multicast-Prefixes-64 and the Multicast-Prefixes-64 received in the Access-Accept message are different from the configured default, then the Multicast-Prefixes-64 attribute received in the Access-Accept message MUST be used for the session.

A summary of the Multicast-Prefixes-64 RADIUS attribute format is shown Figure 3. The fields are transmitted from left to right.
Figure 3: RADIUS attribute format for Multicast-Prefixes-64

Type:

145 for Multicast-Prefixes-64

Length:

This field indicates the total length in octets of this attribute including the Type and Length fields, and the length in octets of all PREFIX fields.

asm-length:

the prefix-length for the ASM IPv4-embedded prefix, as an 8-bit unsigned integer (0 to 128). This field represents the number of valid leading bits in the prefix.

ASM_PREFIX64:

this field identifies the IPv6 multicast prefix to be used to synthesize the IPv4-embedded IPv6 addresses of the multicast groups in the ASM mode. It is a variable size field with the length of the field defined by the asm-length field and is rounded up to the nearest octet boundary. In such case any additional padding bits must be zeroed. The conveyed multicast IPv6 prefix MUST belong to the ASM range. This prefix is likely to be a /96.

ssm-length:

the prefix-length for the SSM IPv4-embedded prefix, as an 8-bit unsigned integer (0 to 128). This field represents the number of valid leading bits in the prefix.

SSM_PREFIX64:

this field identifies the IPv6 multicast prefix to be used to synthesize the IPv4-embedded IPv6 addresses of the multicast groups in the SSM mode. It is a variable size field with the length of the field defined by the ssm-length field and is rounded up to the nearest octet boundary. In such case any additional padding bits must be zeroed. The conveyed multicast IPv6 prefix MUST belong to the SSM range. This prefix is likely to be a /96.

unicast-length:

the prefix-length for the IPv6 unicast prefix to be used to synthesize the IPv6 addresses of the multicast sources, as an 8-bit unsigned integer (0 to 128). This field represents the number of valid leading bits in the prefix.

U_PREFIX64:

this field identifies the IPv6 unicast prefix to be used in SSM mode for constructing the IPv4-embedded IPv6 addresses representing the IPv4 multicast sources in the IPv6 domain. U_PREFIX64 may also be used to extract the IPv4 address from the received multicast data flows. It is a variable size field with the length of the field defined by the unicast-length field and is rounded up to the nearest octet boundary. In such case any additional padding bits must be zeroed. The address mapping MUST follow the guidelines documented in [RFC6052].
5. Table of Attributes

The following tables provide a guide to which attributes may be found in which kinds of packets, and in what quantity.

The following table defines the meaning of the above table entries.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Access-</th>
<th>Access-</th>
<th>Access-</th>
<th>Challenge</th>
<th>Accounting-</th>
<th>#</th>
<th>Request</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Request</td>
<td>Accept</td>
<td>Reject</td>
<td>Request</td>
<td></td>
<td>0-1</td>
<td></td>
</tr>
<tr>
<td>Multicast-Prefixes-64</td>
<td>0-1</td>
<td>0</td>
<td>0</td>
<td>0-1</td>
<td>145</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CoA-</td>
<td>Request</td>
<td>ACK</td>
<td>NACK</td>
<td></td>
<td></td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>CoA-</td>
<td></td>
<td></td>
<td></td>
<td>145</td>
<td>Multicast-Prefixes-64</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

0+ This attribute MUST NOT be present in the packet.
0-1 Zero or one instances of this attribute MAY be present in the packet.
1 Exactly one instances of this attribute MAY be present in the packet.
6. Security Considerations

This document has no additional security considerations beyond those already identified in [RFC2865] for the RADIUS protocol and in [RFC5176] for CoA messages.

The security considerations documented in [RFC3315] and [RFC6052] are to be considered.
7. IANA Considerations

Per this document, IANA has allocated a new RADIUS attribute type from the IANA registry "Radius Attribute Types" located at http://www.iana.org/assignments/radius-types.

Multicast-Prefixes-64 - 145
8. Acknowledgments

The authors would like to thank Ian Farrer, Chongfen Xie, Qi Sun, Linhui Sun and Hao Wang for their contributions to this work.
9. Normative References

[I-D.ietf-softwire-dslite-multicast]

[I-D.ietf-softwire-multicast-prefix-option]


Authors’ Addresses

Qian Wang
China Telecom
No.118, Xizhimennei
Beijing  100035
China

Email: wangqian@ctbri.com.cn

Wei Meng
ZTE Corporation
No.50 Software Avenue, Yuhuatai District
Nanjing
China

Email: meng.wei2@zte.com.cn,vally.meng@gmail.com

Cui Wang
ZTE Corporation
No.50 Software Avenue, Yuhuatai District
Nanjing
China

Email: wang.cuil@zte.com.cn

Mohamed Boucadair
France Telecom
Rennes, 35000
France

Email: mohamed.boucadair@orange.com
Abstract

This document describes when and how an operator uses the technique of Mapping of Address and Port (MAP) for the IPv4 residual deployment in the IPv6-dominant domain.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on December 11, 2015.

Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents before using this document.
Table of Contents

1. Introduction ................................................. 3
2. Conventions ................................................. 4
3. Case Studies ............................................... 5
4. Deployment Consideration ................................. 7
   4.1. Building the MAP Domain ............................... 7
       4.1.1. MAP Deployment Model Planning .................. 7
       4.1.2. MAP Domain Planning ............................. 8
       4.1.3. MAP Rule Provisioning ........................... 8
       4.1.4. MAP DHCPv6 server deployment consideration .... 9
       4.1.5. PSID Consideration ............................... 10
       4.1.6. Addressing and Routing ........................... 10
       4.1.7. MAP vs. MAP-T vs. 4rd ........................... 11
       4.2. BR Settings ......................................... 12
       4.3. CE Settings ......................................... 15
       4.4. Supporting System ................................... 15
5. MAP Address Planning ....................................... 17
   5.1. Planning for Residual Deployment, a Step-by-step Guide . 17
   5.2. Remarks on Deployment Paradigms ...................... 19
6. Migration Methodology ...................................... 21
   6.1. Roadmap for MAP-based Solution ....................... 21
       6.1.1. Start from Scratch ............................... 21
       6.1.2. Coexiting Phases ................................. 21
       6.1.3. Exit Strategy .................................... 21
       6.2. Migration Mode ....................................... 22
           6.2.1. Passive Transition ............................ 22
           6.2.2. Active Transition ............................. 22
7. IANA Considerations ...................................... 23
8. Security Considerations .................................. 24
9. Contributors .............................................. 25
10. Acknowledgements ....................................... 26
11. References ............................................... 27
   11.1. Normative References ............................... 27
   11.2. Informative References .............................. 27
Authors’ Addresses .......................................... 29
1. Introduction

IPv4 address exhaustion has become world-wide reality and the primary solution in the industry is to deploy IPv6-only networking. Meanwhile, having access to legacy IPv4 contents and services is a long-term requirement, will be so until the completion of the IPv6 transition. It demands sharing residual IPv4 address pools for IPv4 communications across the IPv6-only domain(s).

Mapping of Address and Port (MAP) [I-D.ietf-softwire-map] is designed in response to the requirement of stateless residual deployment. The term "residual deployment" refers to utilizing IPv4 addresses for IPv4 communications going across the IPv6 domain backbone. MAP assumes the IPv6-only backbone as the prerequisite of deployment so that native IPv6 services and applications are fully supported and encouraged. The statelessness of MAP ensures only moderate overhead is added to part of the network devices.

Residual deployment with MAP is new to most operators. This document is motivated to provide basic understanding on the usage of MAP, i.e., when and how an operator can do with MAP to meet its own operational requirements of IPv6 transition and its facility conditions, in the phase of IPv4 residual deployment. Potential readers of this document are those who want to know:

1. What are the requirements of MAP deployment?
2. What technical options needs to be considered when deploying MAP, and how?
3. How does MAP impact on the address planning for both IPv6 and IPv4 pools?
4. How does MAP impact on daily network operations and administrations?
5. How do we migrate to IPv6-only network with the help of MAP?

Terminology of this document, unless it is intentionally specified, follows the definitions and abbreviations of [I-D.ietf-softwire-map].

Unless it is specifically specified, the deployment considerations and guidance proposed in this document are also applied to MAP-T [I-D.ietf-softwire-map-t], the translation variation of MAP, and 4rd [I-D.ietf-softwire-4rd], the reversible translation approach that aims to improve end-to-end consistency of double translation.
2. Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].
3. Case Studies

MAP can be deployed for large-scale carrier networks. There are typically two network models for broadband access service: one is to use PPPoE/PPPoA authentication method while the other is to use IPoE. The first one is usually applied to Residential network and SOHO networks. Subscribers in CPNs can access broadband network by PPP dial-up authentication. BRAS is the key network element which takes full responsibility of IP address assignment, user authentication, traffic aggregation, PPP session termination, etc. Then IP traffic is forwarded to Core Routers through Metro Area Network, and finally transited to Internet via Backbone network. The second network scenario is usually applied to large enterprise networks. Subscribers in CPNs can access broadband network by IPoE authentication. IP address is normally assigned by DHCP server, or static configuration.

In either case, a Customer Edge Router (CER) could obtain a prefix via prefix delegation procedure, and the hosts behind CER would get its own IPv6 addresses within the prefix through SLAAC or DHCPv6 statefully. A MAP CE would also obtain a set of MAP rules from DHCPv6 server.

Figure 1 depicts a generic model of stateless IPv4-over-IPv6 communication for broadband access services.

Figure 1: Stateless IPv4-over-IPv6 broadband access network architecture

When deploying MAP in home network, there can be two architecture: A. single ISP B. multihoming with two or more ISPs, sharing one CE. In the single ISP model, CE needs to communicate with only one MAP BR,
while in multihoming model CE has to communicate with multiple MAP BRs. Figure 2 [RFC7368] illustrates a typical case, where the home network has multiple connections to multiple providers or multiple logical connections to the same provider. In the multihoming model, a CE will be provisioned with multiple BMRs. Routing information will also be configured for multihoming; but detail of the routing configuration is out of the scope of this memo.

Figure 2: MAP multihoming
4. Deployment Consideration

4.1. Building the MAP Domain

When deploying stateless MAP in an operational network, a provider should firstly do MAP domain planning based on that existing network. According to the definition of [I-D.ietf-softwire-map], a MAP domain is a set of MAP CEs and BRs connected to the same virtual link. All CEs in the MAP domain are provisioned with the same set of MAP rules by MAP DHCPv6 server [I-D.ietf-softwire-map-dhcp]. There might be multiple BMRs in one MAP domain, e.g. in case of multi-ISP. A CE may be provisioned with multiple IPv6 prefix, which can be used to find the corresponding BMR via longest prefix match. As defined in [I-D.ietf-softwire-map-dhcp], a BMR should be provisioned together with a BR IPv6 address; the CE should maintain this binding, so that the mapping between BMR and BR is achieved which is useful in multi-ISP scenario. In mesh mode, a longest-matching prefix lookup is done in the IPv4 routing table and the correct FMR is chosen.

Basically, operator should firstly determine its own deployment topology for MAP domain as described in Section 4.1.1, as different considerations apply for different deployment models. Next, MAP domain planning, MAP rule provision, addressing and routing, etc., for a MAP domain should be taken into consideration, as discussed in the sections following Section 4.1.1.

For the scenario where one CE is corresponding with multiple MAP border relays, it is possible that those MAP BRs belong to different MAP domains. The CE must pick up its own MAP rules and domain parameters in each domain. This is a typical case of multihoming. The MAP rules must have the information about BR(s) and information about the service types and the ISP.

4.1.1. MAP Deployment Model Planning

In order to do MAP domain planning, an operator should firstly make the decision to choose mesh or hub and spoke topology according to the operator’s network policy. In the hub and spoke topology, all traffic within the same MAP domain has to go through the BR, resulting in less optimal traffic flow; however, it simplifies CE processing since there is no need to do FMR lookup for each incoming packet. Moreover, it provides enhanced manageability as the BR can take full control of all the traffic. As a result, it is reasonable to deploy hub and spoke topology for a network with a relatively flat architecture.

In mesh topology, CE to CE traffic flows are optimized since they pass directly between the two nodes. Mesh topology is recommended
when CE to CE traffic is high and there are not too many MAP rules, say fewer than 10 MAP rules, in the given domain.

4.1.2. MAP Domain Planning

Stateless MAP offers advantages in terms of scalability, high reliability, etc. As a result, it is reasonable to plan for a larger MAP domain to accommodate more subscribers with fewer BRs. Moreover, a larger MAP domain will also be easier for management and maintenance. However, a larger MAP domain may also result in less optimized traffic in the hub and spoke case, where all traffic has to go through a remote BR. In addition, it may result in an increased number of MAP rules and highly centralized address management. Choosing appropriate domain coverage requires the evaluation of tradeoffs.

When multiple IPv4 subnets are deployed in one MAP domain, it is recommended to further divide the MAP domain into multiple subdomains, each with only one IPv4 subnet. This can simplify the MAP domain planning. But there can be a side effect that it will increase the traffic between BRs. Different subdomains could be distinguished by different Rule IPv4 prefixes. As stated previously, all CEs within the same MAP subdomain will have the same Rule IPv4 prefix, Rule IPv6 prefix and PSID parameters.

4.1.3. MAP Rule Provisioning

In stateless MAP, Mesh or Hub and Spoke communications can be achieved among CEs in one MAP domain in terms of assigning appropriate FMR(s) to CEs. We recommend ISP deploy the full Hub and Spoke topology or full mesh topology describe below to simplify the configuration of the DHCPv6 server.

4.1.3.1. Full Hub and Spoke Communication among CEs

In order to achieve the full communication in the Hub and Spoke topology, no FMR is assigned to CEs. In this topology, when a CE sends packets to another CE in the same MAP domain via BR, or using the DMR as FMR, the packets must go through BR before arriving at the destination. DMR is specific for MAP-T only.

4.1.3.2. Full Mesh Communication among CEs

By assigning all BMRs in MAP domain to each CE as FMRs, Mesh communications can be achieved among all CEs. In this case, when CE receives an IPv4 packet, it looks up for an appropriate FMR with a specific Rule IPv4 prefix which has the longest match with the IPv4 destination address.
4.1.3.3. Mesh or Hub/Spoke communication among some CEs

Mesh communications among some CEs along with Hub/Spoke communications among some other CEs can be achieved by which differentiated FMRs are assigned to CEs. For instance, as shown in Figure 3, since both CE1 and CE2 has rule 1 and rule2, the communication between CE1 and CE2 can go directly without going though associated BR (Mesh topology). However, for CE1 and CE3, since there are no rule for each other, the communication between CE1 and CE3 must go though BR before reaching peer each other (Hub/Spoke topology).

![Figure 3:]

<table>
<thead>
<tr>
<th>BMR</th>
<th>CE1</th>
<th>CE2</th>
<th>CE3</th>
</tr>
</thead>
<tbody>
<tr>
<td>rule 1</td>
<td>rule 2</td>
<td>rule 3</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>FMRs</th>
<th>CE1</th>
<th>CE2</th>
<th>CE3</th>
</tr>
</thead>
<tbody>
<tr>
<td>rule 1</td>
<td>rule 1</td>
<td>rule 2</td>
<td></td>
</tr>
<tr>
<td>rule 2</td>
<td>rule 2</td>
<td>rule 3</td>
<td></td>
</tr>
<tr>
<td>rule 3</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

4.1.4. MAP DHCPv6 server deployment consideration

All the CEs within a MAP domain will get a set of MAP rules by DHCPv6 server. Each Mapping Rule keeps a record of Rule IPv6 prefix, Rule IPv4 prefix and Rule EA-bits length. Section 5 would give a step by step example of how to calculate these parameters.

As the MAP is stateless, the deployment of DHCPv6 server is independent of MAP domain planning. So there are three possible cases:

MAP domain : DHCPv6 server = 1:1 This is the ideal solution that each MAP domain would have its own MAP DHCPv6 server. In this case, MAP DHCPv6 server only needs to configure parameters for the specific MAP domain. In this model, it is easy to achieve the configuration in MAP and no extra configuration requirement is needed.

MAP domain : DHCPv6 server = 1:N This might happen when DHCPv6 servers are deployed in a large MAP domain in a distributed manner. In this case, all these DHCPv6 servers should be configured with the same set of MAP rules for the MAP domain, including multiple BMRs, FMRs and DMRs.
MAP domain : DHCPv6 server = N:1 This might happen when MAP domain is relatively small and a single MAP DHCPv6 server is deployed in the network. In this case, multiple MAP domains should be distinguished based on CE’s IPv6 prefix in different MAP domains.

4.1.5. PSID Consideration

If a provider would like to introduce differentiated address sharing ratios for different CEs, it is better to define multiple MAP sub-domains with different Rule IPv4 prefixes. In this way, MAP domain division is only a logical method, rather than a geographical one.

The default PSID offset (a) is chosen as 6 in [I-D.ietf-softwire-map] and this excludes the system ports (0-1023). For MAP, the initial part of the port number (the a-bits) cannot be zero (see Appendix B of [I-D.ietf-softwire-map].) As is shown in the section 3.2.4 of [I-D.tsou-softwire-port-set-algorithms-analysis], it is possible that a lower value of ‘a’ will give a higher sharing ratio and more than 1024 ports are excluded as a result, e.g. ‘a’ = 4 will exclude ports 0 - 4095. The value of ‘a’ should be made explicitly configurable by operators.

With regard to PSID format, both continuous and non-continuous port set can be supported in GMA algorithm. Non-continuous port set has the advantage of better UPnP friendly, while continuous port set is the simplest way to implement. Since PSID format should be supported not only in CPEs, BRs and DHCPv6 server, but also in other sustaining systems as well, e.g. traffic logging system, user management system, a provider should make the decision based on a comprehensive investigation on its demand and the capabilities of existing equipments.

Note that some ISPs may need to offer services in a MAP domain with a shared address, e.g. there are hosts FTP server under CEs. The service provisioning may require well-know port range (i.e. port range belong to 0-1023). MAP would provide operators with an option to generate a port range including those in 0-1023. Afterwards, operators could decide to assign it to any requesting user. However, if the port-set is too small, it is not suggested to assign one with only the port set 0-1023 or even less. Considerable non-well-known ports are surely needed. Another easier approach is assigning a dedicated IPv4 address to such a CE if the demand really exists.

