DMM
Internet-Draft
Intended status: Informational
Expires: January 9, 2017

H. Chan
X. Wei
Huawei Technologies
J. Lee
Sangmyung University
S. Jeon
Instituto de Telecomunicacoes
F. Templin
Boeing Research and Technology

July 8, 2016

Distributed Mobility Anchoring
draft-chan-dmm-distributed-mobility-anchoring-08

Abstract

This document defines distributed mobility anchoring. Multiple anchors and nodes are configured with appropriate mobility functions and work together to enable mobility solutions. Example solution is mid-session switching of the IP prefix anchor. Without ongoing session requiring session continuity, a flow can be started or re-started using the new IP prefix which is allocated from the new network and is therefore anchored to the new network. With ongoing session, the anchoring of the prior IP prefix may be relocated to the new network to enable session continuity.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

A key requirement in distributed mobility management [RFC7333] is to enable traffic to avoid traversing single mobility anchor far from the optimal route. Distributed mobility management solutions do not make use of centrally deployed mobility anchor [Paper-Distributed.Mobility]. As such, the traffic of a flow SHOULD be able to change from traversing one mobility anchor to traversing another mobility anchor as the mobile node moves, or when changing operation and management requirements call for mobility anchor switching, thus avoiding non-optimal routes. This draft proposes distributed mobility anchoring to enable making such route changes.

Distributed mobility anchoring employs multiple anchors in the data plane. In general, the control plane function may be separate from the data plane functions and be centralized but may also co-located with the data plane function at these distributed anchors. Different configurations (Section 3.1) of distributed anchoring are then possible. Yet the distributed anchors need to have expected behaviors (Section 3.2).

A mobile node (MN) attached to an access router of a network may be allocated an IP prefix which is anchored to that router. It may then use the IP address configured from this prefix as the source IP address to run a flow with its correspondent node (CN). When there are multiple anchors, the flow may need to select the anchor when it is initiated (Section 4). Using an anchor in MN’s network of attachment has the advantage that the packets can simply be forwarded according to the forwarding table. Although the anchor is in the MN’s network of attachment when the flow was initiated, the MN may later move to another network, so that the IP address no longer belongs to the new network of attachment of the MN. Whether the flow needs session continuity will determine how to ensure that the IP address of the flow will be anchored to the new network of attachment. If the ongoing IP flow can cope with an IP prefix/ address change, the flow can be reiniated with a new IP address anchored in the new network (Section 4.1.1). On the other hand, if the ongoing IP flow cannot cope with such change, the IP address anchoring can be moved from the original network to the new network (Section 4.2).

2. Conventions 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].

All general mobility-related terms and their acronyms used in this document are to be interpreted as defined in the Mobile IPv6 base specification [RFC6275], the Proxy Mobile IPv6 specification [RFC5213], and the DMM current practices and gap analysis [RFC7429]. This includes terms such as mobile node (MN), correspondent node

(CN), home agent (HA), home address (HoA), care-of-address (CoA), local mobility anchor (LMA), and mobile access gateway (MAG).

In addition, this document uses the following term:

Home network of an application session (or of an HoA): the network that has allocated the IP address (HoA) used for the session identifier by the application running in an MN. An MN may be running multiple application sessions, and each of these sessions can have a different home network.

IP prefix/address anchoring: An IP prefix, i.e., Home Network Prefix (HNP), or address, i.e., Home Address (HoA), allocated to a mobile node is topologically anchored to a node when the anchor node is able to advertise a connected route into the routing infrastructure for the allocated IP prefix.

Internetwork Location Management (LM) function: managing and keeping track of the internetwork location of an MN. The location information may be a binding of the IP advertised address/prefix, e.g., HoA or HNP, to the IP routing address of the MN or of a node that can forward packets destined to the MN. It is a control plane function.

In a client-server protocol model, location query and update messages may be exchanged between a Location Management client (LMc) and a Location Management server (LMs).

With separation of control plane and data plane, the LM function is in the control plane. It may be a logical function at the control plane node, control plane anchor, or mobility controller.

It may be distributed or centralized.

Forwarding Management (FM) function: packet interception and forwarding to/from the IP address/prefix assigned to the MN, based on the internetwork location information, either to the destination or to some other network element that knows how to forward the packets to their destination.

This function may be used to achieve indirection. With separation of control plane and data plane, FM may split into a FM function in the data plane (FM-DP) and a FM function in the control plane (FM-CP).

FM-DP may be distributed with distributed mobility management. It may be a function in a data plane anchor or data plane node.

FM-CP may be distributed or centralized. It may be a function in a control plane node, control plane anchor or mobility controller.

Security Management (SM) function: The security management function controls security mechanisms/protocols providing access control, integrity, authentication, authorization, confidentiality, etc. for the control plane and data plane.

This function resides in all nodes such as control plane anchor, data plane anchor, mobile node, and correspondent node.

3. Distributed anchoring

3.1. Distributed anchoring configurations

The mobility functions may be implemented in different configurations of distributed anchoring in architectures separating the control and data planes. The separation as described in [I-D.wt-dmm-deployment-models] has defined home control plane anchor (Home-CPA), home data plane anchor (Home-DPA), access control plane node (Access-CPN), and access data plane node (Access-DPN), which are respectively abbreviated as CPA, DPA, CPN, and DPN here. Some configurations are described in [I-D.sijeon-dmm-deployment-models].

Figure 1 shows 4 configurations of network-based mobility management. In each configuration, an MN is allocated an IP prefix/address IP1 and is using IP1 to communicate with a correspondent node (CN) not shown in the figure. The flow of this communication session is shown as flow(IP1, ...) which uses IP1 and other parameters.

```
       (a)                (b)                (c)                (d)
                           +-----+                               +-----+
                           |LMs  |                               |LMs  |
                           +-----+                               +-----+

 +------------+     +------------+     +------------+     +------------+
 |CPA:        |     |CPA:        |     |CPA:        |     |CPA:        |
 |FM-CP, LM   |     |FM-CP, LMc  |     |FM-CP, LMs  |     |FM-CP, LMp  |
 +------------+     +------------+     +------------+     +------------+
 +------------+     +------------+     +------------+     +------------+
 |DPA(IPa1):  |     |DPA(IPa1):  |     |DPA(IPa1):  |     |DPA(IPa1):  |
 |anchors IP1 |     |anchors IP1 |     |anchors IP1 |     |anchors IP1 |
 |FM-DP       |     |FM-DP       |     |FM-DP       |     |FM-DP       |
 +------------+     +------------+     +------------+     +------------+


                                       +------------+     +------------+
                                       |CPN:        |     |CPN:        |
                                       |FM-CP, LMc  |     |FM-CP, LMc  |
                                       +------------+     +------------+
                                       +------------+     +------------+
                                       |DPN(IPn1):  |     |DPN(IPn1):  |
                                       |FM-DP       |     |FM-DP       |
                                       +------------+     +------------+


 +------------+     +------------+     +------------+     +------------+
 |MN(IP1)     |     |MN(IP1)     |     |MN(IP1)     |     |MN(IP1)     |
 |flow(IP1,..)|     |flow(IP1,..)|     |flow(IP1,..)|     |flow(IP1,..)|
 +------------+     +------------+     +------------+     +------------+
```

Figure 1. (a) FM-CP and LM at CPA, FM-DP at DPA; (b) Separate LMs, FM-CP and LMc at CPA, FM-DP at DPA; (c) FM-CP and LMs at CPA, FM-DP at DPA, FM-CP and LMc at CPN, FM-DP at DPN; (d) Separate LMs, FM-CP and LMp at CPA, FM-DP at DPA, FM-CP and LMc at CPN, FM-DP at DPN.

In Figures 1(a), 1(b), 1(c), and 1(d), the IP address of the MN, IP1, is anchored to the DPA which has the IP prefix/address IPa1. The data plane is distributed so that there may be multiple instances of the DPA (not shown). The control plane may either be distributed or centralized. When the CPA co-locates with the distributed DPA there will be multiple instances of the co-located CPA and DPA (not shown).

In Figure 1(a) and Figure 1(b), the network is flat with FM-DP at the distributed DPA.

In Figure 1(a), LM and FM-CP co-locate at CPA. Then LM may be distributed or centralized according to whether the CPA is distributed or centralized.

Figure 1(b) differs from Figure 1(a) in that the LM function is split into a server LMs and a client LMc. LMc and FM-CP are at the CPA. The LMs may be centralized whereas the LMc may be distributed or centralized according to whether the CPA is distributed or centralized.