4.1.6. Addressing and Routing

In MAP addressing, it should follow the MAP rule planning in the MAP domain.
For IPv4 addressing, since the number of scattered IPv4 address prefixes would be equal to the number of FMR rules within a MAP domain, one should choose as large IPv4 address pool as possible to reduce the number of FMR rules. For IPv6 addressing, the Rule IPv6 prefixes should be equal to the end user IPv6 prefix in MAP domain.

If ISP has a /24 rule IPv4 prefix with sharing ratio of 64 gives 16000 customers, and a /16 rule IPv4 prefix supports 4 million customer. If up the sharing ratio to 256, 64000 and 16 million customers can be supports respectively. For the ISP who has scattered IPv4 address prefixes, in order to reduce the number of FMRs, according to needs of ports they can divide different classes. For instance, for the enterprise customers class which need many ports to use, provision them the EMR with low sharing ratio while for the private customers class which don’t need so many ports provision them the BMR with high sharing ratio.

For MAP routing, there are no IPv4 routes exported to IPv6 networks.

4.1.7. MAP vs. MAP-T vs. 4rd

Basically, encapsulation provides an architectural building block of virtual link where the underlay behavior is fully hidden, while translation does a delivery participating into the end-to-end transferring path where behaviors are exposed. It is reflected in the following aspects.

1. Option header

If translation or 4rd ‘reversible translation’ is applied, IPv4 options at the IP layer are not translated according to [RFC791][RFC2460], and packets with those options MUST be dropped by Domain-entry nodes, and return ICMPv4 error messages to signal IPv4-option incompatibility. This limitation is acceptable because there are a lot firewalls in current IPv4 Internet also filter IPv4 packets.

2. ICMP

Some IPv4 ICMP codes do not have a corresponding codes in ICMPv6, a detailed analysis on the double translation behavior suggest that some ICMPv4 messages, when they are translated to ICMPv6 and back to ICMPv4 across the IPv6 domain, the accuracy might be sacrificed to some extent. Encapsulation keeps the full transparency of ICMPv4 messages.

Reversible translation approach of 4rd, however, does not translate ICMPv4 messages into ICMPv6 version. Instead, it treats ICMP as same as a transport layer protocol data unit. This behavior is similar to
the encapsulation and keeps ICMP end-to-end transparency as well.

In either the encapsulation or translation mode, if an intermediate node generates an ICMPv6 error message, it should be converted into ICMPv4 version and returned to the source with a special source address and following the behavior specified in [RFC6791]. However, the behavior and semantics of the translation from ICMPv6 to ICMPv4 is different among encapsulation, translation and 4rd reversible translation approaches. Encapsulation treats routing error in the IPv6 domain as an (virtual)link error between the tunnel end points, while translation translate IPv6 routing error into corresponding IPv4 version, and 4rd, however, behaves according to whether the Tunnel Traffic Class option is set. The TTL behavior also reflect the differences among different approaches, which is worth paying attention to for the operating engineers. MAP-T translator is compatible with single translation approach.

3. PMTU and fragmentation

Both translation mode and encapsulation mode have PMTU and fragmentation problem. [RFC6145] discusses the problem in details for the translation, while [RFC2473] could be a reference on the issue in encapsulation.

4.2. BR Settings

1. BR placement

BR placement has important impacts on the operation of a MAP domain.

A first concern should be the avoidance of "triangle routing". In hub and spoke mode, all traffic will be routed through BR which may increase the path from the CE to an IPv4 peer. This can be accomplished easily by placing the BR close to the CE, such that the length of the path from the CE to the BR is minimized.

However, minimizing the CE-BR path would ignore a second concern, that of minimizing IPv4 operations. An ISP deploying MAP will probably want to focus on IPv6 operations, while keeping IPv4 operational expenditures to a minimum. This would imply that the size of the IPv4 network that the ISP has to administer would be kept to a minimum. Placing the BR near the CE means that the length of the IPv4 network between the BR and the IPv4 Internet would be longer.

Moreover, in case where the set of CEs is geographically dispersed, multiple BRs would be needed, which would further enlarge the IPv4 network that the ISP has to maintain.
Therefore, we offer the following guideline: BRs should be placed as close to the border with the IPv4 Internet as possible while keeping triangle routing to a minimum. Regional POPs should probably be considered as potential candidates.

Note also that MAP being stateless, asymmetric routing to/from the IPv4 Internet is natively supported and therefore no path-pinning mechanisms have to be additionally implemented.

Anycast can be used to let the network pick BR closest to a CE for traffic exiting the MAP domain. This is accomplished by provisioning a Default Mapping Rule containing an anycast IPv6 address or prefix. Operationally, this allows incremental deployment of BRs in strategic locations without modifying the provisioning system’s configuration. CE’s close to a newly-deployed BR will automatically start using it. The BR MUST participate in a dynamic IGP so that this can work automatically.

2. Reliability Considerations

Reliability of MAP is derived in major part from its statelessness. This means that MAP can benefit from the usual methods of Internet reliability.

Anycast, already mentioned in section 4.2.1, can be used to ensure reliability of traffic from CE to BR. Since there can be only one Default Mapping Rule per MAP domain, traffic from CE to BR will always use the same destination address. When this address is anycast, reliability is greatly increased. If a BR goes down, it stops advertising the IPv6 anycast address, and traffic is automatically re-routed to other BRs; the BR should also withdraw the routes for traffic from BR to CE, or the upstream routers connected to the BR should dynamically change the routes when it detects the failure of a BR, otherwise there will be a routing blackhole. For this mechanism to work correctly, it is crucial that the anycast route announcement be very closely tied to BR availability. See [RFC4786] for best current practices on the operation of anycast services. In practice, Equal-cost multi-path (ECMP) can be used to achieve active/active configuration. Operator can also increase the metric for one BR to have active/standby.

For reliability within a single link can be achieved with the help of a redundancy protocol such as VRRP [RFC5798]. This allows operation of a pair of BRs in active/standby configuration. No state needs to be shared for the operation of MAP, so there is no need to keep the standby node in a "warm" state: as long as it is up and ready to take over the virtual IPv6 address, quick failover can be achieved. This makes the pair behave as a single, much more reliable node, with less
reliance on quick routing protocol convergence for reliability.

It is expected that production-quality MAP deployments will make use of both anycast and a redundancy protocol such as VRRP.

3. MTU/Fragmentation

If the MTU is well-managed such that the IPv6 MTU on the CE WAN side interface is set so that no fragmentation occurs within the boundary of the MAP domain, then the Tunnel MTU can be set to the known IPv6 MTU minus the size of the encapsulating IPv6 header (40 bytes). For example, if the IPv6 MTU is known to be 1500 bytes, the Tunnel MTU might be set to 1460 bytes. Without more specific information, the Tunnel MTU SHOULD default to 1240 bytes.

BRs using an anycast address as source can cause problems. If traffic sent by a BR with a source anycast address causes an ICMP error to be returned, that error packet’s destination address will be an anycast address, meaning that a different BR might receive it. In the case of a Too Big ICMP error, this could cause a path MTU discovery black hole. Another possible problem could occur if fragmented packets from different BRs using the same anycast address as source happen to contain the same fragment ID. This would break fragment reassembly. Since there is still no simple way to solve it completely, it is recommended to increase the MTU of the IPv6 network so that no fragmentation and Too Big ICMP error occurs.

In MAP domains where IPv4 addresses are not shared, IPv6 destinations are derived from IPv4 addresses alone. Thus, each IPv4 packet can be encapsulated and decapsulated independently of each other. The processing is completely stateless.

On the other hand, in MAP domains where IPv4 addresses are shared, BRs and CEs may have to encapsulate or translate IPv4 packets whose IPv6 destinations depend on destination ports. Precautions are needed, due to the fact that the destination port of a fragmented datagram is available only in its first fragment. A sufficient precaution consists in reassembling each datagram received in multiple packets, and to treat it as though it would have been received in single packet. This function is such that MAP is in this case stateful at the IP layer. (This is common with DS-lite and NAT64/DNS64 which, in addition, are stateful at the transport layer.) At domain entrance, this ensures that all pieces of all received IPv4 datagrams go to the right IPv6 destinations.
4.3. CE Settings

1. bridging vs. routing

In routing manner, the CE runs a standard NAT44 [RFC3022] using the allocated public address as external IP and ports via DHCPv6 option. When receiving an IPv4 packet with private source address from its end hosts, it performs NAT44 function by translating the source address into public and selecting a port from the allocated port-set. Then it encapsulates/translation (depending on whether MAP-E or MAP-T is in use) the packet with the concentrator’s IPv6 address as destination IPv6 address, and forwards it to the concentrator. When receiving an IPv6 packet from the concentrator, the initiator decapsulates/translation the IPv6 packet to get the IPv4 packet with public destination IPv4 address. Then it performs NAPT44 function and translates the destination address into private one based on the entry in NAT state table in the CE.

The CE is responsible for performing ALG functions (e.g., SIP, FTP), as well as supporting NAT Traversal mechanisms (e.g., UPnP, NAT-PMP, manual mapping configuration). This is no different from the standard IPv4 NAT today.

For the bridging manner, end host would run a software performing CE functionalities. In this case, end host gets public address directly. It is also suggested that the host run a local NAT to map randomly generated ports into the restricted, valid port-set. Another solution is to have the IP stack to only assign ports within the restricted, valid range to applications. Either way the host guarantees that every source port number in the outgoing packets falls into the allocated port-set.

2. CE-initiated application

CE-initiated case is applied for situations where applications run on CE directly. If the application in CE use the public address directly, it might conflict with other CEs. So it is highly suggested that CE should also run a local NAT to map a private address to public address in CE. In this way, the CE IPv4 address passed to local applications would be conflict with other CEs.

4.4. Supporting System

1. Lawful Intercept

Sharing IPv4 addresses among multiple CEs is susceptible to issues related to lawful intercept. For details, see [RFC6269] section 12.
2. Traffic Logging

It is always possible for a service provider that operates a MAP domain to determine the IPv6 prefix associated with a MAP IPv4 address (and port number in case of a shared address). This mapping is static, and it is therefore unnecessary to log every IPv4 address assignment. However, changes in that static mapping, such as rule changes in the provisioning system, need to be logged in order to be able to know the mapping at any point in time.

Sharing IPv4 addresses among multiple CEs is susceptible to issues related to traffic logging. For details, see [RFC6269] sections 8 and 13.1.

3. Geo-location aware service

Sharing IPv4 addresses among multiple CEs is susceptible to issues related to geo-location. For details, see [RFC6269] section 7.

4. User Management

MAP IPv4 address assignment, and hence the IPv4 service itself, is tied to the IPv6 prefix lease; thus, the MAP service is also tied to this in terms of authorization, accounting, etc. For example, the MAP address has the same lifetime as its associated IPv6 prefix.
5. MAP Address Planning

This section is purposed to provide a referential guidance to operators, illustrating a common method of address planning with MAP in IPv4 residual deployment.

5.1. Planning for Residual Deployment, a Step-by-step Guide

Residual deployment starts from IPv6 address planning.

(A) IPv6 considerations

(A1) Determine the maximum number N of CEs to be supported, and, for generality, suppose $N = 2^n$.

For example, we suppose $n = 20$. It means there will be up to about one million CEs.

(A2) Choose the length $x$ of IPv6 prefixes to be assigned to ordinary customers.

Consider we have a /32 IPv6 block, it is not a problem for the IPv6 deployment with the given number of CEs. Let $x = 60$, allowing subnets inside in each CE delegated networks.

(A3) Multiply $N$ by a margin coefficient $K$, a power of two ($K = 2^k$), to take into account that:

- Some privileged customers may be assigned IPv6 prefixes of length $x'$, shorter than $x$, to have larger addressing spaces than ordinary customers, both in IPv6 and IPv4;

- Due to the hierarchy of routable prefixes, many theoretically delegable prefixes may not be actually delegable (ref: host density ratio of [RFC3194]).

In our example, let’s take $k = 0$ for simplicity.

(B) IPv4 considerations

(B1) List all (non overlapping, not yet assigned to any in-running networks) IPv4 prefixes $\{H_i\}$ that are available for IPv4 residual deployment.

Suppose that we hold two blocks and not yet assigned to any fixed network: 192.0.2.0/24 and 198.51.100.0/24.
Take enough of them, among the shortest ones, to get a total whose size $M$ is a power of two ($M = 2^m$), and includes a good proportion of the available IPv4 space.

If we use both blocks, $M = 2^{24} + 2^{24}$, and therefore $m = 25$. Suppose the intended sharing ratio is 8 subscribers per address, resulting in $(65536 - 1024)/8 = 8064$ ports per subscriber assuming that the well-known ports are excluded. Then the PSID length to achieve this will be $\log_2(8) = 3$ bits. Bearing in mind the IPv4 24 bit prefix length for each of our two prefixes, the EA-bit length is $(32 - 24) + 3 = 11$ bits.

For each IPv4 prefix, $H_i$, of length $h_i$, choose an prefix extension, say $R_i$ of length $r_i = m - (32 - h_i)$.

All these indexes must be non overlapping prefixes (e.g. 0, 10, 110, 111 for one /10, one /11, and two /12). In our example, we pick 0 for a contiguous address block while 1 for another.

Then we have:

- $H_1 = 192.0.2.0/24, h_1 = 24, r_1 = 17 => R_1 = \text{bin}(0)$;
- $H_2 = 198.51.100.0/24, h_2 = 24, r_2 = 17 => R_2 = \text{bin}(1)$;

Sometimes the IPv4 residual pool is not well aggregated and the contiguous address blocks may have different sizes. For example, in (B1), if we have $H_1 = 59.112.0.0/13$ and $H_2 = 219.120.0.0/16$ as the IPv4 residual pool, then $M = 2^{19} + 2^{16}$, and in such a case, we must pick $m$ so that $m = \text{ceil}(\log_2(M))$, where "ceil(x)" means the minimum integer not less than x, i.e., $m = 20$ in this case. Therefore $r_1 = 20 - (32 - 13) = 1$, while $r_2 = 20 - (32 - 16) = 4$. Several combinations are available for the $R_1$ and $R_2$ and one only needs to pay attention to avoiding overlapping when picking up the values.

After (A) and (B), derive the rule(s)

- Derive the length $c$ of the MAP domain IPv6 prefix, $C$, that will appear at the beginning of all delegated prefixes ($c = x - (n + k)$).
- Take any prefix for this $C$ of length $c$ that starts with a RIR-allocated IPv6 prefix.
- For each IPv4 prefix $H_i$, make the rule, in which the key is $H_i$ and the value is the domain IPv6 prefix $C$ followed by the rule index $R_i$. Then this $i$-th rule’s Rule IPv6 Prefix will have the length of $(c + r_i)$. 

Then we can do that:

\[ c = 40 \Rightarrow C = 2001:0db8:ff00::/40 \]

Rule 1: Rule IPv6 Prefix = 2001:0db8:ff00::/41

Rule 2: Rule IPv6 Prefix = 2001:0db8:ff80::/41

If we have different lengths for the Rule IPv4 prefix (as the extra example discussed at the end of (B)), their Rule IPv6 prefixes should not have the same length, as their rule index length is different.

As a result, for a certain CE delegating 2001:0db8:ff98:7650::/60, its parameters are:

Rule IPv6 Prefix = 2001:0db8:ff80::/41 => Rule 2
IPv4 Suffix = bin(111 0110 0)
PSID = bin(101) = 0x5
Rule IPv4 Prefix = 198.51.100.0/24
CE IPv4 Address = 198.51.100.236

If different sharing ratio is demanded, we may partition CEs into groups and do (A) and (B) for each group, determining the PSID length for them separately.

5.2. Remarks on Deployment Paradigms

1. IPv6 address planning in residual deployment is independent of the usage of the residual IPv4 addresses. The IPv4 address pool for "residual deployment" contains IPv4 addresses not yet allocated to customers/subscribers and/or those already recalled from ex-customers, re-programmed into relatively well-aggregated blocks.

2. It is recommended to have the number of rule entries as less as possible so that the merit of stateless deployment is reflected in practical performances. However, this effort is often constrained by the condition of an operator whether (a): it holds large-enough contiguous IPv4 address block(s) for the residual deployment, and (b): a short-enough IPv6 domain prefix so that the /64 delegation is easily satisfied even the EA-bits is quite long. When condition (a) is not satisfied, sub-domains have to be defined for each relatively small but contiguous aggregated block; when condition (b) is not satisfied, one has to divide the IPv4 aggregates into smaller blocks artificially in order to reduce the length of EA-bits. When we have good conditions fitting (a) and (b), it is NOT recommended to define short EA-bits with small length of IPv4 suffix (the value \( p \)) nor to increase the number of rule entries (also the number of sub-
3. An extreme case is, when EA-bits contain the full IPv4 address while a full IPv4 address is assigned to a CE, i.e., o = p = 32, and q = 0, the MAP address format becomes almost equivalent to RFC6052-format [RFC6052] except the off-domain IPv4 peer’s mapped IPv6 address. This frees the domain to distribute rules but the DMR. In such a case, IPv6 addressing is fully dependent of IPv4, which defers from the typical residual deployment case. MAP is mainly designed for residual deployment but also applied for the case of legacy IPv4 networks keeping communication with the IPv4 world over the IPv6 domain without renumbering, as long as the address planning doesn’t matter.

4. Another extreme case is, when EA-bits’ length becomes to zero, i.e., o = p = q = 0, a rule actually defines a correspondence between an IPv6 address and an IPv4 address (or a prefix), without any algorithmic correlation to each other. Using such a case in practice is not prohibited by the specification, but it is not recommended to deploy null EA-bits in large scale as the concern discussed in the above Remark 2, and as it has the limitation that the PSID must be null (q = 0) and therefore multiple CEs sharing a same IPv4 address is not supported here. It is recommended to apply Lightweight 4over6 [I-D.ietf-softwire-lw4over6], if a full de-correlation between IPv6 address and IPv4 address as well as port range is demanded.

5. A not-so-extreme case, p = 0, o = q, i.e., only PSID is applied for the EA-bits, is also a case possibly happening in practice. It also potentially generates a huge number of rules and therefore large-scale deployment of this case is not recommended either.

6. For operators who would like to utilize "some bits" of IPv6 address to do service identification, QoS differentiation, etc., it is recommended that these special-purpose bits should be embedded before the EA-bits so as to reduce the possibility of bit-conflict. However, it requires quite shorter IPv6 aggregate prefix of the operator. The bit-conflict is more likely to happen in this case if different domains have different Rule prefix lengths. Operators with this demand should pay attention to the impact on the domain rule planning.
6. Migration Methodology

6.1. Roadmap for MAP-based Solution

6.1.1. Start from Scratch

IPv6 deployment normally involves a step-wise approach where parts of
the network should properly updated gradually. As IPv6 deployment
progresses it may be simpler for operators to employ a single-version
network, since deploying both IPv4 and IPv6 in parallel would cost
more than IPv6-only network. Therefore switching to an IPv6-only
network in relatively small scale will become more prevalent.
Meanwhile, a significant part of network will still stay in IPv4 for
long time, especially at early stage of IPv6 transition. There may
not be enough public or private IPv4 addresses to support end-to-end
network communication, without segmenting the network into small
parts with sharing one IPv4 address space. That is a time to
introduce MAP to bridge these IPv4 islands through IPv6 network.

6.1.2. Coexisting Phases

SP has various deployment strategy in the middle of transition. It’s
foreseeable that IPv6 would likely coexist with IPv4 in a long
period. The MAP deployment would also fit into the coexisting mode.
To be specific, dual-stack technology is recommended in RFC6180 as
the simplest deployment model to advance IPv6 deployment. MAP
technology could get along well with native IPv6 connections and
compatible with residual IPv4 networks. RFC6264 described a
incremental transition approach in order to migrate networks to IPv6-
only. DS-Lite is treated as a technology to accelerate the whole
process. MAP can also take the same role to achieve a smooth
transition.

6.1.3. Exit Strategy

The benefit of IPv6-only + MAP is that all IPv6 flows would go
directly to the Internet, no need for encapsulation or translation.
In this way, as more content providers and service are available over
IPv6, the utilization on MAP CE and BR goes down since fewer
destinations require MAP progressing. This way would advance IPv6,
because it provides everyone incentives to use IPv6, and eventually
the result is an pure IPv6 network with no need for IPv4. As more
content providers and hosts equipped with IPv6 capabilities, the MAP
utilization goes down until it is eventually not used at all when all
content is IPv6. In this way, MAP has an "exit strategy". The
corresponding solutions will leave the network in time.
6.2. Migration Mode

IPv4 Residual deployment is an interim phase during IPv6 migration. It would be beneficial to ISPs, if this phase is as short as possible since end-to-end IPv6 traversal is the really goal. When IPv6 is getting more and more mature, MAP would be retired in a natural way.

6.2.1. Passive Transition

Passive Transition is following IPv4 retirement law. In another word, MAP would always get along with IPv4, even all nodes is dual-stack capable. At a later stage of IPv6 migration, MAP can also be served for dual-stack hosts, which is sending traffic through the IPv4 stack. There is still a value for this approach because it could steer IPv4 traffic to IPv6 going through a MAP CE processing. When it comes the time ISP decide to turn off IPv4, MAP would be unnecessary due to IPv4 disappearance.

6.2.2. Active Transition

Active Transition is targeting to accelerate IPv4 exit and increase native IPv6 utilization. A desirable way deploying MAP is only providing IPv6 traversal ability to a IPv4-only host. However, MAP CE can not determine received traffic is send from a IPv4 node or a dual-stack node. In the latter case, IPv6 utilization is preferred for the most part. When a network evolves to a post-IPv6 era, it might be good for ISPs to consider to implement enforcement rules to help IPv6 migration.

- ISP could install only IPv6 record (i.e. AAAA) in DNS server, which would provide users with IPv6 steering effects. When a host is IPv6-capable and gets IPv6 DNS reply in advance, MAP functionalities would be restricted by IPv6-only record response.
- ISP could retrieve shared IPv4 address by increasing sharing ratio. In this case, number of concurrent IPv4 sessions on MAP CE would be suppressed. It would encourage native IPv6 growth in some extent.
- ISP could allocate a dedicated IPv6 prefix for MAP deployment. The allocation could not only facilitate the differentiation between MAPed traffic and native IPv6 traffic, but also clearly observe the change of MAP traffic. When the traffic is reducing for a while, ISP could close the MAP functionalities in some specific area. It would result networks to native IPv6-only capable.
7. IANA Considerations

This specification does not require any IANA actions.
8. Security Considerations

There are no new security considerations pertaining to this document.
9. Contributors

The members of the MAP design team are:

Congxiao Bao, Mohamed Boucadair, Gang Chen, Maoke Chen, Wojciech Dec, Xiaohong Deng, Remi Despres, Jouni Korhonen, Xing Li, Satoru Matsushima, Tomasz Mrugalski, Tetsuya Murakami, Jacni Qin, Qiong Sun, Tina Tsou, Dan Wing, Leaf Yeh, and Jan Zorz.

Thanks to Chunfa Sun who was an active co-author of some earlier versions of this draft. Thanks to Shishio Tsuchiya’s valueable suggestion for this document.
10. Acknowledgements

Remi Despres contributed the original example of step-by-step deployment guidance in discussion with the authors. Ole Troan, as the head of MAP Design Team, joined the discussion directly and contributed a lot of ideas and comments. We also thank other members of the MAP Design Team for their comments and suggestions.

Thanks to Tom Talyer, Qi Sun and Ian Farrer for their thorough review and helpful comments.
11. References

11.1. Normative References

[I-D.ietf-softwire-4rd]
Despres, R., Jiang, S., Penno, R., Lee, Y., Chen, G., and
M. Chen, "IPv4 Residual Deployment via IPv6 - a Stateless
Solution (4rd)", draft-ietf-softwire-4rd-10 (work in
progress), December 2014.

[I-D.ietf-softwire-map]
Troan, O., Dec, W., Li, X., Bao, C., Matsushima, S.,
Murakami, T., and T. Taylor, "Mapping of Address and Port
with Encapsulation (MAP)", draft-ietf-softwire-map-13
(work in progress), March 2015.

[I-D.ietf-softwire-map-dhcp]
Mrugalski, T., Troan, O., Farrer, I., Perreault, S., Dec,
W., Bao, C., Yeh, L., and X. Deng, "DHCPv6 Options for
configuration of Softwire Address and Port Mapped
Clients", draft-ietf-softwire-map-dhcp-12 (work in
progress), March 2015.

[I-D.ietf-softwire-map-t]
Li, X., Bao, C., Dec, W., Troan, O., Matsushima, S., and
T. Murakami, "Mapping of Address and Port using
Translation (MAP-T)", draft-ietf-softwire-map-t-08 (work
in progress), December 2014.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate

[RFC6145] Li, X., Bao, C., and F. Baker, "IP/ICMP Translation
Algorithm", RFC 6145, April 2011.

[RFC6791] Li, X., Bao, C., Wing, D., Vaithianathan, R., and G.
Huston, "Stateless Source Address Mapping for ICMPv6

11.2. Informative References

[RFC2473] Conta, A. and S. Deering, "Generic Packet Tunneling in

Address Assignment Efficiency An Update on the H ratio",

Authors’ Addresses

Qiong Sun  
China Telecom  
Room 708 No.118, Xizhimenneidajie  
Beijing, 100035  
P.R.China  
Phone: +86 10 5855 2923  
Email: sunqiong@ctbri.com.cn

Maoke Chen  
BBIX, Inc.  
Tokyo Shiodome Building, Higashi-Shimbashi 1-9-1  
Minato-ku, Tokyo 105-7310  
Japan  
Email: maoke@bbix.net

Gang Chen  
China Mobile  
28 Xuanwumenxi Ave; Xuanwu District  
Beijing  
P.R. China  
Email: chengang@chinamobile.com

Tina Tsou  
Huawei Technologies  
2330 Central Expressway  
Santa Clara, CA 95050  
USA  
Phone: +1-408-330-4424  
Email: tina.tsou.zouting@huawei.com

Simon Perreault  
Jive Communications  
Quebec, QC  
Canada  
Email: sperreault@jive.com

Abstract

Mapping of Address and Port (MAP) is a stateless mechanism for running IPv4 over IPv6-only infrastructure. It provides both IPv4 and IPv6 connectivity services simultaneously during the IPv4/IPv6 co-existing period. The Dynamic Host Configuration Protocol for IPv6 (DHCPv6) MAP options has been defined to configure MAP Customer Edge (CE). However, in many networks, the configuration information may be stored in Authentication Authorization and Accounting (AAA) servers while user configuration is mainly from Broadband Network Gateway (BNG) through DHCPv6 protocol. This document defines a Remote Authentication Dial In User Service (RADIUS) attribute that carries MAP configuration information from AAA server to BNG. The MAP RADIUS attribute are designed following the simplify principle. It provides just enough information to form the correspondent DHCPv6 MAP option.
Table of Contents

1. Introduction ................................................. 3
2. Terminology .................................................. 3
3. MAP Configuration process with RADIUS ....................... 3
4. Attributes .................................................... 6
   4.1. MAP-Configuration Attribute ............................. 6
   4.2. MAP Rule Options ...................................... 6
   4.3. Sub Options for MAP Rule Option ....................... 7
       4.3.1. Rule-IPv6-Prefix Sub Option ..................... 7
       4.3.2. Rule-IPv4-Prefix Sub Option ..................... 8
       4.3.3. EA Length Sub Option ............................ 9
       4.3.4. BR-IPv6-Address Sub Option ...................... 9
       4.3.5. PSID Sub Option ................................ 9
       4.3.6. PSID Length Sub Option .......................... 10
       4.3.7. PSID Offset Sub Option .......................... 10
   4.4. Table of attributes .................................... 11
5. Diameter Considerations ........................................ 11
6. IANA Considerations .......................................... 12
7. Security Considerations ...................................... 12
8. Acknowledgements ............................................ 12
1. Introduction

Recently providers start to deploy IPv6 and consider how to transit to IPv6. Mapping of Address and Port (MAP)[I-D.ietf-softwire-map] is a stateless mechanism for running IPv4 over IPv6-only infrastructure. It provides both IPv4 and IPv6 connectivity services simultaneously during the IPv4/IPv6 co-existing period. MAP has adopted Dynamic Host Configuration Protocol for IPv6 (DHCPv6) [RFC3315] as auto-configuring protocol. The MAP Customer Edge (CE) uses the DHCPv6 extension options [I-D.ietf-softwire-map-dhcp] to discover MAP Border Relay (in tunnel model only) and to configure relevant MAP rules.

In many networks, user configuration information may be stored by AAA (Authentication, Authorization, and Accounting) servers. Current AAA servers communicate using the Remote Authentication Dial In User Service (RADIUS) [RFC2865] protocol. In a fixed line broadband network, the Broadband Network Gateways (BNGs) act as the access gateway of users. The BNGs are assumed to embed a DHCPv6 server function that allows them to locally handle any DHCPv6 requests initiated by hosts.

Since the MAP configuration information is stored in AAA servers and user configuration is mainly transmitted through DHCPv6 protocol between BNGs and hosts/CEs, new RADIUS attributes are needed to propagate the information from AAA servers to BNGs. The MAP RADIUS attributes designed in this document are especially for the MAP encapsulation mode, while providing enough information to form the correspondent DHCPv6 MAP option [I-D.ietf-softwire-map-dhcp].

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

The terms MAP CE and MAP Border Relay are defined in [I-D.ietf-softwire-map].

3. MAP Configuration process with RADIUS

The below Figure 1 illustrates how the RADIUS protocol and DHCPv6 cooperate to provide MAP CE with MAP configuration information.
The BNG acts as a RADIUS client and as a DHCPv6 server. First, the MAP CE MAY initiate a DHCPv6 Solicit message that includes an Option Request option (6) [RFC3315] with the MAP option [I-D.ietf-softwire-map-dhcp] from the MAP CE. But note that the ORO (Option Request option) with the MAP option could be optional if the network was planned as MAP-enabled as default. When BNG receives the SOLICIT, it SHOULD initiates radius Access-Request message, in which the User-Name attribute (1) SHOULD be filled by the MAP CE MAC address or interface-id or both, to the RADIUS server and the User-password attribute (2) SHOULD be filled by the shared MAP password that has been preconfigured on the DHCPv6 server, requesting authentication as defined in [RFC2865] with MAP-Configuration attribute, which will be defined in the next Section. If the authentication request is approved by the AAA server, an Access-Accept message MUST be acknowledged with the IPv6-MAP-Configuration Attribute. After receiving the Access-Accept message with MAP-Configuration Attribute, the BNG SHOULD respond the user an Advertisement message. Then the user can requests for a MAP Option, and the BNG SHOULD reply the user with the message containing the MAP option. The recommended format of the MAC address is defined as Calling-Station-Id (Section 3.20 in [RFC3580] without the SSID (Service Set Identifier) portion.

Figure 2 describes another scenario, in which the authorization operation is not coupled with authentication. Authorization relevant to MAP is done independently after the authentication process. As similar to above scenario, the ORO with the MAP option in the initial DHCPv6 request could be optional if the network was planned as MAP-enabled as default.
In the above mentioned scenario, the Access-Request packet SHOULD contain a Service-Type attribute (6) with the value Authorize Only (17); thus, according to [RFC5080], the Access-Request packet MUST contain a State attribute that obtained from the previous authentication process.

In both above-mentioned scenarios, Message-authenticator (type 80) [RFC2869] SHOULD be used to protect both Access-Request and Access-Accept messages.

After receiving the MAP-Configuration Attribute in the initial Access-Accept, the BNG SHOULD store the received MAP configuration parameters locally. When the MAP CE sends a DHCPv6 Request message to request an extension of the lifetimes for the assigned address, the BNG does not have to initiate a new Access-Request towards the AAA server to request the MAP configuration parameters. The BNG could retrieve the previously stored MAP configuration parameters and use them in its reply.

If the BNG does not receive the MAP-Configuration Attribute in the Access-Accept it MAY fall back to a pre-configured default MAP configuration, if any. If the BNG does not have any pre-configured default MAP configuration or if the BNG receives an Access-Reject, the tunnel cannot be established.

As specified in [RFC3315], section 18.1.4, "Creation and Transmission of Rebind Messages ", if the DHCPv6 server to which the DHCPv6 Renew message was sent at time T1 has not responded by time T2, the MAP CE (DHCPv6 client) SHOULD enter the Rebind state and attempt to contact any available server. In this situation, the secondary BNG receiving the DHCPv6 message MUST initiate a new Access-Request towards the AAA server.
server. The secondary BNG MAY include the MAP-Configuration Attribute in its Access-Request.

4. Attributes

This section defines MAP-Rule Attribute which is used in the MAP scenario. The attribute design follows [RFC6158] and refers to [RFC6929].

The MAP RADIUS attribute are designed following the simplify principle. The sub options are organized into two categories: the necessary and the optional.