In Figure 1(c) and Figure 1(d), the network is hierarchical where there may be multiple DPN’s for each DPA. There is FM-DP at each of the distributed DPA and at each of the distributed DPN.

In Figure 1(c), LMs and FM-CP are at the CPA. In addition, there are FM-CP and LMc at the CPN. Again, LMs may be distributed or centralized according to whether the CPA is distributed or centralized. The CPA may co-locate with DPA or may separate.

Figure 1(d) differs from Figure 1(c) in that the LMs is separated out, and a proxy LMp is added between the LMs and LMc. LMp and FM-CP are at the CPA. Again, there are FM-CP and LMc at the CPN. The LMs may be centralized whereas the LMp may be distributed or centralized according to whether the CPA is distributed or centralized.

Host-based variants of the mobility function configurations from Figures 1(c) and 1(d) are shown in Figures 2(a) and 2(b) where the role to perform mobility functions by CPN and DPN are now taken by the MN. The MN then need to possess the mobility functions FM and LMc.

```
         (a)                (b)
                             +-----+
                             |LMs  |
                             +-----+

   +------------+     +------------+
   |CPA:        |     |CPA:        |
   |FM-CP, LMs  |     |FM-CP, LMp  |
   +------------+     +------------+
   +------------+     +------------+
   |DPA(IPa1):  |     |DPA(IPa1):  |
   |anchors IP1 |     |anchors IP1 |
   |FM-DP       |     |FM-DP       |
   +------------+     +------------+


   +------------+     +------------+
   |MN(IP1)     |     |MN(IP1)     |
   |flow(IP1,..)|     |flow(IP1,..)|
   |FM,    LMc  |     |FM,    LMc  |
   +------------+     +------------+
```

Figure 2. (a) FM-CP and LMs at CPA, FM-DP at DPA, FM and LMc at MN; (b) Separate LMs, FM-CP and LMp at CPA, FM-DP at DPA, FM and LMc at MN.

In Figure 2(a) and Figure 2(b), FM-DP is at the distributed DPA as before.

In Figure 2(a), LMs and FM-CP are at the CPA. The LMs may be distributed or centralized according to whether the CPA is distributed or centralized.

Figure 2(b) differs from Figure 2(a) in that the LMs is separated out and the proxy LMp is added between the LMs and LMc. LMp and FM-CP are at the CPA. The FMs may be centralized whereas the LMp may be distributed or centralized according to whether the CPA is distributed or centralized.

3.2. Distributed anchoring behaviors and message information elements

The behaviors of distributed anchoring are defined in this section in order that they may work together in expected manners to produce a distributed mobility solution. The needed information elements are passed as message parameters.

3.2.1. Location management behaviors and message information elements

It is seen in (Section 3.1) that

(1) LMs may be a separate server or may co-locate with LMc at CPA;
(2) LMc may be at CPA, CPN, or MN.

Example LM design may consists of a distributed database of LMs servers in a pool of distributed servers. The location information about the prefix/address of a MN is primarily at a given LMs. Peer LMs may exchange the location information with each other. LMc may retrieve a given record or send a given record update to LMs.

Location information bebaviors:

(LM:1) LM may manage the location information in a client-server database system. The example LM database functions are:

(LM:1-1) LMc may query LMs about location information for a prefix of MN (pull).
Parameters:
IP prefix of MN.

(LM:1-2) LMs may reply to LMc query about location information for a prefix of MN (pull).
Parameters:
IP prefix of MN,
IP address of FM-DP/DPA/DPN to forward the packets of the flow.

(LM:1-3) LMs may inform LMc about location information for a prefix of MN (push).
Parameters:
IP prefix of MN,
IP address of FM-DP/DPA/DPN to forward the packets of the flow.

(LM:1-4) LMc may inform LMs about update location information for a prefix of MN.
Parameters:
IP prefix of MN,
IP address of FM-DP/DPA/DPN to forward the packets of the flow.

(LM:2) The LM may be a distributed database with multiple LMs servers. For example:

(LM:2-1) A LMs may join a pool of LMs servers.

Parameters:
IP address of the LMs,
IP prefixes for which the LMs will host the primary location information.

(LM:2-2) LMs may query a peer LMs about location information for a prefix of MN.
Parameters:
IP prefix.

(LM:2-3) LMs may reply to a peer LMs about location information for a prefix of MN.
Parameters:
IP prefix of MN,
IP address of FM-DP/DPA/DPN to forward the packets of the flow.

3.2.2. Forwarding management behaviors and message information elements

It is seen in (Section 3.1) that

(1) FM-CP may be at CPA, CPN, MN;
(2) FM-DP may be at DPA, DPN, MN.

The FM behaviors and message information elements are:

(FM:1) With distributed FM functions, the role of FM for a flow may pass to another FM as the DPA or DPN changes.

(FM:2) In addition to above, a flow/session may be stateful for the required information for QoS, charging, etc. are needed. These states need to be transferred from the old anchor to the new anchor.

(FM:3) An anchor may act on packets on a per flow basis and perform the changes to the forwarding path upon a change of point of attachment of a MN:

(FM:3-1) FM filters the packets up to the granularity of a flow.
Example matching parameters are the 5-tuple of a flow.

(FM:3-2) FM makes the necessary changes to the forwarding path of a flow.
Example mechanism is through forwarding table update activated by DHCPv6-PD.

(FM:3-3) FM reverts the previously made changes to the forwarding path of a flow when such changes are no longer needed, e.g., when ongoing flows using an IP prefix/address requiring session continuity have closed.
Example mechanism is through expiration of DHCPv6-PD.

(FM:4) An anchor may discover and be discovered such as through an anchor registration system:

(FM:4-1) FM registers and authenticates itself with a centralized mobility controller.
Parameters:
IP address of DPA and its CPA;
IP prefix anchored to the DPA.

(FM:4-2) registration reply: acknowledge of registration and echo the input parameters.

(FM:4-3) FM discovers the FM of another IP prefix by querying the mobility controller based on the IP prefix.
Parameters:
IP prefix of MN.

(FM:4-4) when making anchor discovery FM expects the answer parameters as: IP address of DPA to which IP prefix of MN is anchored; IP prefix of the corresponding CPA.

(FM:5) With separation of control plane function and data plane function, these function must work together.

(FM:5-1) CPA/FM-CP sends forwarding table updates to DPA/FM-DP.
Parameters:
new forwarding table entries to add;
expired forwarding table entries to delete.

(FM:5-2) DPA/FM-DP sends to CPA/FM-CP about its status and load.
Parameters:
state of forwarding function being active or not;
loading percentage.

(FM:6) An anchor can buffer packets of a flow in a mobility event:

(FM:6-1) CPA/FM-CP informs DPA/FM-DP to buffer packets of a flow.
Trigger:
MN leaves DPA in a mobility event.
Parameters:
IP prefix of the flow for which packets need to be buffered.

(FM:6-2) CPA/FM-CP on behalf of a new DPA/FM-DP informs the CPA/FM-CP of the prior DPA/FM-DP that it is ready to receive any buffered packets of a flow.
Parameters:
destination IP prefix of the flow’s packets;
IP address of the new DPA.

4. Example mobility solutions with distributed anchoring

The IP prefix/address at the MN’s side of a flow may be anchored at the access router to which the MN is attached. For example, when an MN attaches to a network (Net1) or moves to a new network (Net2), it is allocated an IP prefix from that network. It configures from this prefix an IP address which is typically a dynamic IP address. It then uses this IP address when a flow is initiated. Packets to the MN in this flow are simply forwarded according to the forwarding table.

There may be multiple IP prefixes/addresses to choose from. They may be from the same access network or different access networks. The network may advertise these prefixes with cost options [I-D.mccann-dmm-prefixcost] so that the mobile node may choose the one with the least cost. In addition, these IP prefixes/addresses may be of different types regarding whether mobility support is needed [I-D.ietf-dmm-ondemand-mobility]. A flow will need to choose the appropriate one according to whether it needs IP mobility support.

4.1. IP mobility support only when needed

IP mobility support may be provided only when needed instead of being provided by default. The simplest configuration in this case is shown in Figures 1(a) and 1(b) in Section 3.1 for which the LM and FM functions are utilized only when needed.

A straightforward choice of mobility anchoring is for a flow to use the IP prefix of the network to which the MN is attached when the flow is initiated [I-D.seite-dmm-dma].