4.1. MAP-Configuration Attribute

The MAP-Configuration Attribute is structured as follows:

```
0                   1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Type     |    Length     |                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                               +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Type

TBD

Length

2 + the length of the Rule option(s)

MAP Rule Option(s)

A variable field that may contains one or more Rule option(s), defined in Section 4.2

4.2. MAP Rule Options

Depending on deployment scenario, one Basic Mapping Rule and zero or more Forwarding Mapping Rules MUST be included in one MAP-Configuration Attribute.
### MAP Rule Option

**4.3. Sub Options for MAP Rule Option**

**4.3.1. Rule-IPv6-Prefix Sub Option**

The Rule-IPv6-Prefix Sub Option is necessary for every MAP Rule option. It should appear for once and only once.

The IPv6 Prefix sub option is followed the framed IPv6 prefix designed in [RFC3162].
### Rule-IPv6-Prefix Sub Option

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>+-----------------------------------------------+---------------+-----------+</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SubType</td>
<td>SubLen</td>
<td>Reserved</td>
<td>prefix6-len</td>
</tr>
<tr>
<td>+-----------------------------------------------+---------------+-----------+</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>rule-ipv6-prefix</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**SubType**
- 1 (SubType number, for the Rule-IPv6-Prefix6 sub option)

**SubLen**
- 20 (the length of the Rule-IPv6-Prefix6 sub option)

**Reserved**
- Reserved for future usage. It should be set to all zero

**prefix6-len**
- Length of the IPv6 prefix, specified in the rule-ipv6-prefix field, expressed in bits

**rule-ipv6-prefix**
- A 128-bits field that specifies an IPv6 prefix that appears in a MAP rule

### Rule-IPv4-Prefix Sub Option

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>+-----------------------------------------------+---------------+-----------+</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SubType</td>
<td>SubLen</td>
<td>Reserved</td>
<td>prefix4-len</td>
</tr>
<tr>
<td>+-----------------------------------------------+---------------+-----------+</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>rule-ipv4-prefix</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**SubType**
- 2 (SubType number, for the Rule-IPv4-Prefix6 sub option)

**SubLen**
- 8 (the length of the Rule-IPv4-Prefix6 sub option)

**Reserved**
- Reserved for future usage. It should be set to all zero

**prefix4-len**
- Length of the IPv6 prefix, specified in the rule-ipv4-prefix field, expressed in bits

**rule-ipv4-prefix**
- A 32-bits field that specifies an IPv6 prefix that appears in a MAP rule
4.3.3. EA Length Sub Option

SubType 3 (SubType number, for the EA Length sub option)
SubLen 4 (the length of the EA Length sub option)
EA-len 16 bits long field that specifies the Embedded-Address (EA) bit length. Allowed values range from 0 to 48

4.3.4. BR-IPv6-Address Sub Option

SubType 4 (SubType number, for the BR-ipv6-address sub option)
SubLen 20 (the length of the BR-ipv6-address sub option)
BR-ipv6-address a 128-bits field that specifies the IPv6 address for the BR.

4.3.5. PSID Sub Option
SubType
5 (SubType number, for the PSID Sub Option sub option)
SubLen
4 (the length of the PSID Sub Option sub option)
PSID (Port-set ID)
Explicit 16-bit (unsigned word) PSID value. The PSID value algorithmically identifies a set of ports assigned to a CE. The first k-bits on the left of this 2-octets field is the PSID value. The remaining (16-k) bits on the right are padding zeros.

4.3.6. PSID Length Sub Option

SubType
6 (SubType number, for the PSID Length sub option)
SubLen
4 (the length of the PSID Length sub option)
PSID-len
Bit length value of the number of significant bits in the PSID field. (also known as ‘k’). When set to 0, the PSID field is to be ignored. After the first ‘a’ bits, there are k bits in the port number representing valid of PSID. Subsequently, the address sharing ratio would be 2 ^k.

4.3.7. PSID Offset Sub Option
### SubType
7 (SubType number, for the PSID Offset sub option)

### SubLen
4 (the length of the PSID Offset sub option)

### PSID Offset
4 bits long field that specifies the numeric value for the MAP algorithm’s excluded port range/offset bits (A-bits), as per section 5.1.1 in [I-D.ietf-softwire-map]. Default must be set to 4.

#### 4.4. Table of attributes

The following table provides a guide to which attributes may be found in which kinds of packets, and in what quantity.

<table>
<thead>
<tr>
<th>Request</th>
<th>Accept</th>
<th>Reject</th>
<th>Challenge</th>
<th>Accounting</th>
<th># Attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-1</td>
<td>0-1</td>
<td>0</td>
<td>0</td>
<td>0-1</td>
<td>TBD1 MAP-Configuration</td>
</tr>
<tr>
<td>0-1</td>
<td>0-1</td>
<td>0</td>
<td>0</td>
<td>0-1</td>
<td>1 User-Name</td>
</tr>
<tr>
<td>0-1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0-1</td>
<td>2 User-Password</td>
</tr>
<tr>
<td>0-1</td>
<td>0-1</td>
<td>0</td>
<td>0-1</td>
<td>0-1</td>
<td>5 Service-Type</td>
</tr>
<tr>
<td>0-1</td>
<td>0-1</td>
<td>0-1</td>
<td>0</td>
<td>0-1</td>
<td>80 Message-Authenticator</td>
</tr>
</tbody>
</table>

The following table defines the meaning of the above table entries.

- **0** This attribute MUST NOT be present in packet.
- **0+** Zero or more instances of this attribute MAY be present in packet.
- **0-1** Zero or one instance of this attribute MAY be present in packet.
- **1** Exactly one instance of this attribute MUST be present in packet.

#### 5. Diameter Considerations

This attribute is usable within either RADIUS or Diameter [RFC6733]. Since the Attributes defined in this document will be allocated from the standard RADIUS type space, no special handling is required by Diameter entities.
6. IANA Considerations

This document requires the assignment of two new RADIUS Attributes Types in the "Radius Types" registry (currently located at http://www.iana.org/assignments/radius-types for the following attributes:

- MAP-Configuration TBD1

IANA should allocate the numbers from the standard RADIUS Attributes space using the "IETF Review" policy [RFC5226].

7. Security Considerations

In MAP scenarios, both CE and BNG are within a provider network, which can be considered as a closed network and a lower security threat environment. A similar consideration can be applied to the RADIUS message exchange between BNG and the AAA server.

Known security vulnerabilities of the RADIUS protocol are discussed in [RFC2607], [RFC2865], and [RFC2869]. Use of IPsec [RFC4301] for providing security when RADIUS is carried in IPv6 is discussed in [RFC3162].

A malicious user may use MAC address proofing and/or dictionary attack on the shared MAP password that has been preconfigured on the DHCPv6 server to get unauthorized MAP configuration information.

Security considerations for MAP specific between MAP CE and BNG are discussed in [I-D.ietf-softwire-map]. Furthermore, generic DHCPv6 security mechanisms can be applied DHCPv6 intercommunication between MAP CE and BNG.

Security considerations for the Diameter protocol are discussed in [RFC6733].

8. Acknowledgements

The authors would like to thank the valuable comments made by Peter Lothberg, Wojciech Dec, and Suresh Krishnan for this document.

This document was produced using the xml2rfc tool [RFC2629].

9. References
9.1. Normative References


9.2. Informative References


Internet-Draft                 MAP RADIUS                      June 2015


Authors’ Addresses

Sheng Jiang
Huawei Technologies Co., Ltd
Q14, Huawei Campus, No.156 Beiqing Road
Hai-Dian District, Beijing, 100095
P.R. China

Email: jiangsheng@huawei.com

Yu Fu
CNNIC
No.4 South 4th Street, Zhongguancun
Hai-Dian District, Beijing, 100190
P.R. China

Email: fuyu@cnnic.cn

Bing Liu
Huawei Technologies Co., Ltd
Q14, Huawei Campus, No.156 Beiqing Road
Hai-Dian District, Beijing, 100095
P.R. China

Email: leo.liubing@huawei.com
Unified IPv4-in-IPv6 Softwire CPE
draft-ietf-softwire-unified-cpe-02

Abstract

In IPv6-only provider networks, transporting IPv4 packets encapsulated in IPv6 is a common solution to the problem of IPv4 service continuity. A number of differing functional approaches have been developed for this, each having their own specific characteristics. As these approaches share a similar functional architecture and use the same data plane mechanisms, this memo describes a specification whereby a single CPE can interwork with all of the standardized and proposed approaches to providing encapsulated IPv4 in IPv6 services.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on April 21, 2016.
IPv4 service continuity is one of the major technical challenges which must be considered during IPv6 migration. Over the past few years, a number of different approaches have been developed to assist with this problem. These approaches, referred to as ‘S46 mechanisms’ in this document, exist in order to meet the particular deployment, scaling, addressing and other requirements of different service provider’s networks.

A common feature shared between all of the differing modes is the integration of softwire tunnel end-point functionality into the CPE router. Due to this inherent data plane similarity, a single CPE may be capable of supporting several different approaches. Users may also wish to configure a specific mode of operation.
A service provider’s network may also have more than one S46 mechanism enabled in order to support a diverse CPE population with differing client functionality, such as during a migration between mechanisms or where services require specific supporting softwire architectures.

For softwire based services to be successfully established, it is essential that the customer end-node, the service provider end-node and provisioning systems are able to indicate their capabilities and preferred mode of operation.

A number of DHCPv6 options for the provisioning of softwires have been standardized:

- RFC6333  Defines DHCPv6 option 64 for configuring B4 elements with the IPv6 address of the AFTR.
- RFC7341  Defines DHCPv6 option 88 for configuring the address of a DHCPv4 server, which can then be used by a softwire client for obtaining further configuration.
- RFC7598  Defines DHCPv6 options 94, 95 and 96 for provisioning MAP-E, MAP-T and lw4o6 respectively.

This document describes a DHCPv6 based prioritisation method whereby a CPE which supports several S46 mechanisms and receives configuration for more than one can prioritise which mechanism to use. The method requires no server side logic to be implemented and only uses a simple S46 mechanism prioritization to be implemented in the CPE.

The prioritisation method as described here does not provide redundancy between S46 mechanisms for the client. I.e. If the highest priority S46 mechanism which has been provisioned to the client is not available for any reason, the means for identifying this and falling back to the S46 mechanism with the next highest priority is not in the scope of this document.

1.1. Rationale

The following rationale has been adopted for this document:

1. Simplify solution migration paths: Define unified CPE behavior, allowing for smooth migration between the different S46 mechanisms.
2. Deterministic CPE co-existence behavior: Specify the behavior when several S46 mechanisms co-exist in the CPE.
3. Deterministic service provider co-existence behavior: Specify the behavior when several modes co-exist in the service providers network.
(4) Re-usability: Maximize the re-use of existing functional blocks including tunnel end-points, port restricted NAPT44, forwarding behavior, etc.

(5) Solution agnostic: Adopt neutral terminology and avoid (as far as possible) overloading the document with solution-specific terms.

(6) Flexibility: Allow operators to compile CPE software only for the mode(s) necessary for their chosen deployment context(s).

(7) Simplicity: Provide a model that allows operators to only implement the specific mode(s) that they require without the additional complexity of unneeded modes.

1.2. S46 Priority Option

The S46 Priority Option is used to convey a priority order of IPv4 service continuity mechanisms. Figure 1 shows the format of the S46 Priority option.

```
0                   1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   OPTION_V6_S46_PRIORITY      |         option-length         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        s46-option-code        |  s46-priority   |      option-|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|code |        s46-option-code        |  s46-priority   |       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 1: S46 Priority Option

- option-code: OPTION_V6_S46_PRIORITY (TBD)
- option-length: variable-length
- s46-option-code: 16-bits long IANA registered option code of the DHCPv6 option which is used to identify the softwire mechanism.
- priority: 8-bit integer indicating the priority of the mechanism.

The fields s46_option_code and s46_priority are a key-value pair. Both fields MUST appear together sequentially.

Each defined s46_option_code, and its associated s46_priority field MAY appear exactly once within the list of S46 option codes. The option MUST contain at least two s46_option_code/s46_priority pairs, and MAY contain more as necessary.

DISCUSSION: The prioritisation field is included to offer additional flexibility, making it possible to give the same weight to two or
more mechanisms to instruct the client to configure them simultaneously. If this is not seen to be a requirement (i.e. only one active mechanism is needed), then the prioritisation field could be removed. The prioritisation would then be done as an ordered list of option codes as described in section 17 of RFC7227.

The s46_priority field associated with each s46_option_code is an 8-bit integer, which tells the recipient the priority of the associated S46 mechanism. A higher s46_priority is taken by the client to be a higher priority. If two or more s46_option_codes with the same s46_priority value are received, then the client should attempt to configure these mechanisms simultaneously. As this document is solely concerned with provisioning, issues related to the client’s usage of multiple active softwires are out of scope.

1.3. Client Behavior

Clients MAY request option OPTION_V6_S46_PRIORITY, as defined in [RFC3315], Sections 17.1.1, 18.1.1, 18.1.3, 18.1.4, 18.1.5, and 22.7. As a convenience to the reader, we mention here that the client includes requested option codes in the Option Request Option.

Upon receipt of a DHCPv6 Offer message from the server containing OPTION_V6_S46_PRIORITY the client performs the following steps:

1. Check the contents of the DHCPv6 message for options containing valid S46 mechanism configuration. A candidate list of possible S46 mechanisms is created from these option codes.
2. Check the contents of OPTION_V6_S46_PRIORITY for the DHCPv6 option codes contained in the included s46_option_code fields and their related s46_priorities. From this, an S46 mechanism priority list is created, ordered from highest to lowest s46_priority value.
3. Sequentially check the priority list against the candidate list until a match is found.
4. When a match is found, the client SHOULD configure the resulting S46 mechanism. Configuration for other S46 mechanisms MUST be discarded.

In the event that no match is found between the priority list and the candidate list, the client MAY proceed with configuring one or more of the provisioned S46 softwire mechanism(s). In this case, which mechanism(s) are chosen by the client is implementation specific and not defined here.

In the event that the client receives OPTION_V6_S46_PRIORITY with the following errors, it MUST be discarded:
Less than two instances of s46_option_code/s46_priority key-value pair.

Two s46_option_code fields with the same value.

If an invalid OPTION_V6_S46_PRIORITY option is received, the client MAY proceed with configuring the provisioned S46 mechanisms as if OPTION_V6_S46_PRIORITY had not been received.

In the event that a client receives an OPTION_V6_S46_PRIORITY option containing a value in s46-option-code representing an S46 mechanism which the client has not implemented, this is not considered an error.

1.4. Server Behavior

Sections 17.2.2 and 18.2 of [RFC3315] govern server operation in regards to option assignment. As a convenience to the reader, we mention here that the server will send option foo only if configured with specific values for foo and if the client requested it.

Option OPTION_V6_S46_PRIORITY is a singleton. Servers MUST NOT send more than one instance of the OPTION_V6_S46_PRIORITY option.

1.5. S46 Mechanisms and their Identifying Option Codes

The following table shows the currently defined option codes and the S46 mechanisms which they represent. This list is complete at the time of writing, but should not be considered definitive as new S46 mechanisms may be defined in the future.

<table>
<thead>
<tr>
<th>Option Code</th>
<th>S46 Mechanism</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>64</td>
<td>DS-Lite</td>
<td>RFC6334</td>
</tr>
<tr>
<td>88</td>
<td>DHCPv6 over DHCPv6</td>
<td>RFC7341</td>
</tr>
<tr>
<td>94</td>
<td>MAP-E</td>
<td>RFC7598</td>
</tr>
<tr>
<td>95</td>
<td>MAP-T</td>
<td>RFC7598</td>
</tr>
<tr>
<td>96</td>
<td>Lightweight 4over6</td>
<td>RFC7598</td>
</tr>
</tbody>
</table>

Table 1: DHCPv6 Option to S46 Mechanism Mappings

2. Security Considerations

Security considerations discussed in [RFC6334] and [RFC7598] apply for this document.
Misbehaving intermediate nodes may alter the content of the S46 option. This may lead to setting a different IPv4 service continuity mechanism than the one initially preferred by the network side.

3. IANA Considerations

IANA is kindly requested to allocate the following DHCPv6 option code:

TBD for OPTION_V6_S46_PRIORITY

All values should be added to the DHCPv6 option code space defined in Section 24.3 of [RFC3315].

4. Acknowledgements

Many thanks to O. Troan, S. Barth, A. Yourtchenko, B. Volz, T. Mrugalski for their input and suggestions.

5. References

5.1. Normative References


5.2. Informative References


Authors' Addresses

Mohamed Boucadair
France Telecom
Rennes
France

Email: mohamed.boucadair@orange.com
Ian Farrer
Deutsche Telekom
Germany

Email: ian.farrer@telekom.de
Deployment Considerations for Lightweight 4over6
draft-sun-softwire-lightweigh-4over6-deployment-04

Abstract

Lightweight 4over6 is a mechanism which moves the translation function from tunnel lwAFTR (AFTR) to lwB4s (B4s), and hence reduces the mapping scale on the lwAFTR to per-customer level. This document discusses various deployment models of Lightweight 4over6. It also describes the deployment considerations and applicability of the Lightweight 4over6 architecture.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on January 15, 2014.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents.
carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction ........................................................................ 3
2. Requirements Language ...................................................... 4
3. Deployment Model ............................................................. 5
4. Overall Deployment Considerations ..................................... 7
  4.1. Addressing and Routing ............................................... 7
  4.2. Port-set Management .................................................. 7
  4.3. lWFTR Discovery ....................................................... 8
  4.4. Impacts on Accounting ................................................ 8
5. lWFTR Deployment Consideration ....................................... 9
  5.1. Logging at the lWFTR .................................................. 9
  5.2. MTU and Fragmentation Considerations ......................... 9
  5.3. Reliability Considerations of lWFTR ............................. 9
  5.4. Placement of AFTR .................................................... 10
  5.5. Port set algorithm consideration .................................. 10
  5.6. Path Consistency Consideration .................................. 10
6. lWB4 Deployment Consideration ........................................ 12
  6.1. NAT traversal issue ................................................... 12
  6.2. Static Port Forwarding Configuration ............................. 12
7. DS-Lite Compatibility Consideration .................................. 13
  7.1. Case 1: Integrated Network Element with Lightweight 4over6 and DS-Lite AFTR Scenario .................. 13
  7.2. Case 2: DS-Lite Coexistent scenario with Separated AFTR .. 14
8. Acknowledgement ............................................................ 15
9. References ................................................................. 16
Appendix 1. Appendix: Experimental Result ............................ 19
  1.1. Experimental environment ......................................... 19
  1.2. Experimental results ............................................... 20
  1.3. Conclusions ......................................................... 21
Authors’ Addresses .......................................................... 22
1. Introduction

Lightweight 4over6 [I-D.ietf-softwire-lw4over6] is an extension to DS-Lite which simplifies the AFTR module [RFC6333] with distributed NAT function among B4 elements. The lwB4 in Lightweight 4over6 is provisioned with an IPv6 address, an IPv4 address and a port-set. It performs NAPT on end user’s packets with the provisioned IPv4 address and port-set. IPv4 packets are forwarded between the lwB4 and the lwAFTR over a Softwire using IPv4-in-IPv6 encapsulation. The lwAFTR maintains one mapping entry per subscriber with the IPv6 address, IPv4 address and port-set. Therefore, this extension removes the NAT44 module from the AFTR and replaces the session-based NAT table to a per-subscriber based mapping table. This should relax the requirement to create dynamic session-based log entries. This mechanism preserves the dynamic feature of IPv4/IPv6 address binding as in DS-Lite, so it has no coupling between IPv6 address and IPv4 address/port-set as any full stateless solution ([RFC6052] or [I-D.ietf-softwire-map]) requires. This document discusses deployment models of Lightweight 4over6. It also describes the deployment considerations and applicability of the Lightweight 4over6 architecture.

Terminology of this document follows the definitions and abbreviations of [I-D.ietf-softwire-lw4over6].
2. Requirements Language

   The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].
3. Deployment Model

Lightweight 4over6 is suitable for operators who would like to free any correlation of the IPv6 address with IPv4 address and port-set (or port-range). In comparison to full stateless solutions like MAP [I-D.ietf-softwire-map] and 4rd [I-D.ietf-softwire-4rd], Lightweight 4over6 frees address planning of IPv6 delegation for CPE from mapping rule administration and management in the network. Thus, IPv6 addressing is completely flexible to fit other deployment requirements, e.g., auto-configuration, service classification, user management, QoS support, etc. The philosophy here is that bits of IPv6 address should be left for IPv6 usage first.

Lightweight 4over6 can be deployed in a residential network (depicted in Figure 1). In this scenario, a lwB4 would acquire an IPv4 address and a port-set after a successful user authentication process and IPv6 provisioning process. Then, it establishes an IPv4-in-IPv6 softwire using the IPv6 address to deliver IPv4 services to its connected host via the lwAFTR in the network. The lwB4 can act as a CPE, or software located in the host. The lwAFTR supports Lightweight 4over6 which keeps the mapping between lwB4’s IPv6 address and its allocated IPv4 address + port set. The supporting system may keep the binding information as well for logging and user management.

![Figure 1 Deployment Model](image)

There are two deployment models in practice: one is called bottom-up and the other is top-down. In bottom-up model, after port-restricted
IPv4 address is allocated to a given subscriber, the lwAFTR will report mapping records to the supporting system on creating a binding for traffic logging if necessary. Operators may use [I-D.ietf-behave-syslog-nat-logging] or [I-D.ietf-behave-ipfix-nat-logging] to report the port set allocated by lwAFTR. In this way, the lwAFTR can determine the binding by its own and there is little impact on existing network architecture. In top-down model, the Supporting system should firstly determine the binding information for each subscriber and then synchronize it with the lwAFTR. With this method, one binding record can be easily synchronized with multiple lwAFTRs and stateless failover can be achieved. However, new mechanism (e.g. [I-D.zhou-dime-4over6-provisioning]) needs to be introduced to notify each individual binding record between the Supporting system and the lwAFTR.
4.  Overall Deployment Considerations

4.1.  Addressing and Routing

In Lightweight 4over6, there is no inter-dependency between IPv4 and IPv6 addressing schemes. IPv4 address pools are configured centralized in lwAFTR for IPv6 subscribers. These IPv4 prefix must advertise to IPv4 Internet accordingly.