4.1.1. Not needed: Changing to the new IP prefix/address

When IP mobility support is not needed for a flow, the LM and FM functions are not utilized so that the configuration from Figures 1(a) and 1(b) in Section 3.1 simplifies to that shown in Figure 3.

```
Net1                                                   Net2

+---------------+                                      +---------------+
|AR1            |                                      |AR2            |
+---------------+                                      +---------------+
|CPA:           |                                      |CPA:           |
|---------------|                                      |---------------|
|DPA(IPa1):     |                                      |DPA(IPa2):     |
|anchors IP1    |                                      |anchors IP2    |
+---------------+                                      +---------------+


+...............+                                      +---------------+
.MN(IP1)        .                 move                 |MN(IP2)        |
.flow(IP1,...)  .               =======>               |flow(IP2,...)  |
+...............+                                      +---------------+
```

Figure 3. Changing to the new IP prefix/address. MN running a flow using IP1 in Net1 changes to running a flow using IP2 in Net2.

When there is no need to provide IP mobility to a flow, the flow may use a new IP address acquired from a new network as the MN moves to the new network.

Regardless of whether IP mobility is needed, if the flow has terminated before the MN moves to a new network, the flow may subsequently restart using the new IP address allocated from the new network.

When session continuity is needed, even if a flow is ongoing as the MN moves, it may still be desirable for the flow to change to using the new IP prefix configured in the new network. The flow may then close and then restart using a new IP address configured in the new network. Such a change in the IP address of the flow may be enabled using a higher layer mobility support which is not in the scope of this document.

In Figure 3, a flow initiated while the MN was in Net1 has terminated before the MN moves to a new network Net2. After moving to Net2, the MN uses the new IP prefix anchored in Net2 to start a new flow. The packets may then be forwarded without requiring IP layer mobility support.

The call flow is outlined in Figure 4.

```
 MN                     p-AR          n-AR                             CN
  |MN attaches to p-AR: |             |                                |
  |acquire MN-ID and profile          |                                |
  |--RS---------------->|             |                                |
  |                     |             |                                |
  |<----------RA(HNP1)--|             |                                |
  |                     |             |                                |
Allocated prefix HNP1
IP1 address configuration
  |                     |             |                                |
  |<-Flow(IP1,IPcn,...)-+--------------------------------------------->|
  |                     |             |                                |
  |MN detaches from p-AR|             |                                |
  |MN attaches to n-AR  |             |                                |
  |                     |             |                                |
  |--RS------------------------------>|                                |
  |                     |             |                                |
  |<--------------RA(HNP2)------------|                                |
  |                     |             |                                |
Allocated prefix HNP2
IP2 address configuration
  |                     |             |                                |
  |<-new Flow(IP2,IPcn,...)-----------+------------------------------->|
  |                     |             |                                |
```

Figure 4. A flow uses the IP allocated from the network at which the MN is attached when the flow is initiated.

The security management function in the anchor node at a new network must allow to assign a valid IP prefix/address to a mobile node.

4.1.2. Needed: Providing IP mobility support

When IP mobility is needed for a flow, the LM and FM functions in Figures 1(a) and 1(b) in Section 3.1 are utilized. The mobility support may be provided by IP prefix anchor switching to the new network to be described in Section 4.2 or by using other mobility management methods ([Paper-Distributed.Mobility.PMIP] and [Paper-Distributed.Mobility.Review]). Then the flow may continue to use the IP prefix from the prior network. Yet some time later, the user application for the flow may be closed. If the application is started again, the new flow may not need to use the prior network’s IP address to avoid having to invoke IP mobility support. This may be the case where a permanent IP prefix/address is not used. The flow may then use the new IP prefix in the network where the flow is being initiated. Routing is again kept simpler without employing IP mobility and will remain so as long as the MN has not moved away from that network.

The call flow in this case is outlined in Figure 5.

```
 MN                    p-AR          n-AR                             CN
  |MN attaches to p-AR: |             |                                |
  |acquire MN-ID and profile          |                                |
  |--RS---------------->|             |                                |
  |                     |             |                                |
  |<----------RA(HNP1)--|             |                                |
  |                     |             |                                |
Allocated prefix HNP1
IP1 address configuration
  |                     |             |                                |
  |<-Flow(IP1,IPcn,...)-+--------------------------------------------->|
  |                     |             |                                |
  |MN detach from p-AR  |             |                                |
  |MN attach to n-AR    |             |                                |
  |                     |             |                                |
  |--RS------------------------------>|                                |

IP mobility support such as that described in next sub-section
  |<--------------RA(HNP2,HNP1)-------|                                |
  |                     |             |                                |
  |<-Flow(IP1,IPcn,...)---------------+------------------------------->|
  |                     |             |                                |
Allocated prefix HNP2
IP2 address configuration
  |                     |             |                                |
Flow(IP1,IPcn) teminates
  |                     |             |                                |
  |<-new Flow(IP2,IPcn,...)-----------+------------------------------->|
  |                     |             |                                |
```

Figure 5. A flow uses the IP allocated from the network at which the MN is attached when the flow is initiated.

To provide IP mobility support with distributed anchoring, the distributed anchors may need to message with each other. When such messaging is needed, the anchors may need to discover each other as described in the FM behaviors and information elements (FM:2) in Section 3.2.2.

Then the anchors need to properly forward the packets of the flows as described in the FM behaviors and information elements (FM:1) in Section 3.2.2.

If there are in-flight packets toward the old anchor while the MN is moving to the new anchor, it may be necessary to buffer these packets and then forward to the new anchor after the old anchor knows that the new anchor is ready. Such are described in the FM behaviors and information elements (FM:4) in Section 3.2.2.

4.2. IP prefix/address anchor switching to the new network

The IP prefix/address anchoring may move without changing the IP prefix/address of the flow. Here the LM and FM functions in Figures 1(a) and 1(b) in Section 3.1 are implemented as shown in Figure 6.

```
Net1                                                   Net2

+---------------+                                      +---------------+
|AR1            |                                      |AR2            |
+---------------+                                      +---------------+
|CPA:           |                                      |CPA:           |
|LM:IP1<-->IPa2 |                                      |LM:IP1<-->IPa2 |
|---------------|                                      |---------------|
|DPA(IPa1):     |                                      |DPA(IPa2):     |
|anchors IP1    |                 move                 |anchors IP2,IP1|
|FM:DHCPv6-PD   |               =======>               |FM:DHCPv6-PD   |
+---------------+                                      +---------------+


+...............+                                      +---------------+
.MN(IP1)        .                 move                 |MN(IP2,IP1)    |
.flow(IP1,...)  .               =======>               |flow(IP1,...)  |
+...............+                                      +---------------+
```

Figure 6. IP prefix/address anchor switching to the new network. MN with flow using IP1 in Net1 continues to run the flow using IP1 as it moves to Net2.

As an MN with an ongoing session moves to a new network, the flow may preserve session continuity by moving the anchoring of the original IP prefix/address of the flow to the new network. An example is in the use of BGP UPDATE messages to change the forwarding table entries as described in [I-D.mccann-dmm-flatarch] and also for 3GPP Evolved Packet Core (EPC) network in [I-D.matsushima-stateless-uplane-vepc]. However, the response time and scalability of using a distributed routing protocol to update forwarding tables may be controversial.

Use of a centralized routing protocol with a centralized control plane as described in Section 4.2.1 will be more scalable.

The location management provides information about which IP prefix from an AR in the original network is being used by a flow in which AR in a new network. Such information needs to be deleted or updated when such flows have closed so that the IP prefix is no longer used in a different network. The LM behaviors are described in Section 3.2.1.

The FM functions are implemented through the DHCPv6-PD protocol. Here the anchor behavior to properly forward the packets for a flow as described in the FM behaviors and information elements FM:1 in Section 3.2.2 is realized by changing the anchor with DHCPv6-PD and also by reverting such changes later after the application has already closed and when the DHCPv6-PD timer expires. If there are in-flight packets toward the old anchor while the MN is moving to the new anchor, it may be necessary to buffer these packets and then forward to the new anchor after the old anchor knows that the new anchor is ready. Such are described in the FM behaviors and information elements FM:4 in Section 3.2.2. The anchors may also need to discover each other as described in the FM behaviors and information elements FM:2.

The security management function in the anchor node at a new network must allow to assign the original IP prefix/address used by the mobile node at the previous (original) network. As the assigned original IP prefix/address is to be used in the new network, the security management function in the anchor node must allow to advertise the prefix of the original IP address and also allow the mobile node to send and receive data packets with the original IP address.

The security management function in the mobile node must allow to configure the original IP prefix/address used at the previous (original) network when the original IP prefix/address is assigned by the anchor node in the new network. The security management function in the mobile node also allows to use the original IP address for the previous flow in the new network.

4.2.1. Centralized control plane

An example of IP prefix anchor switching is in the case where Net1 and Net2 both belong to the same operator network with separation of control and data planes ([I-D.liu-dmm-deployment-scenario] and [I-D.matsushima-stateless-uplane-vepc]), where the controller may send to the switches/routers the updated information of the forwarding tables with the IP address anchoring of the original IP prefix/address at AR1 moved to AR2 in the new network. That is, the IP address anchoring in the original network which was advertising the prefix will need to move to the new network. As the anchoring in the new network advertises the prefix of the original IP address in the new network, the forwarding tables will be updated so that packets of the flow will be forwarded according to the updated forwarding tables. The configuration in Figures 1(a) and 1(b) in Section 3.1 for which FM-CP and LM are centralized and FM-DP’s are distributed. applies here. Figure 7 shows its implementation where LM is a binding between the original IP prefix/address of the flow and the IP address of the new DPA, whereas FM uses the DHCPv6-PD protocol.

```
Net1                                                   Net2
+----------------------------------------------------------------------+
|                            CPA:                                      |
|                            LM:IP1<-->IPa2                            |
|                            FM-CP                                     |
+----------------------------------------------------------------------+


+---------------+                                      +---------------+
|AR1            |                                      |AR2            |
+---------------+                                      +---------------+
|DPA(IPa1):     |                                      |DPA(IPa2):     |
|anchors IP1    |                 move                 |anchors IP2,IP1|
|FM:DHCPv6-PD   |               =======>               |FM:DHCPv6-PD   |
+---------------+                                      +---------------+


+...............+                                      +---------------+
.MN(IP1)        .                 move                 |MN(IP2,IP1)    |
.flow(IP1,...)  .               =======>               |flow(IP1,...)  |
+...............+                                      +---------------+
```

Figure 7. IP prefix/address anchor switching to the new network with with LM and FM-CP in a centralized control plane whereas the FM-DP’s are distributed.

The call flow in Figure 8 shows that MN is allocated HNP1 when it attaches to the p-AR. A flow running in MN may or may not need IP mobility. If it does, it may continue to use the previous IP prefix. If it does not, it may use a new IP prefix allocated from the new network.

```
 MN                    p-AR          n-AR              DHCP Servers   CN
  |MN attaches to p-AR: |             |                     |          |
  |acquire MN-ID and profile          |                     |          |
  |--RS---------------->|             |                     |          |
  |<----------RA(HNP1)--|             |                     |          |
  |                     |             |             Allocate MN-HNP1   |
IP addr config          |             |                     |          |
  |                     |             |                     |          |
  |<-Flow(IP1,IPcn,...)-+--------------------------------------------->|
  |                     |             |                     |          |
  |MN detach from p-AR  |             |                     |          |
  |MN attach to n-AR    |             |                     |          |
  |                     |             |                     |          |
  |--RS------------------------------>|                     |          |
  |                     |             |                     |          |
  |                     |------DHCPv6 release-------------->|          |
  |                     |             |                     |          |
  |                     |             |--DHCPv6 PD request->|          |
  |                     |             |<-DHCPv6 PD reply--->|          |
  |                     |             |                     |          |
  |                 forwarding table updates                |          |
  |                     |             |                     |          |
  |<--------------RA(HNP2,HNP1)-------|                     |          |
  |                     |             |             Allocate MN-HNP2   |
IP addr config          |             |                     |          |
  |                     |             |                     |          |
  |<-Flow(IP1,IPcn,...)---------------+------------------------------->|
  |                     |             |                     |          |
  |  Flow(IP1,IPcn,...) terminates    |                     |          |
  |                     |             |                     |          |
  |                     | DHCPv6-PD timeout                 |          |
  |                     |             |                     |          |
  |                 forwarding table updates                |          |
  |                     |             |                     |          |
  |                     |             |                     |          |
  |<-new Flow(IP2,IPcn,...)-----------+------------------------------->|
  |                     |             |                     |          |
```

Figure 8. DMM solution. MN with flow using IP1 in Net1 continues to run the flow using IP1 as it moves to Net2.

As the MN moves from p-AR to n-AR, the p-AR as a DHCP client may send a DHCP release message to release the HNP1. It is now necessary for n-AR to learn the IP prefix of the MN from the previous network so that it will be possible for Net2 to allocate both the previous network prefix and the new network prefix. The network may learn the previous prefix in different methods. For example, the MN may provide its previous network prefix information by including it to the RS message [I-D.jhlee-dmm-dnpp].

Knowing that MN is using HNP1, the n-AR sends to a DHCP server a DHCPv6-PD request to move the HNP1 to n-AR. The server sends to n-AR a DHCPv6-PD reply to move the HNP1. Then BGP route updates will take place here.

In addition, the MN also needs a new HNP in the new network. The n-AR may now send RA to n-AR, with prefix information that includes HNP1 and HNP2. The MN may then continue to use IP1. In addition, the MN is allocated the prefix HNP2 with which it may configure its IP addresses. Now for flows using IP1, packets destined to IP1 will be forwarded to the MN via n-AR.

As such flows have terminated and DHCP-PD has timed out, HNP1 goes back to Net1. MN will then be left with HNP2 only, which it will use when it now starts a new flow.

The anchor behavior to properly forward the packets for a flow as described in the FM behaviors and information elements (FM:1) in Section 3.2.2 is realized by changing the anchor with DHCPv6-PD and undoing such changes later when its timer expires and the application has already closed. With the anchors being separated in control and data planes with LMs and FM-CP centralized in the same control plane, messaging between anchors and the discovery of anchors become internal to the control plane. However, the centralized FM-CP needs to communicate with the distributed FM-DP as described as described in the FM behaviors and information elements (FM:3). Such may be realized by the appropriate messages in [I-D.ietf-dmm-fpc-cpdp]. Again, if there are in-flight packets toward the old anchor while the MN is moving to the new anchor, it may be necessary to buffer these packets and then forward to the new anchor after the old anchor knows that the new anchor is ready. The corresponding FM behaviors and information elements (FM:4) are however realized by the internal behavior in the control plane together with signaling between the control plane and distributed data plane.

4.2.2. Hierarchical network

The configuration for a hierarchical network is shown in Figures 1(c) and 1(d) in Section 3.1. With centralized control and with a centralized anchor, LM, CPA, CPN are co-located at the centralized control, and there is an AR with the DPA function supporting multiple forwarding switches (FW’s) each with a DPN function. A mobility event in this configuration involving change of FW but not of AR is shown in Figure 9.

Here the IP prefix allocated to the MN is anchored at the access router (AR) supporting the old FW to which the MN was originally attached as well as the new FW to which the MN has moved.

The realization of LM may bet the binding between the IP prefix/ address of the flow used by the MN and the IP address of the DPN to which MN has moved. The implementation of FM to enable change of FW without changing AR may be accomplished using tunneling between the AR and the FW as described in [I-D.korhonen-dmm-local-prefix] and in [I-D.templin-aerolink] or using some other L2 mobility mechanism.

```
Net1                                                   Net2
+----------------------------------------------------------------------+
|                            CPA,CPN:                                  |
|                            LM:IP1<-->IPn2                            |
|                            FM-CP                                     |
+----------------------------------------------------------------------+


                           +---------------+
                           |AR1            |
                           +---------------+
                           |DPA(IPa1):     |
                           |anchors IP1    |
                           |FM:DHCPv6-PD   |
                           +---------------+


+---------------+                                      +---------------+
|FW1            |                                      |FW2            |
+---------------+                 move                 +---------------+
|DPN(IPn1):     |               =======>               |DPN(IPn2):     |
+---------------+                                      +---------------+


+...............+                                      +---------------+
.MN(IP1)        .                 move                 |MN(IP2)        |
.flow(IP1,...)  .               =======>               |flow(IP1,...)  |
+...............+                                      +---------------+
```

Figure 9. Mobility without involving change of IP anchoring in a network with hierarchy in which the IP prefix allocated to the MN is anchored at an Edge Router supporting multiple access routers to which the MN may connect.

Here, the LM behaviors and information elements described in Section 3.2.1 provides information of which IP prefix from its FW needs to be used by a flow using which new FW. The anchor behaviors to properly forward the packets of a flow described in the FM behaviors and information elements (FM:1) may be realized with PMIPv6 protocol ([I-D.korhonen-dmm-local-prefix]) or with AERO protocol ([I-D.templin-aerolink]) to tunnel between the AR and the FW.

4.2.3. Hierarchical network with anchoring change

The configuration for a hierarchical network is still shown in Figures 1(c) and 1(d) in Section 3.1. Again, with centralized control and with a centralized anchor, LM, CPA, CPN are co-located at the centralized control, and there is an AR with the DPA function supporting multiple forwarding switches (FW’s) each with a DPN function. However, the mobility event involving change of FW may also involve a change of AR. Such configuration is shown in Figure 10.