For IPv6 addressing and routing, there are no additional addressing and routing requirements. The existing IPv6 address assignment and routing announcement should not be affected. For example, in PPPoE scenario, a CPE could obtain a prefix via prefix delegation procedure, and the hosts behind CPE would get its own IPv6 addresses within the prefix through SLAAC or DHCPv6 statefully. This IPv6 address assignment procedure has nothing to do with restricted IPv4 address allocation.

4.2.  Port-set Management

In Lightweight 4over6, each lwB4 will get its restricted IPv4 address and a port-set after successful user authentication process and IPv6 provisioning process. This port-set assignment can be achieved by DHCPv4-over-DHCPv6 [I-D.ietf-dhc-dhcpv4-over-dhcpv6] and PCP [I-D.ietf-pcp-port-set].

Operator may use DHCPv4 to provision IPv4 address to the lwB4. In a typical deployment, the DHCP server is a centralized DHCP server and lwAFTR is the DHCP relay agent to relay the dhcp messages to the server over unicast. Rarely DHCP server will collocate with the lwAFTR to provision IPv4 resources to the lwB4.

Operator may also use PCP Port-set Option to provision IPv4 address and port-set to the lwB4. In a typical deployment, PCP server will collocate with lwAFTR, and the subscriber’s binding can be determined by lwAFTR. The PCP request should be sent to the lwAFTR’s tunnel end-point address. It is not common that PCP server will be centralized deployed in which the lwAFTR is the PCP proxy to relay PCP requests.

It is also possible that subscriber’s binding is determined in AAA server. In this case, the BNGs will embed with a DHCPv4-over-DHCPv6 server function which allows them to locally handle any DHCPv4-over-DHCPv6 requests initiated by hosts. The AAA server will pass the subscriber’s binding to a BNG using the AAA attribute in [I-D.sun-softwire-lw4over6-radext] and in turn populates the mapping of the lwB4.
Some operators may offer different service level agreements (SLA) to users that some users may require more ports than others. In this deployment scenario, the operator can implement differentiated policies in provisioning system specified to a user’s lwB4 or a group of lwB4s to allocate a certain range of port-set. The lwAFTR may also run multiple instances with different port-set sizes to build the mapping table.

4.3. lwAFTR Discovery

A Lightweight 4over6 lwB4 must discover the lwAFTR’s IPv6 address before offering any IPv4 services. This IPv6 address can be learned through an out-of-band channel, static configuration, or dynamic configuration. In practice, Lightweight 4over6 lwB4 can use the same DHCPv6 option [RFC6334] to discover the FQDN of the lwAFTR.

When Lightweight 4over6 is deployed in the same place with DS-Lite, either different FQDNs can be configured for Lightweight 4over6 and DS-Lite separately or different DHCPv6 options can be used for Lightweight 4over6 [I-D.sun-softwire-lw4over6-dhcpv6] and DS-Lite. More detailed considerations on DS-Lite compatibility will be discussed in Section 6.

4.4. Impacts on Accounting

In Lightweight 4over6, the accounting impact due to the tunneling protocol is the same with DS-Lite (see section 6.2 of [RFC6908]). However, since in Lightweight 4over6, the IPv4 service is only available after port-set allocation, if operators will regard IPv4 service as a on-demand value-added service, e.g. IPv6 connectivity is offered by default, while IPv4 connectivity will be offered until a subscriber requires, etc., IPv4 service accounting should start after port-set allocation has completely.
5. lwAFTR Deployment Consideration

As Lightweight 4over6 is an extension to DS-Lite, both technologies share similar deployment considerations. For example: Interface consideration, Lawful Intercept Considerations, Blacklisting a shared IPv4 Address, AFTR’s Policies, AFTR Impacts on Accounting Process, etc., in [RFC6908] can also be applied here. This document only discusses new considerations specific to Lightweight 4over6.

5.1. Logging at the lwAFTR

In Lightweight 4over6, operators only log one entry per subscriber. The log should include subscriber’s IPv6 address used for the softwire, the public IPv4 address and the port-set. The port set algorithm implemented in Lightweight 4over6 lwAFTR should be synchronized with the one implemented in logging system. For example, if contiguous port set algorithm is adopted in the lwAFTR, the same algorithm should also been applied to the logging system.

Since the mapping in lwAFTR does not contain destination-specific information, operator should be aware that they will not be able to have destination-specific log.

5.2. MTU and Fragmentation Considerations

As Lightweight 4over6 is also a tunneling protocol, the same consideration regarding to the fragmentation and reassembly in DS-Lite [RFC6908] can also be applied. The only difference is that NAT functionality has been removed to lwB4 from lwAFTR in Lightweight 4over6. Therefore, on receiving an IPv4 fragmented packet after decapsulation in the lwB4, the lwB4 should further re-assemble the packets before doing NAT since the transport protocol information is only available in the first fragment.

5.3. Reliability Considerations of lwAFTR

Operators may deploy multiple lwAFTRs for robustness, reliability, and load balancing. In Lightweight 4over6, subscriber to IPv4 and port-set mapping must be pre-provisioned in the lwAFTR before providing IPv4 servies. For redundancy, the backup lwAFTR must either have the subscriber mapping already provisioned or notify the lwB4 to create a new mapping in the backup lwAFTR. The first option can be considered as Hot Standby mode, which requires state synchronization between multiple lwAFTRs. In Hot Standby mode, the bindings are replicated on-the-fly from the Primary lwAFTR to the Backup lwAFTR. When the Primary lwAFTR fails, the Backup lwAFTR will take over all the existing established sessions. In this mode, the internal hosts are not required to re-initiate the bindings with the
external hosts. In Lightweight 4over6, since the number of mapping states has been greatly reduced compared to DS-Lite, it is reasonable to adopt Hot Standby mode when there are only two lwAFTRs (one for Primary lwAFTR and one for Backup lwAFTR). However, if the number of lwAFTRs is larger than two, it is not scalable to deploy Hot Standby mode since each two of the lwAFTRs should to synchronize the binding states.

The second option is to use Cold Standby mode which does not require a Backup Standby lwAFTR to synchronize binding states. In failover, the lwAFTR has to notify the lwB4 to create a new binding, or fetch the binding by itself. [I-D.lee-softwire-lw4over6-failover] describes these two approaches for simple Cold Standby mode. For most deployment scenarios, we believe that Cold Standby mode should be sufficient enough and is thus recommended.

5.4. Placement of AFTR

The lwAFTR can be deployed in a "centralized model" or a "distributed model".

In the "centralized model", the lwAFTR could be located at the higher place, e.g. at the exit of MAN, etc. Since the lwAFTR has good scalability and can handle numerous concurrent sessions, we recommend to adopt the "centralized model" for Lightweight 4over6 as it is cost-effective and easy to manage.

In the "distributed model", lwAFTR is usually integrated with the BRAS/SR. Since newly emerging customers might be distributed in the whole Metro area, we have to deploy lwAFTR on all BRAS/SRs. This will cost a lot in the initial phase of the IPv6 transition period.

5.5. Port set algorithm consideration

If each lwB4 is given a set of ports, port randomization algorithm can only select port in the given port-set. This may introduce security risk because hackers can make a more predictable guess of what port a subscriber may use. Therefore, non-continuous port set algorithms (e.g. as defined in [I-D.ietf-softwire-map]) can be used to improve security.

5.6. Path Consistency Consideration

In Lightweight 4over6, if the binding state is not synchronized among multiple lwAFTRs, the lwAFTR in which the subscriber’s binding state is stored should be exactly the one to service the subscriber. Otherwise, there will be no match in lwAFTR. This requires the provision packets (either using DHCPv4-over-DHCPv6 or PCP Port-set)
should arrive at the same lwAFTR as the subsequent IP-in-IP traffic. If multiple lwAFTRs are using the same Tunnel End Point address and there are intermediate routers between lwB4 and lwAFTR, there might be a problem when intermediate routers perform ECMP based on L4 hash for the plain provisioning packets while doing L3 hash for subsequent IP-in-IP traffic. In this case, it is recommended that the provisioning packet is sent over IPv6 tunnel so that intermediate routers can only process ECMP using L3 hash.
6. lwB4 Deployment Consideration

For lwB4 consideration, the DNS Deployment Considerations and B4 Remote Management in [RFC6908] can also be applied here. In this section, we only describe the considerations specific to Lightweight 4over6.

6.1. NAT traversal issue

In Lightweight 4over6, since the subscriber’s source port will be restricted to the port-set allocated from the provisioning system, this will have impact on some NAT traversal mechanisms. For example, in UPnP 1.0, the external port number which can be used by remote peer is selected by UPnP client in end host. If the client randomly selects a port number which is not in that valid port-set, the UPnP process will fail. This is likely to happen because end-host does not know the port-set in lwB4. More detailed experimental results can be found in [I-D.deng-aplusp-experiment-results]. This problem will not exist in UPnP 2.0 because the UPnP client in the end-host will negotiate the external port number with the server. Another way is to implement a mechanism (e.g. [I-D.ietf-pcp-port-set], etc.) in end host to fetch the port-set from lwB4. The UPnP client can then select the port number within the port-set.

6.2. Static Port Forwarding Configuration

Currently, some external initiated applications rely on manual port configuration to reserve a port in the CPE. The restricted port-set in lwB4 will also have impacts on manual port forwarding configuration. It is recommended that the port-set allocated from the provisioning system should be shown explicitly in the lwB4, which can be used as a hint for subscribers to add port forwarding mapping.
7. DS-Lite Compatibility Consideration

Lightweight 4over6 can be either deployed all alone, or combined with DS-Lite [RFC6333]. Since Lightweight 4over6 does not any have extra requirement on IPv6 addressing, it can use use the same addressing scheme with DS-Lite, together with routing policy, user management policy, etc. Besides, the bottom-up model has quite similar requirement and workflow on the supporting system with DS-Lite. Therefore, it is suitable for operators to deploy incrementally in existing DS-Lite network.

7.1. Case 1: Integrated Network Element with Lightweight 4over6 and DS-Lite AFTR Scenario

In this case, DS-Lite has been deployed in the network. Later in the deployment schedule, the operator decided to implement Lightweight 4over6 lwAFTR function in the same network element (depicted in Figure2). Therefore, the same network element needs to support both transition mechanisms.

There are two options to distinguish the traffic from two transition mechanisms.

The first one is to distinguish using the client’s source IPv4 address. The IPv4 address from Lightweight 4over6 is public address as NAT has been done in the lwB4, and IPv4 address for DS-lite is private address as NAT will be done on AFTR. When the network element receives an encapsulated packet, it would de-capulate packet and apply the transition mechanism based on the IPv4 source address in the packet. This requires the network element to examine every packet and may introduce significant extra load to the network element. However, both the B4 element and Lightweight 4over6 lwB4 can use the same DHCPv6 option [RFC6334] with the same FQDN of the AFTR and lwAFTR.

The second one is to distinguish using the destination’s tunnel IPv6 address. One network element can run separated instances for Lightweight 4over6 and DS-Lite with different tunnel addresses. Then B4 element and Lightweight 4over6 lwB4 can use the same DHCPv6 option [RFC6334] with different FQDNs pointing to corresponding tunnel addresses. This requires the supporting system should distinguish different types of users when assigning the FQDNs in DHCPv6 process. Another option is to use a new DHCPv6 option [I-D.sun-softwire-lw4over6-dhcpv6] to discover lwAFTR’s FQDN.
7.2. Case 2: DS-Lite Coexistent scenario with Separated AFTR

This is similar to Case 1. The difference is the lwAFTR and AFTR functions won’t be co-located in the same network element (depicted in Figure3). This use case decouples the functions to allow more flexible deployment. For example, an operator may deploy AFTR closer to the edge and lwAFTR closer to the core. Moreover, it does not require the network element to pre-configure with the CPE’s IPv6 addresses. An operator can deploy more AFTR and lwAFTR at needed. However, this requires the B4 and lwB4 to discover the corresponding network element. In this case, B4 element and Lightweight 4over6 lwB4 can still use [RFC6334] with different FQDNs pointing to corresponding tunnel end-point addresses, and the supporting system should distinguish different types of users.
8. Acknowledgement

TBD
9. References

[I-D.bajko-pripaddrassign]

[I-D.cui-softwire-b4-translated-ds-lite]

[I-D.deng-aplusp-experiment-results]
Deng, X., Boucadair, M., and F. Telecom, "Implementing A+P in the provider’s IPv6-only network", draft-deng-aplusp-experiment-results-00 (work in progress), March 2011.

[I-D.ietf-behave-ipfix-nat-logging]

[I-D.ietf-behave-syslog-nat-logging]

[I-D.ietf-dhc-dhcpv4-over-ipv6]

[I-D.ietf-pcp-base]

[I-D.ietf-pcp-port-set]
[I-D.ietf-softwire-4rd]

[I-D.ietf-softwire-dslite-deployment]

[I-D.ietf-softwire-lw4over6]

[I-D.ietf-softwire-map]

[I-D.lee-softwire-lw4over6-failover]

[I-D.sun-softwire-lw4over6-dhcpv6]

[I-D.zhou-dime-4over6-provisioning]


October 2010.


1. Appendix: Experimental Result

We have deployed Lightweight 4over6 in our operational network of HuNan province, China. It is designed for broadband access network, and different versions of lwB4 have been implemented including a linksys box, a software client for windows XP, vista and Windows 7. It can be integrated with existing dial-up mechanisms such as PPPoE, etc. The major objectives listed below aimed to verify the functionality and performance of Lightweight 4over6:

- Verify how to deploy Lightweight 4over6 in a practical network.
- Verify the impact of applications with Lightweight 4over6.
- Verify the performance of Lightweight 4over6.

1.1. Experimental environment

The network topology for this experiment is depicted in Figure 2.

Figure 2 Lightweight 4over6 experiment topology

In this deployment model, lwAFTR is co-located with a extended PCP server to assign restricted IPv4 address and port set for lwB4. It also triggers subscriber-based logging event to a centralized syslog server. IPv6 address pools for subscribers have been distributed to
BRASs for configuration, while the public available IPv4 address pools are configured by the centralized lwAFTR with a default address sharing ratio. It is rather flexible for IPv6 addressing and routing, and there is little impact on existing IPv6 architecture.

In our experiment, lwB4 will firstly get its IPv6 address and delegated prefix through PPPoE, and then initiate a PCP-extended request to get public IPv4 address and its valid port set. The lwAFTR will thus create a subscriber-based state accordingly, and notify syslog server with \{IPv6 address, IPv4 address, port set, timestamp\}.

1.2. Experimental results

In our trial, we mainly focused on application test and performance test. The applications have widely include web, email, Instant Message, ftp, telnet, SSH, video, Video Camera, P2P, online game, voip and so on. For performance test, we have measured the parameters of concurrent session numbers and throughput performance.

The experimental results are listed as follows:

<table>
<thead>
<tr>
<th>Application Type</th>
<th>Test Result</th>
<th>Port Number Occupation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web</td>
<td>ok</td>
<td>normal websites: 10˜20</td>
</tr>
<tr>
<td></td>
<td>IE, Firefox, Chrome</td>
<td>Ajex Flash webs: 30˜40</td>
</tr>
<tr>
<td>Video</td>
<td>ok, web based or client based</td>
<td>30˜40</td>
</tr>
<tr>
<td>Instant Message</td>
<td>ok</td>
<td>QQ, MSN, gtalk, skype</td>
</tr>
<tr>
<td>P2P</td>
<td>ok</td>
<td>utorrent,emule,xunlei</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(per seed)</td>
</tr>
<tr>
<td>FTP</td>
<td>need ALG for active mode, flashxp</td>
<td>2</td>
</tr>
<tr>
<td>SSH, TELNET</td>
<td>ok</td>
<td>1 for SSH, 3 for telnet</td>
</tr>
<tr>
<td>online game</td>
<td>ok for QQ, flash game</td>
<td>20˜40</td>
</tr>
</tbody>
</table>

Figure 3 Lightweight 4over6 experimental result
The performance test for lwAFTR is taken on a normal PC. Due to limitations of the PC hardware, the overall throughput is limited to around 800 Mbps. However, it can still support more than one hundred million concurrent sessions.

1.3. Conclusions

From the experiment, we can have the following conclusions:

- Lightweight 4over6 has good scalability. As it is a lightweight solution which only maintains per-subscription state information, it can easily support a large amount of concurrent subscribers.

- Lightweight 4over6 can be deployed rapidly. There is no modification to existing addressing and routing system in our operational network. And it is simple to achieve traffic logging.

- Lightweight 4over6 can support a majority of current IPv4 applications.
Authors’ Addresses

Qiong Sun
China Telecom
Room 708, No.118, Xizhimennel Street
Beijing  100035
P.R.China

Phone: +86-10-58552936>
Email: sunqiong@ctbri.com.cn

Chongfeng Xie
China Telecom
Room 708, No.118, Xizhimennel Street
Beijing  100035
P.R.China

Phone: +86-10-58552116>
Email: xiechf@ctbri.com.cn

Yiu L. Lee
Comcast
One Comcast Center
Philadelphia, PA  19103
USA

Email: yiu_lee@cable.comcast.com

Maoke Chen
FreeBit Co., Ltd.
13F E-space Tower, Maruyama-cho 3-6
Shibuya-ku, Tokyo  150-0044
Japan

Email: fibrib@gmail.com
Abstract

lightweight 4over6 (lw4over6) [I-D.ietf-softwire-lw4over6] is an extension to DS-Lite in which the amount of state maintained in lwAFTR has been reduced to per-subscriber-level. The lwB4 needs to be provisioned with the public IPv4 address and port set it is allowed to use. The DHCPv4 over DHCPv6 Transport [I.D-ietf-dhc-dhcpv4-over-dhcpv6] and Dynamic Host Configuration Protocol (DHCP) Option for Port Set [I.D-sun-dhc-port-set-option] can be used for lwB4 to provision with the public IPv4 address and port set.