This deployment case involves both a change of anchor from AR1 to AR2 and a network hierarchy AR-FW. It can be realized by a combination of changing the IP prefix/address anchoring from AR1 to AR2 with the mechanism as described in Section 4.2.1 and then forwarding the packets with network hierarchy AR-FW as described in Section 4.2.2.

To change AR, AR1 acting as a DHCP-PD client may exchange message with the DHCP server to release the prefix IP1. Meanwhile, AR2 acting as a DHCP-PD client may exchange message with the DHCP server to delegate the prefix IP1 to AR2.

```
Net1                                                   Net2
+----------------------------------------------------------------------+
|                            CPA,CPN:                                  |
|                            LM:IP1<-->IPa2,IPn2                       |
|                            FM-CP                                     |
+----------------------------------------------------------------------+

                            +---------------+
                            |Aggregate Point|
                            |---------------|
                            |FM,    LM      |
                            +---------------+


+---------------+                                      +---------------+
|AR1            |                                      |AR2            |
+---------------+                                      +---------------+
|DPA(IPa1):     |                                      |DPA(IPa2):     |
|anchors IP1    |                 move                 |anchors IP2,IP1|
|FM:DHCPv6-PD   |               =======>               |FM:DHCPv6-PD   |
+---------------+                                      +---------------+


+---------------+                                      +---------------+
|FW1            |                                      |FW2            |
+---------------+                 move                 +---------------+
|DPN(IPn1):     |               =======>               |DPN(IPn2):     |
+---------------+                                      +---------------+


+...............+                                      +---------------+
.MN(IP1)        .                 move                 |MN(IP2,IP1)    |
.flow(IP1,...)  .               =======>               |flow(IP1,...)  |
+...............+                                      +---------------+
```

Figure 10. Mobility involving change of IP anchoring in a network with hierarchy in which the IP prefix allocated to the MN is anchored at an Edge Router supporting multiple access routers to which the MN may connect.

5. Security Considerations

TBD

6. IANA Considerations

This document presents no IANA considerations.

7. Contributors

This document has benefited from other work on mobility solutions using BGP update, on mobility support in SDN network, on providing mobility support only when needed, and on mobility support in enterprise network. These work have been referenced. While some of these authors have taken the work to jointly write this document, others have contributed at least indirectly by writing these drafts. The latter include Philippe Bertin, Dapeng Liu, Satoru Matushima, Peter McCann, Pierrick Seite, Jouni Korhonen, and Sri Gundavelli.

Valuable comments have also been received from John Kaippallimil, ChunShan Xiong, and Dapeng Liu.

8. References

8.1. Normative References

[I-D.ietf-dmm-fpc-cpdp]
Liebsch, M., Matsushima, S., Gundavelli, S., Moses, D., and L. Bertz, "Protocol for Forwarding Policy Configuration (FPC) in DMM", draft-ietf-dmm-fpc-cpdp-03 (work in progress), March 2016.

[I-D.ietf-dmm-ondemand-mobility]
Yegin, A., Moses, D., Kweon, K., Lee, J., and J. Park, "On Demand Mobility Management", draft-ietf-dmm-ondemand-mobility-07 (work in progress), July 2016.

[I-D.jhlee-dmm-dnpp]
Lee, J. and Z. Yan, "Deprecated Network Prefix Provision", draft-jhlee-dmm-dnpp-01 (work in progress), April 2016.

[I-D.korhonen-dmm-local-prefix]
Korhonen, J., Savolainen, T., and S. Gundavelli, "Local Prefix Lifetime Management for Proxy Mobile IPv6", draft-korhonen-dmm-local-prefix-01 (work in progress), July 2013.

[I-D.liu-dmm-deployment-scenario]
Liu, V., Liu, D., Chan, A., Lingli, D., and X. Wei, "Distributed mobility management deployment scenario and architecture", draft-liu-dmm-deployment-scenario-05 (work in progress), October 2015.

[I-D.matsushima-stateless-uplane-vepc]
Matsushima, S. and R. Wakikawa, "Stateless user-plane architecture for virtualized EPC (vEPC)", draft-matsushima-stateless-uplane-vepc-06 (work in progress), March 2016.

[I-D.mccann-dmm-flatarch]
McCann, P., "Authentication and Mobility Management in a Flat Architecture", draft-mccann-dmm-flatarch-00 (work in progress), March 2012.

[I-D.mccann-dmm-prefixcost]
McCann, P. and J. Kaippallimalil, "Communicating Prefix Cost to Mobile Nodes", draft-mccann-dmm-prefixcost-03 (work in progress), April 2016.

[I-D.seite-dmm-dma]
Seite, P., Bertin, P., and J. Lee, "Distributed Mobility Anchoring", draft-seite-dmm-dma-07 (work in progress), February 2014.

[I-D.sijeon-dmm-deployment-models]
Jeon, S. and Y. Kim, "Deployment Models for Distributed Mobility Management", draft-sijeon-dmm-deployment-models-03 (work in progress), July 2016.

[I-D.templin-aerolink]
Templin, F., "Asymmetric Extended Route Optimization (AERO)", draft-templin-aerolink-67 (work in progress), June 2016.

[I-D.wt-dmm-deployment-models]
Gundavelli, S., "DMM Deployment Models and Architectural Considerations", draft-wt-dmm-deployment-models-00 (work in progress), April 2016.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC5213] Gundavelli, S., Ed., Leung, K., Devarapalli, V., Chowdhury, K., and B. Patil, "Proxy Mobile IPv6", RFC 5213, DOI 10.17487/RFC5213, August 2008, <http://www.rfc-editor.org/info/rfc5213>.

[RFC6275] Perkins, C., Ed., Johnson, D., and J. Arkko, "Mobility Support in IPv6", RFC 6275, DOI 10.17487/RFC6275, July 2011, <http://www.rfc-editor.org/info/rfc6275>.

[RFC7333] Chan, H., Ed., Liu, D., Seite, P., Yokota, H., and J. Korhonen, "Requirements for Distributed Mobility Management", RFC 7333, DOI 10.17487/RFC7333, August 2014, <http://www.rfc-editor.org/info/rfc7333>.

[RFC7429] Liu, D., Ed., Zuniga, JC., Ed., Seite, P., Chan, H., and CJ. Bernardos, "Distributed Mobility Management: Current Practices and Gap Analysis", RFC 7429, DOI 10.17487/RFC7429, January 2015, <http://www.rfc-editor.org/info/rfc7429>.

8.2. Informative References

[Paper-Distributed.Mobility]
Lee, J., Bonnin, J., Seite, P., and H. Chan, "Distributed IP Mobility Management from the Perspective of the IETF: Motivations, Requirements, Approaches, Comparison, and Challenges", IEEE Wireless Communications, October 2013.

[Paper-Distributed.Mobility.PMIP]
Chan, H., "Proxy Mobile IP with Distributed Mobility Anchors", Proceedings of GlobeCom Workshop on Seamless Wireless Mobility, December 2010.

[Paper-Distributed.Mobility.Review]
Chan, H., Yokota, H., Xie, J., Seite, P., and D. Liu, "Distributed and Dynamic Mobility Management in Mobile Internet: Current Approaches and Issues", February 2011.

Authors’ Addresses

H Anthony Chan
Huawei Technologies
5340 Legacy Dr. Building 3
Plano, TX 75024
USA

Email: h.a.chan@ieee.org

Xinpeng Wei
Huawei Technologies
Xin-Xi Rd. No. 3, Haidian District
Beijing, 100095
P. R. China

Email: weixinpeng@huawei.com

Jong-Hyouk Lee
Sangmyung University
708 Hannuri Building
Cheonan 330-720
Korea

Email: jonghyouk@smu.ac.kr

Seil Jeon
Instituto de Telecomunicacoes
Campus Universitario de Santiago
Aveiro 3810-193
Portugal

Email: seiljeon@av.it.pt

Fred L. Templin
Boeing Research and Technology
P.O. Box 3707
Seattle, WA 98124
USA

Email: fltemplin@acm.org

Internet Engineering Task Force
Internet-Draft
Intended status: Informational
Expires: April 21, 2016

Q. Fu, Ed.
H. Deng
China Mobile

October 19, 2015

Motivations, usecases and Models of VCPE
draft-fu-dmm-vcpe-models-01

Abstract

This document introduces the concept of Virtual Customer Premises Equipment (VCPE). Such concept was first proposed in Broadband Forum (BBF) as Network Enhanced Residential Gateway (NERG). The concept is further expanded as not only referring to virtual CPE of residential network, but all the virtual network and service functions shifted from the customer side to the operator side. Deployment of VCPE in some typical DMM (Distributed Mobility Management) scenarios brings specific requirements and even protocol extension in DMM. In this document, we will first explain the motivation and advantages of VCPE. A usecases of VCPE in the community Wi-Fi deployment is further discussed so as to explain the deployment of VCPE in a DMM scenario. Three models of field deployment of VCPE are discussed afterwards to indicate the possible CP/DP decomposition requirement and protocol extension.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

This document introduces the concept of VCPE. The concept of VCPE is to shift most of the networking and service functionalities from the customer side to the network side. In this way, the customer side's equipment, that is the pCPE (Physical Customer Premises Equiptment), can be simplified. The VCPE refers to one or a set of equipments at the network side to execute the networking and service functionalities used to be executed at the CPE. In such architecture, the CPE can be a simple L2 switch, which is only responsible for forwarding packets to a certain next hop. The concept of VCPE was first introduced in BBF as NERG (WT-317), which mainly focuses on shifting some of the functionalities of a residential gateway to the operator's network, for enabling network based features. The aim is to facilitate the deployment, maintenance and evolution of both existing and new capabilities without adding complexity to the RG and/or the home network.

Figure 1 shows the architecture of the pCPE and the VCPE.

```
                    ...............         ..............
                    :Customer side:         :Network side:
                    :   +-----+   :         :  +-----+   :
                    :   |pCPE |================|VCPE |   :
                    :   +-----+   :         :  +-----+   :
                    ..............:         .............:
```

Figure 1: VCPE Architecture

In this document, we would like to further propose such concept in the following aspects:

(1) Motivation and advantages of VCPE.

(2) Usecases of VCPE. A usecase of VCPE in the community Wi-Fi is explained in detail.

(3) Models of VCPE deployment. We propose three models for the field deployment of VCPE.

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].

3. Motivation and Advantage of VCPE

The motivation and advantage of introduting VCPE can be concluded as follows:

(1) It will greatly speed up the service launching period. Since most of the complicated functions are located at the VCPE in the network side, operators have more power over services. Benefitting from the recent NFV (Network Function Virtualization) and cloud technologies, VCPE can be accomplished using SFC in the virtual network, where different services can act as different VNFs (Virtual Network Functions). Operators only need to add new VNFs on the VCPE side to launch new services to the customers. In this way, Operators can provide a variety of services through the network.

(2) It will reduce the cost of the pCPE. By shifting most of the complicated functions from the customer’s side to the operator’s side, the cost of the pCPE can be reduced significantly. Such reduncton can be remarkable in the enterprise network, since network functions, such as Firewall and NAT(Network Address Translator) at the customer side can be expensive. In the meantime, the cost of

upgrading tens of thousands of pCPE when launching new services can be saved, since only software upgrade at the VCPE side is required.

(3) It will simplify the maintainance of the pCPE. Since most of the complicated functionalities are shifted to the network side, the maintainance of the pCPE can be greatly simplified. On-line maintainance is possible in lots of cases since the pCPE is only a L2 devices and can be considered transparent to the operators.

(4) It will provide user-define-network experience. By introducing SFC concept into the VCPE, users can define his own service order and sequence. Therefore, customers can enjoy the self-defined services over the public network.

4. Use case of VCPE

The concept of VCPE can be used in multiple scenarios. In this section, we will propose a usecase of VCPE when deploying community Wi-Fi.

The community Wi-Fi is a new service that operators provide to leverage unused capacity on existing residential Wi-Fi infrastructure to offer Wi-Fi network access to visitors and passers by near the neighbourhood. An operator can also use this excess capacity to offer services to retail and roaming-parter operators’ subscribers. The residential subscribers accessing the network from inside their homes have prioritized access to the Wi-Fi resources. The residential Wi-Fi infrastructure is configured in a manner that allows for a secure and independent access channel to retain service quality, safety, and privacy for both residential and visitor customers. Roaming users are only allowed to use the Wi-Fi network capacity that is not currently used by the subscriber at home.

Basically, the wireless Access Point (AP) in the home will provide two networks: a private one for the home owner/subscriber, and a community network for on-the-go subscribers passing through the neighborhood. Home users can have all of their Wi-Fi devices (smartphone, tablet, etc.) automatically connect to the private network. In the meantime users travel outside can connect to the community network, and can roaming through different APs supporting community Wi-Fi as he/she is moving. The community Wi-Fi service is a typical usecase of DMM.

Deploying community Wi-Fi on the pCPE means upgrading tens of thousands of existing pCPE devices at the customer side, which is not cost-effective and may bring extra complexity for maintainance. Therefore VCPE becomes an optimized solution for such deployment. In such deployment, the private users access to the pCPE (which is the

AP at home) as usual. The public users are roaming through different pCPEs. The traffic all goes though the tunnel from the pCPE to the VCPE. The deployment of VCPE in the community Wi-Fi sencario brings specific requirement and protocol extensions to DMM. The deployment model of VCPE and its possible influence to DMM is further discussed in the following section.

5. Models of VCPE Deployment

There are multiple models when deploying VCPE in use cases as are discussed in the previous section. In this document, we conclude the deployment of VCPE into three models. In the first model, a logical instance of VCPE is deployed in the cloud for each pCPE instance. That is, the pCPE and VCPE is deployed in an 1:1 manner. All traffic from pCPE goes through the vCPE.

```
            +------+     +------+
            | pCPE +-----| VCPE |-----+
            +------+     +------+     |
                                      |         +------+     ________
            +------+     +------+   +-+---+     |SFC in|    /        \
            | pCPE +-----+ VCPE +---+ BNG +-----+WAN DC+---+ Internet |
            +------+     +------+   +-+---+     +------+    \________/
                                      |
            +------+     +------+     |
            | pCPE +-----| VCPE |-----+
            +------+     +------+
```

Figure 2: VCPE deployment model NO.1: Logical Instance of VCPE

In the second model, vCPE is modeled service function chains in Gi-LAN. BNG knows how to classify the traffic from a given CPE with the help of the control plane, and run it through the service chain. In such model, the CP/DP interface should be used between the control plane (which might be the controller) and the pCPE.

```
             +----------+
             +Controller|
             +--+-------+
                +CP/DP interface        .......................
                +                       :VCPE                 :
                +  +---+--+             : +-----------------\ :
                ++++ pCPE +---------+   :|SFP1:DPI->FW->NAT + :
                +  +------+         |   : +-----------------/ :
                +                   |   :                     :
                +  +------+     +--+--+ : +------------\      :
                ++++ pCPE +------+ BNG +---+SFP2:FW->NAT +    :
                +  +------+     +--+--+ : +------------/      :
                +                   |   :                     :
                + +------+         |   : +--------\          :
                ++++ pCPE +---------+   : |SFP3:NAT +         :
                   +------+             : +--------/          :
                                        .......................
```

Figure 3: VCPE deployment model NO.2: VCPE as SFC

The third model is almost the same with the second one, except that the BNG is also CP/DP decomposed. In this model, The control plane is composed of the controller of the pCPE and the control plane of the BNG. The CP/DP interface is used between the controller and the pCPE, and between the control plane and the data plane of the BNG. Both of model No.2 and No.3 may have specific requirement and protoco extensions for the CP/DP interface due to the usecase of VCPE.

```
            .................................
            : Control Plane                 :
            :       +----------+  +------+  :
            : +++++++Controller|  |BNG CP|  :
            : +     +----------+  +--+---+  :
            ..+......................+......:
              +   CP/DP interface    +    .......................
              +                      +    :VCPE                 :
              +  +---+--+            +    : +-----------------\ :
              ++++ pCPE +--------+   +    :|SFP1:DPI->FW->NAT + :
              +  +------+        |   +    : +-----------------/ :
              +                  |   +    :                     :
              +  +------+     +-+----+-+  : +------------\      :
              ++++ pCPE +-----+ BNG DP +----+SFP2:FW->NAT +     :
              +  +------+     +-+------+  : +------------/      :
              +                 |         :                     :
              +  +------+       |         : +--------\          :
              ++++ pCPE +--------+        : |SFP3:NAT +         :
                 +------+                 : +--------/          :
                                          .......................
```

Figure 4: VCPE deployment model NO.3: VCPE as SFC, with CP/DP decomposition of BNG