However, in many networks, the configuration information may be stored in Authentication Authorization and Accounting (AAA) servers while user configuration is mainly from Broadband Network Gateway (BNG). This document defines a Remote Authentication Dial In User Service (RADIUS) attribute that carries lightweight 4over6 configuration information from AAA server to BNG.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any
1. Introduction

Lightweight 4over6 (lw4over6) [I-D.ietf-softwire-lw4over6] defines a model for providing IPv4 access over an IPv6 network in which the Network Address Translation (NAT) function is performed by the Customer-Premises Equipment (CPE) instead of being centralized on a Carrier-Grade NAT (CGN). Lightweight 4over6 features keeping per-subscriber binding state in the service provider’s network. This per-subscriber binding state is assigned by the provisioning system and should be synchronized between lwAFTRs. In lw4over6, there are multiple mechanisms to provision an lwB4 with the binding state,
including [I-D.ietf-dhc-dhcpv4-over-dhcpv6], [I-D.ietf-softwire-map-dhcp], or [I-D.ietf-pcp-port-set], etc.

In many networks, user configuration information may be managed by AAA (Authentication, Authorization, and Accounting) servers. Current AAA servers communicate using the Remote Authentication Dial In User Service (RADIUS) [RFC2865] protocol. In a fixed line broadband network, the Broadband Network Gateways (BNGs) act as the access gateway of users. For lw4over6 case, the BNGs are assumed to embed a DHCPv4-over-DHCPv6 server function which allows them to locally handle any DHCPv4-over-DHCPv6 requests issued by hosts. The operators may per-configure subscriber’s binding state in AAA server which then passes the information to a BNG and in turn populates the mapping of the subscribe.

This document defines a new RADIUS attribute that can be used in lightweight 4over6 to carry subscriber’s binding state.

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

Terminology defined in [I-D.ietf-softwire-lw4over6] is used extensively in this document.

3. Lightweight 4over6 configuration process with RADIUS

The below Figure 1 illustrates how the RADIUS protocol and DHCPv4-over-DHCPv6 cooperate to provide lwB4 with the binding state.
<table>
<thead>
<tr>
<th>lwB4</th>
<th>BNG</th>
<th>lwAFTR</th>
<th>AAA Server</th>
</tr>
</thead>
<tbody>
<tr>
<td>--PPP LCP Config-Request-----&gt;</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt;---PPP LCP Config-ACK ------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>--PPP IPv6CP Config-Request-&gt;</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt;---PPP IPv6CP Config-ACK ----</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>--------DHCPv6 Solicit--------&gt;</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt;--------DHCPv6 Advertisement--</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt;--------DHCPv6 Request--------&gt;</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt;--------DHCPv6 Reply----------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>--------DHCPv4-QUERY----------&gt;</td>
<td></td>
<td>Access-Request-------&gt;</td>
<td></td>
</tr>
<tr>
<td>(OPTION_DHCPV4_MSG)</td>
<td>lh4o6 attr</td>
<td>(lh4o6 attr)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Configuration-</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>(Optional)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>-----ACK-------&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;------Access-Accept---------</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>(lh4o6 attr)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
AAA server on-demand, it should inform the lwAFTR with the subscriber’s binding state using [I-D.zhou-dime-4over6-provisioning] or COA requests.

Figure 2 illustrates how the RADIUS protocol and DHCPv6 cooperate to provide lwB4 and lwAFTR with tunnel configuration information.

```
<table>
<thead>
<tr>
<th>lwB4</th>
<th>BNG</th>
<th>AAA Server</th>
<th>lwAFTR</th>
</tr>
</thead>
<tbody>
<tr>
<td>--DHCPv6 Request---&gt;</td>
<td>--Access-Request---&gt;</td>
<td>--configuration---&gt;</td>
<td>&lt;--DHCPv6 Reply----</td>
</tr>
<tr>
<td>(OPTION_S46_CONT_LW)</td>
<td>(lw4o6 attr)</td>
<td>(Optional)</td>
<td>(OPTION_S46_CONT_LW)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>DHCPv6</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Radius</td>
</tr>
</tbody>
</table>
```

Figure 2: Lightweight 4over6 configuration process with RADIUS case 2

BNGs act as a RADIUS client and as a DHCPv6 server. Before the tunnel establishes, lwB4 MAY initiate a DHCPv6 Solicit message that includes an Option Request option [RFC3315] with OPTION_S46_CONT_LW option defined in [I-D.ietf-softwire-map-dhcp]. When BNG receives the SOLICIT, it SHOULD initiates radius Access-Request message, in which the User-Name attribute (1) SHOULD be filled by the lwB4 MAC address, to the RADIUS server, the User-password attribute (2) SHOULD be filled by the shared lw4over6 password that has been preconfigured on the DHCPv6 server to get lw4over6 attribute.

If the authentication request is approved by the AAA server, AAA server will determine the IPv6 address, IPv4 address and Port Set for the subscriber. The subscriber’s binding state should be synchronized between AAA server and lwAFTR. If the bindings are pre-configured statically in both AAA server and lwAFTR, the AAA server does not need to configure lwAFTR anymore. Otherwise, if the bindings are locally created in AAA server on-demand, it should inform the lwAFTR as mentioned above.

Similarly, BNGs can act as a RADIUS client and as a PCP server in case an lwB4 runs a PCP client (as depicted in Figure 3).
In the above-mentioned scenarios, Message-Authenticator (type 80) [RFC2865] SHOULD be used to protect both Access-Request and Access-Accept messages.

After receiving the lw4over6-binding attribute in the initial Access-Accept, the BNG SHOULD store the received lw4over6 configuration parameters locally. When the lw4over6 CE sends a DHCP or PCP Request message to request an extension of the lifetime for the assigned address, the BNG does not have to initiate a new Access-Request towards the AAA server to request the lw4o6 binding state. The BNG could retrieve the previously stored lw4o6 configuration parameters and use them in its reply. The BNG will then inform the AAA server with updated lifetime.

If the BNG does not receive the lw4over6-binding attribute in the Access-Accept or if the BNG receives an Access-Reject, the tunnel cannot be established.

4. Attributes

This section defines the lw4o6_binding attribute that is used in both above-mentioned scenarios. The attribute design follows [RFC6158] and refers to [RFC6929].

4.1. lw4o6_binding Attribute

The lw4o6_binding RADIUS attribute contains the subscriber’s binding information including IPv6 address, IPv4 address and the port-set. The BNG SHALL use the binding entry returned in the RADIUS lw4o6_binding attribute to populate the requests.
If the BNG includes the lw4o6_binding attribute, but the AAA server does not recognize it, this attribute MUST be ignored by the AAA server.

If the BNG does not receive the lw4o6_binding attribute in the Access-Accept message and there is the unified server in BNG is not configured to allocate the port-set by itself, the unified SHOULD not response and the tunnel can not be established.

When the Access-Request message is triggered by a DHCP Rebind message, if the binding attribute received in the Access-Accept message is different from the currently used one for that session, the BNG MUST force the lwB4 to re-establish the tunnel using the new binding information received in the Access-Accept message.

The lw4o6_binding Attribute is structured as follows:
0                   1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Type     |    Length     |          Reserved             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ 
|                     IPv6 address                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ 
|                                                               |
|                     IPv4 address                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ 
|    Port Set Index             |        Port Set Mask          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ 

Type
TBD

Length
28

Port Set Index:
Port Set Index identifies a set of ports assigned to a device. The first k bits on the left of the 2-octet field is the Port Set Index value, with the rest of the field right padding zeros.

Port Set Mask:
Port Set Mask indicates the position of the bits used to build the mask. The first k bits on the left is padding ones while the remained (16-k) bits of the 2-octet field on the right is padding zeros.

IPv4 address
The translated IPv4 address for a subscriber.
IPv6 address
The IPv6 address for a subscriber.

Figure 4: Lightweight 4over6 Attribute

5. Table of attributes

The following table provides a guide to which attributes may be found in which kinds of packets, and in what quantity.
The following table defines the meaning of the above table entries.

<table>
<thead>
<tr>
<th>Request</th>
<th>Accept</th>
<th>Reject</th>
<th>Challenge</th>
<th>Accounting</th>
<th># Attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-1</td>
<td>0-1</td>
<td>0</td>
<td>0</td>
<td>0-1</td>
<td>TBD1 lw4o6-binding</td>
</tr>
<tr>
<td>0-1</td>
<td>0-1</td>
<td>0</td>
<td>0</td>
<td>0-1</td>
<td>User-Name</td>
</tr>
<tr>
<td>0-1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0-1</td>
<td>User-Password</td>
</tr>
<tr>
<td>0-1</td>
<td>0-1</td>
<td>0</td>
<td>0</td>
<td>0-1</td>
<td>Service-Type</td>
</tr>
<tr>
<td>0-1</td>
<td>0-1</td>
<td>0-1</td>
<td>0-1</td>
<td>0-1</td>
<td>80</td>
</tr>
</tbody>
</table>

0     This attribute MUST NOT be present in packet.
0+    Zero or more instances of this attribute MAY be present in packet.
0-1   Zero or one instance of this attribute MAY be present in packet.
1     Exactly one instance of this attribute MUST be present in packet.

Figure 5: Lightweight 4over6 Attribute Table

6. Security Considerations

TO BE COMPLETED

7. IANA Considerations

This document has no IANA actions.

8. Acknowledgements

The authors would like to thank the following individuals who have participated in the drafting, review, and discussion of this memo: TO BE COMPLETED

9. References

9.1. Normative References

[I-D.ietf-pcp-port-set]
Internet-Draft   Radius Extension for Lightweight 4over6      March 2014

[I-D.ietf-softwire-lw4over6]
Cui, Y., Sun, Q., Boucadair, M., Tsou, T., Lee, Y., and I.
Farrer, "Lightweight 4over6: An Extension to the DS-Lite
Architecture", draft-ietf-softwire-lw4over6-00 (work in
progress), April 2013.

[RFC2119]  Bradner, S., "Key words for use in RFCs to Indicate

[RFC3484]  Draves, R., "Default Address Selection for Internet

[RFC6334]  Hankins, D. and T. Mrugalski, "Dynamic Host Configuration
Protocol for IPv6 (DHCPv6) Option for Dual-Stack Lite",
RFC 6334, August 2011.

[RFC6887]  Wing, D., Cheshire, S., Boucadair, M., Penno, R., and P.
Selkirk, "Port Control Protocol (PCP)", RFC 6887, April
2013.

9.2.  Informative References

[RFC6333]  Durand, A., Droms, R., Woodyatt, J., and Y. Lee, "Dual-
Stack Lite Broadband Deployments Following IPv4
Exhaustion", RFC 6333, August 2011.

Authors’ Addresses

Chongfeng Xie
China Telecom
P.R.China

Phone: 86 10 58552116
Email: xiechf@ctbri.com.cn

Qiong Sun
China Telecom
P.R.China

Phone: 86 10 58552936
Email: sunqiong@ctbri.com.cn
This document defines a YANG data model for the configuration and management of IPv4-in-IPv6 Softwire Border Routers and Customer Premises Equipment. It covers the Lightweight 4over6, MAP-E and MAP-T Softwire mechanisms.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on April 8, 2016.
Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction ................................................. 2
   1.1. Terminology ........................................... 3
   1.2. YANG Tree Diagrams ..................................... 3
   1.3. YANG Modelling of NAT44 Functionality ................. 4
2. Objectives .................................................. 4
   2.1. Common ................................................. 4
   2.2. Lightweight 4over6 ..................................... 5
   2.3. MAP-E ................................................ 5
   2.4. MAP-T ................................................ 5
3. Softwire YANG Tree Diagrams ................................. 5
   3.1. Common Tree Diagrams .................................. 5
   3.2. Lightweight 4over6 Tree Diagrams ...................... 6
   3.3. MAP-E Tree Diagrams .................................. 8
   3.4. MAP-T Tree Diagrams .................................. 10
   3.5. Notifications for Softwire YANG ....................... 12
4. Softwire YANG Model ......................................... 12
5. Example of Configure Lw4over6 Binding-Table .............. 30
6. Security Considerations ..................................... 31
7. IANA Considerations ........................................ 32
8. Acknowledgements .......................................... 32
9. References ................................................ 32
   9.1. Normative References ................................. 32
   9.2. Informative References ............................... 33
9. Authors’ Addresses ......................................... 34

1. Introduction

The IETF Softwire Working Group has developed several IPv4-in-IPv6 Softwire mechanisms to address various deployment contexts and constraints. As a companion to the architectural specification documents, this document focuses on the provisioning of softwire

Sun, et al. Expires April 8, 2016 [Page 2]
functional elements: Border Routers (BRs) and Customer Premises Equipment (CPEs).

This document defines a YANG data model that can be used to configure and manage IPv4-in-IPv6 Softwire BRs and/or CEs via NETCONF protocol [RFC6241]. To ensure interoperability in mixed vendors environments, it is important that the models can be easily reused between different vendors and implementations.

There are three different mechanisms in this YANG model. Each specific mechanism has their separate YANG modules respectively:

- Lightweight 4over6 [RFC7596]
- MAP-E [RFC7597]
- MAP-T [RFC7599]

This model is structured into two root containers:

1. Container "softwire-config" holds the collection of YANG definitions common to all softwire configuration of BRs and CEs.

2. Container "softwire-state" holds YANG definitions for the operational state of the Softwire BRs and CEs.

The model also includes a notification module. The aim is to notify the client that a specific status has been changed.

This approach has been taken so that the model can be easily extended in the future to support additional softwire mechanisms, should this be necessary.

1.1. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

The reader should be familiar with the terms defined in [RFC7596] [RFC7597] [RFC7599], and the YANG data modelling language [RFC6020].

1.2. YANG Tree Diagrams

The Softwire YANG tree diagrams provide a concise representation of the YANG modules to help the reader understand the module structure. The meaning of the symbols in these diagrams are as follows:
1.3. YANG Modelling of NAT44 Functionality

This documented model does not include NAT-specific provisioning parameters other than the external IP address and port set which a softwire client may use for NAT44. Additional NAT-specific considerations are out of scope of this document. A YANG model for the configuration and management of NAT gateways is described in [I-D.sivakumar-yang-nat].

2. Objectives

This document defines a YANG data model that can be used to configure and manage BRs and CEs for the following IPv4-in-IPv6 Softwire mechanisms: Lightweight 4over6, MAP-E and MAP-T.

For Lightweight 4over6, the configuration and management information of lwB4 and lwAFTR are different. The lwAFTR needs to maintain a binding table of configured lwB4s. The lwB4 holds the configuration for its local tunnel, the lwAFTR address and a NAPT table of active translations.

For the MAP-E and MAP-T, CE and BR both need to maintain the map-rule table. Thus, there is no need to distinguish BR and CE.

2.1. Common

The common model abstracts the shared features of different BRs and CEs making a single node softwire description for common features.

The following sections of the document are structured with the root of the softwire YANG model (common to all mechanisms) described first. The subsequent sections describe the models relevant to the different softwire mechanisms. All functions are listed, but the YANG models use the "feature" statement to distinguish among the different softwire mechanisms.
2.2. Lightweight 4over6

Lightweight 4over6 includes two elements: lwAFTR and lwB4. The lwAFTR holds configuration for IPv4-IPv6 address bindings which is used for the forwarding of traffic originating from lwB4s. And the lwB4 is configured with the relevant parameters for establishing the IPv4 in IPv6 tunnel including an IPv6 address for the lwAFTR and the IPv4 configuration for NAPT44.

2.3. MAP-E

MAP-E elements (BR and CE) are provisioned with the MAP rules necessary for defining MAP domains and forwarding rules.

2.4. MAP-T

MAP-T elements (BR and CE) are provisioned with the MAP rules necessary for defining MAP domains and forwarding rules. MAP-T CEs an additional "ipv6-prefix" parameter is also configured.

3. Softwire YANG Tree Diagrams

3.1. Common Tree Diagrams

Figure 1 describes the softwire data model which is common to all of the different softwire mechanisms listed in Section 1:

```
+--rw softwire-config
    |  +--rw description?               string
    |  +--rw lw4over6 {lw4over6}?
    |     |  +--rw lwafr {lwafr}?
    |     |  +--rw lw4 {lw4}?
    |  +--rw map-e {map-e}?
    |  +--rw map-t {map-t}?

+--ro softwire-state
    |  +--ro description?               string
    |  +--ro lw4over6 {lw4over6}?
    |     |  +--ro lwafr {lwafr}?
    |     |  +--ro lw4 {lw4}?
    |  +--ro map-e {map-e}?
    |  +--ro map-t {map-t}?
```

Figure 1: Softwire Common Data Model Structure

The mechanism specific models for lw4over6, MAP-E and MAP-T are described in detail in the following sections.
3.2. Lightweight 4over6 Tree Diagrams