SDN (Software Define Network) controllers can also be introduced in the third model. In which case, all of the pCPEs and the BNG data plane (BNG DP) can be controlled by the SDN-controller. When the customer selects a set of services, the SDN-controller will inform the pCPE and the BNG DP to direct the traffic flow to a certain SFC.

```
            .................................
            : Control Plane                 :
            :       +--------------+        :
            : +++++++SDN Controller|++      :
            : +     +--------------+ +      :
            ..+......................+......:
              +   CP/DP interface    +    .......................
              +                      +    :VCPE                 :
              +  +---+--+            +    : +-----------------\ :
              ++++ pCPE +-------+    +    :|SFP1:DPI->FW->NAT + :
              +  +------+       |    +    : +-----------------/ :
              +                 |    +    :                     :
              +  +------+     +-+----+-+  : +------------\      :
              ++++ pCPE +-----+ BNG DP +----+SFP2:FW->NAT +     :
              +  +------+     +-+------+  : +------------/      :
              +                 |         :                     :
              +  +------+       |         : +--------\          :
              ++++ pCPE +-------+         : |SFP3:NAT +         :
                 +------+                 : +--------/          :
                                          .......................
```

Figure 5: VCPE deployment model NO.3: SFC realization of VCPE, with SDN controller as control plane

6. VCPE Deployment for Community Wi-Fi

In this section, we will discuss about the VCPE deployment for Community Wi-Fi in detail. In the following deployment, we assume the VCPE is deployed following the third model we discussed in section 5. That is, the VCPE is a bounch of SFCs at the operator side behind the BNG. The pCPEs and BNG-DP are all controlled by a mutual control plane. The FPC protocol is used between the control plane and the pCPEs, and that and the BNG-DP.

As we discussed in section 4, Community Wi-Fi can be deployed with the help of deploying VCPE. In order to provide the Community Wi-Fi service, the pCPE should provide two SSIDs, one for the pubic Wi-Fi users, and the other for the private Wi-Fi users. Packets from different SSID are marked with different VLAN ID. The VCPE should know of the corresponding relation between the SSID and the VLAN ID, so as to provide distinguished services to the publice users and the private users. For instance, the private users should experience a better QoS than the publice ones. In the meantime, the private users and the public users may choose different SFC in the VCPE. All of these different services are classified based on the VLAN ID.

Such deployment requirs the FPC client to support the following task:

1) The FPC client should be able to set specific VLAN to each SSID.

2) The FPC client should be able to set the QoS for specific VLAN ID.

3) The FPC client should be able to inform the agent the specific SFC for each VLAN ID.

4) The FPC client should be capable of instruct the agent to handle the MN hand-over of the public Wi-Fi users.

In the meantime, such deployment requirs the FPC agent to support the following task:

1) The FPC agent should be able to set specific VLAN to each SSID following the command from the client.

2) The FPC agent should be able to set the QoS for specific VLAN ID following the command from the client.

3) The FPC agent should be able to direct the traffic for specific VLAN ID to a certain SFC following the command of the client.

4) The FPC agent should be able to handle the MN hand-over of the public Wi-Fi users.

7. Conclusion

In this document, the concept of VCPE is illustrated in detail. The basic concept of VCPE is to shift the complicated functions from the pCPE at the customer side to the VCPE at the service provider side. The motivation of such shifting can be concluded as providing quick launched customer defined services, reducing the Capex and Opex of the pCPE, and simlify the maintainance of both pCPE and VCPE. A use cases of community Wi-Fi is proposed for VCPE, which is a typical scenario for DMM. Three models are then discussed for the field deployment of VCPE. And CP/DP interface is suggested to be utilized in the deployment models.

8. Informative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

Authors' Addresses

Qiao Fu (editor)
China Mobile
Xuanwumenxi Ave. No.32
Beijing
China

Email: fuqiao1@outlook.com

Hui Deng
China Mobile
Xuanwumenxi Ave. No.32
Beijing
China

Email: denghui@chinamobile.com

IPSECME WG D. Patki
Internet-Draft S. Gundavelli
Intended status: Standards Track Cisco
Expires: April 14, 2016 J. Lee
Sangmyung University
Q. Fu
China Mobile
L. Bertz
Sprint
October 12, 2015

LMA Controlled MAG Session Parameters
draft-gundavelli-dmm-lma-controlled-mag-params-00.txt

Abstract

This specification defines a new extension, LMA-Controlled-MAG-Session-Params to Proxy Mobile IPv6. This option can be used by the LMA in PMIPv6 signaling for notifying the MAG to conform to various parameters contained in this extension.

Status of this Memo

Copyright Notice

Table of Contents

1. Introduction

A large PMIPv6 deployment, such as residential deployment, can have tens of thousands of MAGs spread across geographical locations. While it can be operationally challenging to manage such large number of MAGs, it can also be very difficult to ensure configuration consistency across all the MAGs if they are not centrally managed. Configuring aggresive values of parameters such as re-registration timeout and heartbeat interval can potentially create considerable signaling load on the LMA. This document provides a new option to enable the LMA to control various parameters on the MAG such as the re-registration frequency [RFC5213] and heartbeat frequency [RFC5847]. With this option, the configuration of these tunable parameters done centrally on the LMA enables Service Providers to have better control on the behavior of the MAGs with deterministic singaling load on the LMA.

2. Conventions and Terminology

2.1. 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 [RFC2119].

2.2. Terminology

All the terms used in this document are to be interpreted as defined in [RFC5213], [RFC5847] and [RFC7563].

3. Protocol Extension

The LMA Controlled MAG Parameters (LCMP) option is a mobility header option used to exchange information related to the parameters that a local mobility anchor enforces on a mobile access gateway. The option can be included in Proxy Binding Acknowledgement (PBA) message only, and there MUST NOT be more than a single instance of this mobility option in a mobility message. This mobility option MUST contain one or more LMA Controlled MAG Parameters sub-options. The suboptions are defined in Section 3.1. The alignment requirement for this option is 4n [RFC2460].

```
    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
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |      Type     |   Length      |           Reserved            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                  ... LCMP Sub-option(s) ...                  ~
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 1: LMA Controlled MAG Parameters Option

Type

MUST be set to the value of IANA-1, indicating that it is a LMA-Controlled-MAG-Parameters option.

Length

8-bit unsigned integer indicating the length in octets of the option, excluding the Type and Length fields.

Reserved

MUST be set to zero when sending and ignored when received.

3.1. Format of the LCMP Sub-Options

The LMA Controlled MAG Parameters sub-options are used for carrying information elements related to various parameters that need to be configured on the MAG. These sub-options can be included in the LMA Controlled MAG Parameters option defined in Section 3. The format of this sub-option is as follows. The alignment requirement for the sub-option is 4n. The sub-options are optional and can be present in any order.

```
    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
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |   LCMP Type   | LCMP Length   |       Sub-Option Data         ~
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2: LMA Controlled MAG Parameters Sub-Option

Type

8-bit unsigned integer indicating the type of the LMA Controlled MAG Parameters sub-option. This specification defines the following types:

0 - Reserved

1 - Binding Refresh Control Sub-Option

2 - Heartbeat Control Sub-Option

Length

8-bit unsigned integer indicating the number of octets needed to encode the Option Data, excluding the LCMP Type and LCMP Length fields of the sub-option.

3.1.1. Binding Re-registration Control Sub-Option

The Binding Re-registration Control Sub-Option is a mobility sub-option carried in the LMA Controlled MAG Parameters mobility option defined in Section 3.1. This sub-option carries re-registration related timer values. There MUST be no more than a single instance of this sub-option in LMA Controlled MAG Parameters option. The format of this sub-option is defined below.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| LCMP Type = 1 |  LCMP Length  |  Re-registration-Start-Time   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Initial-Retransmission-Time  | Maximum-Retransmission-Time   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 3: Binding Re-registration Control Sub-Option

Re-registration-Start-Time

16-bit unsigned integer indicating the number of time units before the expiry of the PMIPv6 binding lifetime when the registration refresh process needs to be activated. One time unit is 4 seconds.

Initial-Retransmission-Time

16-bit unsigned integer indicating minimum delay in seconds before the first PBU retransmission of the exponential back-off process.

Maximum-Retransmission-Time

16-bit unsigned integer indicating maximum delay in seconds before the last PBU retransmission message of the exponential back-off process.

3.1.2. Heartbeat Control Sub-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 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    | LCMP Type = 2 |  LCMP Length  |        HB-Interval            |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |    HB-Retransmission-Delay    |    HB-Max-Retransmissions     |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 4: Heartbeat Control Sub-Option

HB-Interval

16-bit unsigned integer indicating heartbeat interval, i.e. time delay in seconds after a successful heartbeat exchange (request followed by response) when the next heartbeat exchange can be triggered.