Figure 2 defines the softwire data model for Lightweight 4over6 which includes lwAFR and lwB4:

```
module: ietf-softwire
  +--rw softwire-config
    +--rw lw4over6 {lw4over6}?
      +--rw lwaftr {lwaftr}?
        +--rw enable?  boolean
        +--rw lwaftr-instances
          +--rw lwaftr-instance* [id]
            +--rw id  uint32
            +--rw name?  string
            +--rw softwire-num-threshold  uint32
            +--rw tunnel-mtu  uint16
            +--rw fragment-mru  uint16
            +--rw binding-table
              +--rw binding-entry* [binding-ipv6info]
                +--rw binding-ipv6info union
                  +--rw binding-ipv4-addr inet:ipv4-address
                +--rw port-set
                  +--rw offset  uint8
                  +--rw psid  uint16
                  +--rw psid-len  uint8
                +--rw lwaftr-ipv6-addr inet:ipv6-address
              +--rw lifetime?  uint32
        +--rw lwb4 {lwb4}?
          +--rw enable?  boolean
          +--rw lwb4-instances
            +--rw lwb4-instance* [binding-ipv6info]
              +--rw name?  string
              +--rw tunnel-mtu  uint16
              +--rw fragment-mru  uint16
              +--rw b4-ipv6-address-format  boolean
              +--rw binding-ipv6info union
                +--rw binding-ipv4-addr inet:ipv4-address
              +--rw port-set
                +--rw offset  uint8
                +--rw psid  uint16
                +--rw psid-len  uint8
              +--rw lwaftr-ipv6-addr inet:ipv6-address
              +--rw lifetime?  uint32
      +--ro softwire-state
        +--...
Figure 2: Softwire Lightweight 4over6 Data Model Structure

Some of the important lwAFTR nodes:

- binding-entry: used to define the binding relationship between 3-tuples, which contains the lwB4’s IPv6 address, the allocated IPv4 address and restricted port-set. For detail information, please refer to [RFC7596].

- tunnel-mtu: used to set the value of MTU for the Lightweight 4over6 tunnel.

- fragment-mru: used to set the value of fragment for Lightweight 4over6 tunnel.

- tunnel-num-threshold: used to set the maximum number of tunnels that can be created on the lw4over6 device simultaneously.

- active-tunnel-num (ro): used to present the number of tunnels currently provisioned on the device.
o active (ro): used to show the status of particular binding-entry.

Some of the important lwB4 nodes:

o b4-ipv6-addr-format: indicates the format of lwB4 IPv6 address. If set to true, it indicates that the IPv6 source address of the lwB4 is constructed according to the description in [RFC7596]; if set to false, the lwB4 can use any /128 address from the assigned IPv6 prefix.

o binding-ipv6info: used to set the IPv6 address type which is combined in a binding entry, for a complete address or a prefix.

3.3. MAP-E Tree Diagrams

Figure 3 defines the softwire data model for MAP-E:
module: ietf-softwire
   +--rw softwire-config
      +--rw enable? boolean
      +--rw map-e {map-e}?
         +--rw map-e-instances boolean
            +--rw map-e-instance* [id]
               +--rw id uint32
               +--rw name? string
               +--rw map-rules
                  +--rw map-rule* [id]
                     +--rw id uint8
                     +--rw forwarding boolean
                     +--rw rule-ipv6-prefix inet:ipv6-prefix
                     +--rw rule-ipv4-prefix inet:ipv4-prefix
                     +--rw port-set
                        +--rw offset uint8
                        +--rw psid uint16
                        +--rw psid-len uint8
                        +--rw ea-len uint8
                        +--rw tunnel-mtu uint16
                        +--rw fragment-mru uint16
                        +--rw br-ipv6-addr inet:ipv6-address
   +--ro softwire-state
      +--ro map-e {map-e}?
         +--ro map-e-instances
            +--ro map-e-instance* [id]
               +--ro id int32
               +--ro name? string
               +--ro sentPacket? yang:zero-based-counter64
               +--ro sentByte? yang:zero-based-counter64
               +--ro rcvdPacket? yang:zero-based-counter64
               +--ro rcvdByte? yang:zero-based-counter64
               +--ro droppedPacket? yang:zero-based-counter64
               +--ro droppedByte? yang:zero-based-counter64

Figure 3: Softwire MAP-E Data Model Structure

Some of the important MAP-E nodes:

- forwarding: This parameter specifies whether the rule is to be used for forwarding (FMR). If set, this rule is used as an FMR; if not set, this rule is a BMR only and MUST NOT be used for forwarding. See Section 4.1 [RFC7598].

- offset: used to set the number of offset bits.
o psid: used to algorithmically identify a set of ports exclusively for a specific softwire.

o ea-len: used to set the length of the Embedded-Address (EA), which defined in the mapping rule for a MAP domain.

o tunnel-mtu: used to set the value of MTU for MAP-E tunnel.

o fragment-mru: used to the value of fragment for MAP-E tunnel.

o stat-count (ro): use to show the numbers of packets and bytes information of specific device respectively.

3.4. MAP-T Tree Diagrams

Figure 4 defines the softwire data model for MAP-T:
module: ietf-softwire
  +--rw softwire-config
  |  +--... 
  |  +--rw map-t {map-t}? 
  |     +--rw enable? boolean
  |     +--rw map-t-instances 
  |        +--rw map-t-instance* [id] 
  |           +--rw id uint32
  |           +--rw name? string
  |           +--rw map-rules 
  |              +--rw map-rule* [id] 
  |                  +--rw id uint8
  |                  +--rw forwarding boolean
  |                  +--rw rule-ipv6-prefix inet:ipv6-prefix
  |                  +--rw rule-ipv4-prefix inet:ipv4-prefix
  |                  +--rw port-set
  |                     +--rw offset uint8
  |                     +--rw psid uint16
  |                     +--rw psid-len uint8
  |                     +--rw ea-len uint8
  |                  +--rw dmr-ipv6-prefix? inet:ipv6-prefix
  +--ro softwire-state
  |  +--... 
  |  +--ro map-t {map-t}? 
  |     +--ro map-t-instances 
  |        +--ro map-t-instance* [id] 
  |           +--ro id int32
  |           +--ro name? string
  |           +--ro sentPacket? yang:zero-based-counter64
  |           +--ro sentByte? yang:zero-based-counter64
  |           +--ro rcvdPacket? yang:zero-based-counter64
  |           +--ro rcvdByte? yang:zero-based-counter64
  |           +--ro droppedPacket? yang:zero-based-counter64
  |           +--ro droppedByte? yang:zero-based-counter64

Figure 4: Softwire MAP-T Data Model Structure

Some of the important MAP-T nodes:

- dmr-ipv6-prefix: defines the DMR in MAP-T. This parameter is optional when configuring a MAP-T BR.

- stat-count (ro): use to show the numbers of packets and bytes information of specific device respectively.
3.5. Notifications for Softwire YANG

This section describes the diagram tree for the notifications. These notifications pertain to configuration and monitoring portions of specific Softwire mechanisms. The logic is that, the softwire instance notifies the NETCONF client with the index for a mapping entry and then the NETCONF client retrieves the related information from the operational datastore of that instance.

module: ietf-softwire
notifications:
  +++-n softwire-lwaftr-event {lw4over6,lwaftr}?
    +--ro lwaftr-id?        -> /softwire-state/lw4over6/lwaftr/.../id
    +--ro invalid-entry*    -> /softwire-config/lw4over6/lwaftr/.../binding-table/binding-entry/binding-ipv6info
    +--ro modified-entry*   inet:ipv6-address
  +++-n softwire-lwb4-event {lw4over6,lwb4}?
    +--ro lwb4-binding-ipv6-addr-change inet:ipv6-address
  +++-n softwire-map-e-event {map-e}?
    +--ro map-e-id?        -> /softwire-state/map-e/.../id
    +--ro invalid-entry-id*    -> /softwire-config/map-e/.../map-rules/map-rule/id
    +--ro added-entry*      uint32
    +--ro modified-entry*   -> /softwire-config/map-e/.../map-rules/map-rule/id
  +++-n softwire-map-t-event {map-t}?
    +--ro map-t-id?        -> /softwire-state/map-t/.../id
    +--ro invalid-entry-id*    -> /softwire-config/map-t/.../map-rules/map-rule/id
    +--ro added-entry*      uint32
    +--ro modified-entry*   -> /softwire-config/map-t/.../map-rules/map-rule/id

Figure 5: Softwire Notifications Data Model Structure

Some of the important notification nodes:

- invalid-entry, added-entry, modified-entry: used to notify the client that a specific binding entry or MAP rule is expired/invalid, added, or modified.

- lwb4-binding-ipv6-addr-change: use to notify that the lwB4’s binding-ipv6-address has been changed or the value of the 'b4-ipv6-addr-format' is "false".

4. Softwire YANG Model

This module imports typedefs from [RFC6991].

<CODE BEGINS> file "ietf-softwire@2015-08-31.yang"

module ietf-softwire {

namespace "urn:ietf:params:xml:ns:yang:ietf-softwire";
prefix "softwire";
import ietf-inet-types {prefix inet; }
import ietf-yang-types {prefix yang; }

organization "Softwire Working Group";

contact
Qi Sun sunqi.ietf@gmail.com
Hao Wang wanghai3@mail.tsinghua.edu.cn
Yong Cui yong@csnet1.cs.tsinghua.edu.cn
Ian Farrer ian.farrer@telekom.de
Mohamed Boucadair mohamed.boucadair@orange.com
Rajiv Asati rajiva@cisco.com
";

description
"This document defines a YANG data model for the configuration and management of IPv4-in-IPv6 Softwire Border Routers and Customer Premises Equipment. It covers Lightweight 4over6, MAP-E and MAP-T Softwire mechanisms.

Copyright (c) 2014 IETF Trust and the persons identified as authors of the code. All rights reserved.
This version of this YANG module is part of RFC XXX; see the RFC itself for full legal notices.";

revision 2015-09-30 {

description
"Version-04: fix YANG syntax; Add flags to map-rule; Remove the map-rule-type element. ";
reference "tbc";
}

revision 2015-04-07 {

description
"Version-03: Integrate lw4over6; Update state nodes; Correct grammar errors; Reuse groupings; Update descriptions.
Simplify the model.";
reference "tbc";
}

revision 2015-02-10 {

description
"Version-02: Add notifications.";
reference "tbc";
}
/*
* Features
*/

feature lw4over6 {
  description
  "Lightweight 4over6 (lw4over6) is an IPv4-over-IPv6 tunnelling transition mechanism. Lightweight 4over6 is a solution designed specifically for complete independence between IPv6 subnet prefix (and /128 IPv6 address) and IPv4 address with or without IPv4 address sharing.

This is accomplished by maintaining state for each softwire (per-subscriber state) in the central lwAFTR and a hub-and-spoke forwarding architecture. In order to delegate the NAPT function and achieve IPv4 address sharing, port-restricted IPv4 addresses needs to be allocated to CEs.";

  reference
  "I-D.ietf-softwire-lw4over6";
}

feature lwaftr {
  if-feature lw4over6;
  description
  "The AFTRs (BRs) for Lightweight 4over6, so-called lwAFTR. This feature indicates that a instance functions as a lwAFTR.
A lwAFTR is an IPv4-in-IPv6 tunnel concentrator that maintains per-subscriber IPv4-IPv6 address binding.";
}

feature lwb4 {

}
if-feature lw4over6;

description "The B4s (CEs) for Lightweight 4over6, so-called lwB4. This feature indicates that a instance functions as a lwB4. A lwB4 is an IPv4-in-IPv6 tunnel initiator. It is dual-stack capable node, either a directly connected end-host or a CE. It sources IPv4 connections using the configured port-set and the public IPv4 address.";
}

feature map-e {

description "MAP-E is an IPv6 transition mechanism for transporting IPv4 packets across an IPv6 network using IP encapsulation. MAP-E allows for a reduction of the amount of centralized state using rules to express IPv4/IPv6 address mappings. This introduces an algorithmic relationship between the IPv6 subnet and IPv4 address. This relationship also allows the option of direct, meshed connectivity between users. Alternatively, MAP-E can be configured to support IPv4/IPv6 independent binding. This feature indicates the instance functions as a MAP-E instance.";

reference "I-D.ietf-softwire-map";
}

feature map-t {

description "The Mapping of Address and Port - Translation (MAP-T) architecture is a double stateless NAT64 based solution. It uses the stateless algorithmic address & transport layer port mapping scheme defined in MAP-E. The MAP-T solution differs from MAP-E in the use of IPv4-IPv6 translation, rather than encapsulation, as the form of IPv6 domain transport. This feature indicates the instance functions as a MAP-T instance. ";

reference "I-D.ietf-softwire-map-t";
}

/* * Grouping */

grouping port-set {

description "Use the PSID algorithm to represent a range of transport layer
leaf offset {
    type uint8 {
        range 0..16;
    }
    mandatory true;
    description
        "The number of offset bits. In Lightweight 4over6, the default
        value is 0 for assigning one contiguous port range. In MAP-E/T,
        the default value is 6, which excludes system ports by default
        and assigns distributed port ranges. If the this parameter is
        larger than 0, the value of offset MUST be greater than 0."
}

leaf psid {
    type uint16;
    mandatory true;
    description
        "Port Set Identifier (PSID) value, which identifies a set
        of ports algorithmically."
}

leaf psid-len {
    type uint8 {
        range 0..16;
    }
    mandatory true;
    description
        "The length of PSID, representing the sharing ratio for an
        IPv4 address."
}

grouping binding-entry {
    description
        "The lwAFTR maintains an address binding table that contains
        the binding between the lwB4’s IPv6 address, the allocated IPv4
        address and restricted port-set."
    leaf binding-ipv6info {
        type union {
            type inet:ipv6-address;
            type inet:ipv6-prefix;
        }
        mandatory true;
        description
            "The IPv6 information for a binding entry.
            If it’s an IPv6 prefix, it indicates that
            the IPv6 source address of the lwB4 is constructed
            according to the description in RFC7596;
            if it’s an IPv6 address, it means the lwB4 uses
any /128 address from the assigned IPv6 prefix.

leaf binding-ipv4-addr {
    type inet:ipv4-address;
    mandatory true;
    description
        "The IPv4 address assigned to the lwB4, which is
         used as the IPv4 external address
         for lwB4 local NAPT44.";
}

container port-set {
    description
        "For Lightweight 4over6, the default value
         of offset should be 0, to configure one contiguous
         port range.";
    uses port-set {
        refine offset {
            default "0";
        }
    }
}

leaf lwaftr-ipv6-addr {
    type inet:ipv6-address;
    mandatory true;
    description
        "The IPv6 address for lwaftr.";
}

leaf lifetime {
    type uint32;
    units seconds;
    description "The lifetime for the binding entry";
}

/*
grouping nat-table {

description
    "Grouping ‘nat-table’ is not extended. The current mechanism
    is focusing on the provisioning of external IP address and
    port set; other NAT-specific considerations are out of scope for
    this model."
}
*/

grouping map-instance {
    description "A map-instance could be a MAP-CE or a MAP-BR";
leaf id {
    type uint32;
    mandatory true;
    description "MAP Instance ID";
}
leaf name {
    type string;
    description "MAP Instance Name";
}
container map-rules {
    description "A MAP instance could be configured with multiple sets of MAP rules";
    list map-rule {
        key "id";
        description "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.";
        leaf id {
            type uint8;
            description "Rule ID";
        }
        leaf forwarding {
            type boolean;
            mandatory true;
            description "This parameter specifies whether the rule may be used for forwarding (FMR). If set, this rule is used as an FMR; if not set, this rule is a BMR only and MUST NOT be used for forwarding.";
        }
        leaf map-rule-type {
            mandatory true;
            type enumeration {
                enum "BMR";
                enum "FMR";
            }
            description "The BMR and FMR share the rule format. BMR is used for a node to configure itself with IPv4 information retrieved from the rule. FMR is designed for the in-domain 4-in-6 routing, used in mesh mode. A BMR can be FMR in some case. The DMR for map-t is defined separately.";
        }
    }
}

Sun, et al. Expires April 8, 2016 [Page 18]
leaf rule-ipv6-prefix {
  type inet:ipv6-prefix;
  mandatory true;
  description "The Rule IPv6 prefix defined in the mapping rule.";
}
leaf rule-ipv4-prefix {
  type inet:ipv4-prefix;
  mandatory true;
  description "The Rule IPv4 prefix defined in the mapping rule.";
}
container port-set {
  description "Port set parameters specify a set of port ranges. For MAP, the default value of offset is 6.";
  uses port-set {
    refine offset {
      default "6";
    }
  }
}
leaf ea-len {
  type uint8;
  mandatory true;
  description "Embedded Address (EA) bits are the IPv4 EA-bits in the IPv6 address identify an IPv4 prefix/address (or part thereof) or a shared IPv4 address (or part thereof) and a port-set dentifier. The length of the EA-bits is defined as part of a MAP rule for a MAP domain.";
}

grouping traffic-stat {
  description "Traffic statistics";
  leaf sentPacket {
    type yang:zero-based-counter64;
    description "Number of packets sent.";
  }
  leaf sentByte {
    type yang:zero-based-counter64;
    description "Traffic sent, in bytes";
  }
}
leaf rcvdPacket {
  type yang:zero-based-counter64;
  description "Number of packets received.";
}
leaf rcvdByte {
  type yang:zero-based-counter64;
  description "Traffic received, in bytes";
}
leaf droppedPacket {
  type yang:zero-based-counter64;
  description "Number of packets dropped.";
}
leaf droppedByte {
  type yang:zero-based-counter64;
  description "Traffic dropped, in bytes";
}

*/
* Configuration Data Nodes
*/

container softwire-config {
  description
    "The configuration data for Softwire instances. And the shared
    data describes the softwire data model which is common to all of
    the different softwire mechanisms, such as description."
  leaf description {
    type string;
    description
      "A textual description of Softwire.";
  }
container lw4over6 {
  if-feature lw4over6;
  description
    "lw4over6 configuration.";
container lwaftr {
  if-feature lwaftr;
  description
    "Indicate this instance supports the lwAFTR function. The
    instances advertise the lwaftr feature through the
    capability exchange mechanism when a NETCONF session is
    established.";
  leaf enable {
    type boolean;
  }
description
   "Enable/disable the lwAFTR function.";
}
container lwaftr-instances {
   description
   "A set of lwAFTRs to be configured.";
   list lwaftr-instance {
      key "id";
      description
      "A set of lwAFTRs to be configured.";
      leaf id {
         type uint32;
         description "An instance identifier.";
      }
      leaf name {
         type string;
         description "The name for the lwaftr.";
      }
      leaf softwire-num-threshold {
         type uint32;
         mandatory true;
         description
         "The maximum number of tunnels that can be created on
         the lwAFTR.";
      }
      leaf tunnel-mtu {
         type uint16;
         mandatory true;
         description
         "The MTU for Lightweight 4over6 tunnel.";
      }
      leaf fragment-mru {
         type uint16;
         mandatory true;
         description
         "The fragmentation MRU for Lightweight 4over6
         tunnel.";
      }
      container binding-table {
         description "id";
         list binding-entry {
            key "binding-ipv6info";
            description "id";
            uses binding-entry;
         }
      }  
   }
}

container lwb4 {
  if-feature lwb4;
  description
      "Indicate this instance supports the lwB4 function. The instances advertise the lwb4 feature through the capability exchange mechanism when a NETCONF session is established."
  leaf enable {
    type boolean;
    description
      "Enable/disable the lwB4 function.";
  }
  container lwb4-instances {
    description
      "A set of lwB4s to be configured.";
    list lwb4-instance {
      key "binding-ipv6info";
      description "id";
      leaf name {
        type string;
        description "The lwb4 name.";
      }
      leaf tunnel-mtu {
        type uint16;
        mandatory true;
        description
          "The MTU for Lightweight 4over6 tunnel.";
      }
      leaf fragment-mru {
        type uint16;
        mandatory true;
        description
          "The fragment MRU for Lightweight 4over6 tunnel.";
      }
      leaf b4-ipv6-addr-format {
        type boolean;
        mandatory true;
        description
          "The format of lwB4 IPv6 address. If set to true, it indicates that the IPv6 source address of the lwB4 is constructed according to the description in [RFC7596]; if set to false, the lwB4 can use any /128 address from the assigned IPv6 prefix.";
      }
      uses binding-entry;
    }
  }
}
container map-e {
  if-feature map-e;
  description
    "Indicate the instances support the MAP-E function. The
    instances advertise the map-e feature through the capability
    exchange mechanism when a NETCONF session is established."
  leaf enable {
    type boolean;
    default "true";
    description
      "Enable/disable the MAP-E function.";
  }
  container map-e-instances {
    description
      "A set of MAP-E instances to be configured,
      applying to BRs and CEs.";
    list map-e-instance {
      key "id";
      description "id";
      uses map-instance;
      leaf tunnel-mtu {
        type uint16;
        mandatory true;
        description
          "The MTU for MAP-E tunnel.";
      }
      leaf fragment-mru {
        type uint16;
        mandatory true;
        description
          "The fragment MRU for MAP-E tunnel.";
      }
      leaf br-ipv6-addr {
        type inet:ipv6-address;
        mandatory true;
        description
          "The IPv6 address of the MAP-E BR.";
      }
    }
  }
}

container map-t {
  if-feature map-t;
  description

"Indicate the instances support the MAP-T function. The instances advertise the map-t feature through the capability exchange mechanism when a NETCONF session is established."

leaf enable {
  type boolean;
  default "true";
  description
    "Enable/disable the MAP-T function."
}

container map-t-instances {
  description
    "A set of the MAP-T instances to be configured, applying to BRs and CEs."
  list map-t-instance {
    key "id";
    description "id";
    uses map-instance;
    leaf dmr-ipv6-prefix {
      type inet:ipv6-prefix;
      description
        "The IPv6 prefix of the MAP-T BR."
    }
  }
}

/*
 * Operational state Data Nodes
 */

container softwire-state {
  config false;
  description
    "The operational state data for Softwire instances."
  leaf description {
    type string;
    description
      "A textual description of the softwire instances."
  }
  container lw4over6 {
    if-feature lw4over6;
    description
      "lw4over6 state."
    container lwafr {
      if-feature lwafr;
      config false;
      description
      ""
"Indicate this instance supports the lwAFTR function. The instances advertise the lwaftr feature through the capability exchange mechanism when a NETCONF session is established."

container lwaftr-instances {
    description "A set of lwAFTRs.";
    list lwaftr-instance {
        key "id";
        description "id";
        leaf id {
            type uint32;
            description "id";
        }
        leaf name {
            type string;
            description "The name for this lwaftr.";
        }
        uses traffic-stat;
        leaf active-softwire-num {
            type uint32;
            description "The number of currently active tunnels on the lw4over6 instance.";
        }
    }
    container binding-table {
        description "id";
        list binding-entry {
            key "binding-ipv6info";
            description "An identifier of the binding entry.";
            leaf binding-ipv6info {
                type union {
                    type inet:ipv6-address;
                    type inet:ipv6-prefix;
                }
                mandatory true;
                description "The IPv6 information used to identify a binding entry."
            }
            leaf active {
                type boolean;
                description "Status of a specific tunnel.";
            }
        }
    }
}

}
}

container lwb4 {
  if-feature lwb4;
  config false;
  description
    "Indicate this instance supports the lwb4 function. The
    instances advertise the lwb4 feature through the
    capability exchange mechanism when a NETCONF session is
    established.";
  container lwb4-instances {
    description
      "Status of the configured lwb4s.";
    list lwb4-instance {
      key "binding-ipv6info";
      description "a lwb4 instance.";
      leaf name {
        type string;
        description "The lwb4 name.";
      }
      leaf binding-ipv6info {
        type union {
          type inet:ipv6-address;
          type inet:ipv6-prefix;
        }
        mandatory true;
        description
          "The IPv6 information used to identify
          a binding entry. ";
        }
        uses traffic-stat;
      }
  }
  }
}

container map-e {
  if-feature map-e;
  config false;
  description
    "Indicate the instances support the MAP-E function. The
    instances advertise the map-e feature through the capability
    exchange mechanism when a NETCONF session is established.";
  container map-e-instances {
    description
      "Status of MAP-E instance(s).";
list map-e-instance {
  key "id";
  description "id";
  leaf id {
    type int32;
    description "id";
  }
  leaf name {
    type string;
    description "The map-e instance name.";
  }
  uses traffic-stat;
}

container map-t {
  if-feature map-t;
  config false;
  description "Indicate the instances support the MAP-T function. The instances advertise the map-t feature through the capability exchange mechanism when a NETCONF session is established."
  container map-t-instances {
    description "Status of MAP-T instances.";
    list map-t-instance {
      key "id";
      description "id";
      leaf id {
        type int32;
        description "";
      }
      leaf name {
        type string;
        description "The map-t instance name.";
      }
      uses traffic-stat;
    }
  }
}

/*
 * Notifications
 */
notification softwire-lwaftr-event {
  if-feature lw4over6;
  if-feature lwaftr;
  description "Notification for lwaftr.";

  leaf lwaftr-id {
    type leafref {
      path
        "/softwire-state/lw4over6/lwaftr/lwaftr-instances/"
        + "lwaftr-instance/id";
      }
    description "id";
  }

  leaf-list invalid-entry {
    type leafref {
      path
        "/softwire-config/lw4over6/lwaftr/lwaftr-instances/"
        + "lwaftr-instance[id=current()]/../lwaftr-id/"
        + "binding-table/binding-entry/binding-ipv6info";
      }
    description "Notify the client that a specific binding entry has been expired/invalid. The binding-ipv6info identifies an entry.";
  }

  leaf-list added-entry {
    type inet:ipv6-address;
    description
      "Notify the client that a binding entry has been added. The ipv6 address of that entry is the index. The client get other information from the lwaftr about the entry indexed by that ipv6 address."
    }

  leaf-list modified-entry {
    type leafref {
      path
        "/softwire-config/lw4over6/lwaftr/lwaftr-instances/"
        + "lwaftr-instance[id=current()]/../lwaftr-id/"
        + "binding-table/binding-entry/binding-ipv6info";
      }
    description "lwaftr";
  }
}

notification softwire-lwb4-event {
  if-feature lw4over6;
  if-feature lwb4;
  description "lwb4 notification";
}
leaf lwb4-binding-ipv6-addr-change {
  type inet:ipv6-address;
  mandatory true;
  description "The source tunnel IPv6 address of the lWB4. If 'b4-ipv6-addr-format' is false, or the lwb4’s binding-ipv6-address changes for any reason, it SHOULD notify the NETCONF client.";
}

notification softwire-map-e-event {
  if-feature map-e;
  description "Notifications for MAP-E.";
  leaf map-e-id {
    type leafref {
      path "/softwire-config/map-e/map-e-instances/map-e-instance/id";
    } mandatory true;
    description "MAP-E event.";
  }
  leaf-list invalid-entry-id {
    type leafref {
      path "/softwire-config/map-e/map-e-instances/" + "map-e-instance[id=current()/../map-e-id]/map-rules/" + "map-rule/id";
    } description "Invalid entry event.";
  }
  leaf-list added-entry {
    type uint32;
    description "Added entry.";
  }
  leaf-list modified-entry {
    type leafref {
      path "/softwire-config/map-e/map-e-instances/" + "map-e-instance[id=current()/../map-e-id]/map-rules/" + "map-rule/id";
    } description "Modified entry.";
  }
}

notification softwire-map-t-event {
  if-feature map-t;
5. Example of Configure Lw4over6 Binding-Table

The lwAFTR maintains an address binding table which contains the following 3-tuples:

- IPv6 Address for a single lwB4
- Public IPv4 Address
- Restricted port-set
The entry has two functions: the IPv6 encapsulation of inbound IPv4 packets destined to the lwB4 and the validation of outbound IPv4-in-IPv6 packets received from the lwB4 for de-capsulation.

Requirement: Add an entry that maintain the relationship between 3-tuples of lwB4 (2001::1) in binding-table, which on the lwAFTR (2001::2). The data value of this 3-tuples are '2001::1', '123.1.1.1' and '1234' respectively.

Here is the example binding-table configuration xml:

```xml
<rpc message-id="101"
     xmlns:nc="urn:params:xml:ns:yang:ietf-softwire:1.0">
   <!-- replace with IANA namespace when assigned. -->
   <edit-config>
     <target>
       <running/>
     </target>
     <softwire-config>
       <lw4over6-aftr>
         <lw4over6-aftr-instances>
           <lw4over6-aftr-instance>
             <aftr-ipv6-addr>2001::2</aftr-ipv6-addr>
             <binding-table>
               <binding-entry>
                 <binding-ipv4-addr>123.1.1.1</binding-ipv4-addr>
                 <port-set>
                   <psid>1234</psid>
                 </port-set>
                 <binding-ipv6-addr>2001::1</binding-ipv6-addr>
                 <active>1</active>
               </binding-entry>
             </binding-table>
           </lw4over6-aftr-instance>
           <lw4over6-aftr-instance>
             <aftr-ipv6-addr>2001::2</aftr-ipv6-addr>
             <binding-table>
               <binding-entry>
                 <binding-ipv4-addr>123.1.1.1</binding-ipv4-addr>
                 <port-set>
                   <psid>1234</psid>
                 </port-set>
                 <binding-ipv6-addr>2001::1</binding-ipv6-addr>
                 <active>1</active>
               </binding-entry>
             </binding-table>
           </lw4over6-aftr-instance>
         </lw4over6-aftr-instances>
       </lw4over6-aftr>
     </softwire-config>
   </edit-config>
</rpc>
```

Figure 6: Lw4over6 Binding-Table Configuration XML

6. Security Considerations

The YANG module defined in this memo is designed to be accessed via the NETCONF protocol [RFC6241]. The lowest NETCONF layer is the secure transport layer and the mandatory to implement secure transport is SSH [RFC6242]. The NETCONF access control model [RFC6536] provides the means to restrict access for particular
NETCONF users to a pre-configured subset of all available NETCONF protocol operations and content.

There are a number of data nodes defined in this YANG module which are writable/creatable/deletable (i.e. config true, which is the default). These data nodes may be considered sensitive or vulnerable in some network environments. Write operations (e.g. edit-config) to these data nodes without proper protection can have a negative effect on network operations. These are the subtrees and data nodes and their sensitivity/vulnerability:

subtrees and data nodes and state why they are sensitive

Some of thereadable data nodes in this YANG module may be considered sensitive or vulnerable in some network environments. It is thus important to control read access (e.g. via get, get-config or notification) to these data nodes. These are the subtrees and data nodes and their sensitivity/vulnerability:

subtrees and data nodes and state why they are sensitive

7. IANA Considerations

A registry for standard YANG modules shall be set up. This document registers one URI for the YANG XML namespace in the IETF XML registry [RFC3688].


8. Acknowledgements

The authors would like to thank Lishan Li, Bert Wijnen, Giles Heron and Ole Troan for their contributions to this work.

9. References

9.1. Normative References


9.2. Informative References

[I-D.sivakumar-yang-nat]
Sivakumar, S., Boucadair, M., and S. <>, "YANG Data Model for Network Address Translation (NAT)", draft-sivakumar-yang-nat-00 (work in progress), September 2015.
Authors’ Addresses

Qi Sun
Tsinghua University
Beijing 100084
P.R. China
Phone: +86-10-6278-5822
Email: sunqi.ietf@gmail.com

Hao Wang
Tsinghua University
Beijing 100084
P.R. China
Phone: +86-10-6278-5822
Email: wangh13@mails.tsinghua.edu.cn

Yong Cui
Tsinghua University
Beijing 100084
P.R. China
Phone: +86-10-6260-3059
Email: yong@csnet1.cs.tsinghua.edu.cn

Ian Farrer
Deutsche Telekom AG
CTO-ATI, Landgrabenweg 151
Bonn, NRW 53227
Germany
Email: ian.farrer@telekom.de

Mohamed Boucadair
France Telecom
Rennes 35000
France
Email: mohamed.boucadair@orange.com
A Redundancy Mechanism for Dual-Stack Lite

draft-xu-v6ops-dslite-redundancy-00

Abstract

Dual-Stack Lite is a solution to offer both IPv4 and IPv6 connectivity to customers that are addressed only with an IPv6 prefix. This document provides a redundancy mechanism for Dual-Stack Lite.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on April 21, 2016.

Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of
1. Introduction

Dual-Stack Lite [RFC6333] is a solution to offer both IPv4 and IPv6 connectivity to customers crossing an IPv6 only infrastructure. The internet service provider no longer to provide public IPv4 address but an IPv6 prefix to the customers as the issue of the IPv4 public address shortage. One of its key components is an IPv4-over-IPv6 tunnel, which is used to provide IPv4 connectivity across a service provider’s IPv6 network. Another key component is a carrier-grade IPv4-IPv4 Network Address Translation (NAT) to share service provider IPv4 addresses among customers. As the exhaustion of the public IPv4 address, service providers have deployed DS-Lite in their network widely in nowadays, where a large number of customers are located. These customers within a network which is served by a single CGN function embeded in AFTR element may experience service degradation due to the presence of the single point of failure or loss of state information. Therefore, redundancy capabilities of the AFTR devices are strongly desired in order to deliver highly available services to customers. Failure detection and repair time must be therefore shortened.

This document describes a redundancy mechanism for DS-Lite. Some deployment consideration and recommendations for network elements are also provided.
2. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119] when they appear in ALL CAPS. When these words are not in ALL CAPS (such as "should" or "Should"), they have their usual English meanings, and are not to be interpreted as [RFC2119] key words.

3. Reliability Considerations of AFTR

As described in [RFC6908], for the robustness, reliability, and load distribution purposes, operators may deploy multiple AFTRs in their network. There are many deployment mechanism for the AFTR in ISP network, the most common type are distribution mode and centralization mode.

For the distribution mode, the CGN card is integrated into the free slot of the BRAS in a metro network. As the BRAS integrates the AFTR function of DS-lite, it provides DS-Lite connection service for a small area customers in this metro network. The service providers always integrated two CGN cards in the BRAS for redundancy consideration as the primary AFTR and backup AFTR. The capital cost of this mode is expensive because it always need two CGN cards for every BRAS. But 50 percent of these cards are idle most of time so that it is a big waste of money. There are various types and versions of BRAS have been deployed in the service provider’s network. Some of them have been used for over ten years and may not support the card insertion. Some of them may also don’t have free slot for the CGN card. It is not operational to replace all of them in a short period which result that it could deploy DS-Lite in some area and others can not in the same metro network.

For the centralization mode, a stand-alone AFTR device is deployed nearby the core router device at the exit of a metro network. It provides the DS-Lite connection service for the whole customers in this metro network. Service providers always deploy two stand-alone AFTR devices nearby the two core router device for the load distribution and redundancy purpose. The capital cost of this mode is more less than the distribution mode. It does not consume the slot resource of the BRAS. But it takes a big challenge for AFTR device for this mode in the large scale metro network because it takes performance requirements for the speed of the session creation and the maximum number of session maintenance. On the other side, it will create extra traffic when the users belong to the same BRAS are communicating with each other because it will connect to the AFTR device in the centralization mode first. It is a waste of bandwidth.
As described above, whether to use distribution mode or centralization mode depends on the trade-off between the investment and operational efficiency requirement of the service providers.

4. The Redundancy Mechanism Overview

The fundamental principle of redundancy mechanism in this document is to make the centralization mode to backup for the distribution mode. The architecture of the redundancy mechanism is illustrated as Figure 1. It deploys one AFTR card into every BRAS which supports card insertion in metro network, as to provide basic distributed DS-Lite connection service. Moreover, it deploys two stand-alone AFTR devices near the core router at the exit of the metro network. So it could provide the DS-Lite connection service for the users of the BRAS which don’t support card insertion and don’t have free slot for the AFTR card. One advantage of this mechanism is that the stand-alone AFTR device is not only a redundancy device but also can provide DS-Lite connection service for the BRAS without AFTR card slot. Then the IGP routing would be configured on the BRAS which has the AFTR card insertion.
It is made that the routing prior selected to the AFTR card on the BRAS and then selected the AFTR stand-alone device near the core router through the Metric value configuration. As the metric values of the two stand-alone AFTR device in centralization mode are the same, it ensure that the traffic of the same session would be forwarded to the same centralized AFTR device by the random selection of the hash algorithm. This mechanism is based on the IPv6 anycast function: when the AFTR card in distribution mode is breakdown, the AFTR address in router advertise message will disappear in the IGP routing table. The IP address of AFTR device in centralization mode is becoming the optimal routing. All the traffic for DS-Lite will be directed to the AFTR device in centralization mode as to keep the application alive.

5. The difference between the software process of the BRAS

The software process of the BRAS for distribution mode is described as Figure 2.
The traffic for IPv6 is flow into the inbound card.

Decapsulation for PPP

Look for IPv6 FIB

Whether the AFTR Card Breakdown? Y PPP Session Interruption

N The local AFTR decapsulated the packets from the IPv6 tunnel The DS-Lite Service are Terminated

NAT44

Look for IPv4 FIB

The traffic for IPv4 flow out to the outbound card
Figure 2: The software process of the BRAS for distribution mode

And the software process of the BRAS for the new mechanism is described as Figure 3:
As compared between Figure 2 and Figure 3, the main difference for the new mechanism is that if the local AFTR card breakdown, the DS-Lite service can be maintained as the backup AFTR will take over the function to keep the application alive.

6. New requirements for the AFTR device

For this DS-Lite redundancy mechanism, there are some new requirements for the AFTR device as below:

1. If the distribution AFTR card breakdown, the AFTR device SHOULD ensure that the traffic will not direct to the other distribution AFTR card.

2. It should use FQDN to describe the AFTR in the DHCPv6 option as described in [RFC6334].

3. How many distribution AFTR device could be covered by one centralization AFTR device will be different depends on the deployment by different ISPs.

4. The speed of the session creation for the centralized AFTR device could be calculated by a formula.
7. Security Considerations

TBD.

8. IANA Considerations

This draft does not request any IANA action.

9. Acknowledgements

The authors would like to thank the valuable comments made by XXX and other members of v6ops WG.

This document was produced using the xml2rfc tool [RFC2629].

10. References

10.1. Normative References


10.2. Informative References


Authors' Addresses

Honglei Xu
China Telecom
No.118 Xizhimennei street, Xicheng District
Beijing, 100035
P.R. China

Email: xuhl@ctbri.com.cn

Qiong Sun
China Telecom
No.118 Xizhimennei street, Xicheng District
Beijing 100035
P.R. China

Email: sunqiong@ctbri.com.cn

Xiaodong Lee
CNNIC
No.4 South 4th Street, Zhongguancun
Beijing, 100190
P.R. China

Email: xl@cnnic.cn

Yu Fu
CNNIC
No.4 South 4th Street, Zhongguancun
Hai-Dian District, Beijing, 100190
P.R. China

Email: fuyu@cnnic.cn