HB-Retransmission-Delay

16-bit unsigned integer indicating minimum time delay in seconds before a heartbeat message is retransmitted.

HB-Max-Retransmissions

16-bit unsigned integer indicating maximum number of heartbeat retransmissions.

4. Protocol Configuration Variables

4.1. Local Mobility Anchor - Configuration Variables

The local mobility anchor MUST allow the following variables to be configured by the system management. The configured values for these protocol variables MUST survive server reboots and service restarts.

EnableLCMPSubOptReregControl

This flag indicates the operational state of the Binding Re-registration Control sub-option support. The default value for this flag is set to (0), indicating that support for the Binding Re-registration Control sub-option is disabled.

When this flag on the mobile access gateway is set to a value of (1), the local mobility anchor SHOULD include this sub-option in the Proxy Binding Acknowledge messages that it sends to the mobile access gateway; otherwise, it SHOULD NOT include the sub-option. There can be situations where the local mobility anchor is unable to obtain the Binding Re-registration Control information and may not be able to construct this sub-option.

EnableLCMPSubOptHeartbeatControl

This flag indicates the operational state of the Heartbeat Control sub-option support. The default value for this flag is set to (0), indicating that support for the Heartbeat Control sub-option is disabled.

When this flag on the mobile access gateway is set to a value of (1), the local mobility anchor SHOULD include this sub-option in the Proxy Binding Acknowledge messages that it sends to the mobile access gateway; otherwise, it SHOULD NOT include the sub-option. There can be situations where the local mobility anchor is unable to obtain the Heartbeat Control information and may not be able to construct this sub-option.

The following variables MAY be defined at various granularity such as per binding, per peering MAG, per cluster of MAGs or any other custom grouping. Regardless of the granularity of this configuration, the local mobility anchor should be able to determine the value of these variables on an individual binding basis by way of configuration hierarchy.

LCMPReregistrationStartTime

This variable is used to set the minimum time interval in number of seconds before the expiry of the PMIPv6 binding lifetime when the registration refresh process SHOULD be activated.

LCMPInitialRetransmissionTime

This variable is used to set the minimum delay in seconds before the first PBU retransmission of the exponential back-off process. This variable is same as INITIAL_BINDACK_TIMEOUT mentioned in Section 6.9.4 of [RFC5213].

LCMPMaximumRetransmissionTime

This variable is used to set the maximum delay in seconds before the last PBU retransmission message of the exponential back-off process. This variable is same as MAX_BINDACK_TIMEOUT mentioned in Section 6.9.4 of [RFC5213].

LCMPHeartbeatInterval

This variable is used to set the time delay in seconds after a successful heartbeat exchange (request followed by response) when the next heartbeat exchange can be triggered. The default value is 60 seconds. It SHOULD NOT be set to less than 30 seconds or more than 3600 seconds. The value of this variable MAY be derived from the variable HEARTBEAT_INTERVAL defined in Section 5 of [RFC5847] if defined on the local mobility anchor.

LCMPHeartbeatRetransmissionDelay

This variable is used to set the minimum time delay in seconds before a heartbeat message is retransmitted.. The value of this variable SHOULD be less than LCMP_HEARTBEAT_INTERVAL. The default value is 5 seconds.

LCMPHeartbeatMaxRetransmissions

This variable is used to set the maximum number of heartbeat retransmissions. The default value for this variable is 3. The value of this variable MAY be derived from the variable MISSING_HEARTBEATS_ALLOWED defined in Section 5 of [RFC5847] if defined on the local mobility anchor.

5. Protocol Considerations

The following considerations apply to the local mobility anchor and the mobile access gateway.

The conceptual Binding Cache Entry data structure maintained by the local mobility anchor, described in Section 5.1 of [RFC5213] and the conceptual Binding Update List entry data structure maintained by the mobile access gateway, described in Section 6.1 of [RFC5213], MUST be extended to store the LMA Controlled MAG Parameters option related information elements associated with the current session. Specifically the following parameters MUST be defined:

- LCMPReregistrationStartTime

- LCMPInitialRetransmissionTime

- LCMPMaximumRetransmissionTime

- LCMPHeartbeatInterval

- LCMPHeartbeatRetransmissionDelay

- LCMPHeartbeatMaxRetransmissions

5.1. Local Mobility Anchor Considerations

- On receiving a Proxy Binding Update message [RFC5213] from a mobile access gateway, the local mobility anchor should check if EnableLCMPSubOptReregControl is set to (1). If yes, and if all of LCMPReregistrationStartTime, LCMPInitialRetransmissionTime and LCMPMaximumRetransmissionTime are set to NON_ZERO values, then in SHOULD include Binding Re-registration Control Sub-Option in the LMA Controlled MAG Parameters mobility option which is in turn included in the Proxy Binding Acknowledge message.

- If EnableLCMPSubOptReregControl is set to (1) and if any of LCMPReregistrationStartTime, LCMPInitialRetransmissionTime and LCMPMaximumRetransmissionTime is set to ZERO value, then the local mobility anchor should report a configuration error.

- The local mobility anchor should also check if EnableLCMPSubOptHeartbeatControl is set to (1). If yes, and if all of LCMPHeartbeatInterval, LCMPHeartbeatRetransmissionDelay and LCMPHeartbeatMaxRetransmissions are set to NON_ZERO values, then in SHOULD include Heartbeat Control Sub-Option in the LMA Controlled MAG Parameters mobility option which is in turn included in the Proxy Binding Acknowledge message.

- If EnableLCMPSubOptHeartbeatControl is set to (1) and if any of LCMPHeartbeatInterval, LCMPHeartbeatRetransmissionDelay and LCMPHeartbeatMaxRetransmissions is set to ZERO value, then the local mobility anchor should report a configuration error.

5.2. Mobile Access Gateway Considerations

- On Receiving Proxy Binding Acknowledge message [RFC5213] from the local mobility anchor with LMA Controlled MAG Parameters mobility option, the mobile access gateway MUST overwrite the binding re-registration related timer parameters with the parameters received in Binding Re-registration Control Sub-Option, if present in the LMA Controlled MAG Parameters mobility option. Similarly, the mobile access gateway MUST overwrite the heartbeat related timer parameters with the parameters reveived in Heartbeat Control Sub-Option, if present in the LMA Controlled MAG Parameters mobility option.

- If any of the parameters in the Binding Re-registration Control Sub-Option is ZERO, then the sub-option MUST be ignored and an error message SHOULD be logged.

- If any of the parameters in the Heartbeat Control Sub-Option except HB-Retransmission-Delay is ZERO, then the sub-option MUST be ignored and error message SHOULD be logged.

6. IANA Considerations

This document requires the following IANA actions.

- Action 1: This specification defines a new mobility header option, the LMA Controlled MAG Parameters. This mobility option is described in Section 3. The type value (IANA-1) for this option needs to be assigned from the same numbering space as allocated for the other mobility options, as defined in [RFC6275].

- Action 2: This specification defines a new mobility sub-option format, the LMA Controlled MAG Parameters sub-option. The format of this mobility sub-option is described in Section 3.1. This sub-option can be carried in the LMA Controlled MAG Parameters option. The type value for this sub-option needs to be managed by IANA, under the registry "LMA Controlled MAG Parameters Sub-Option Type Values". This specification reserves the following type values. Approval of new LMA Controlled MAG Parameters sub-option type values are to be made through IANA Expert Review.

0	Reserved
1	Binding Re-registration Control Sub-Option
2	Heartbeat Control Sub-Option

```
      +---+------------------------------------------------------+
```

7. Security Considerations

The LMA Controlled MAG Parameters option defined in this specification is for use in Proxy Binding Acknowledgement message. This option is carried like any other mobility header option as specified in [RFC6275] and does not require any special security considerations.

8. References

8.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC5213] Gundavelli, S., Ed., Leung, K., Devarapalli, V., Chowdhury, K., and B. Patil, "Proxy Mobile IPv6", RFC 5213, DOI 10.17487/RFC5213, August 2008, <http://www.rfc-editor.org/info/rfc5213>.

[RFC5847] Devarapalli, V., Ed., Koodli, R., Ed., Lim, H., Kant, N., Krishnan, S., and J. Laganier, "Heartbeat Mechanism for Proxy Mobile IPv6", RFC 5847, DOI 10.17487/RFC5847, June 2010, <http://www.rfc-editor.org/info/rfc5847>.

[RFC7563] Pazhyannur, R., Speicher, S., Gundavelli, S., Korhonen, J., and J. Kaippallimalil, "Extensions to the Proxy Mobile IPv6 (PMIPv6) Access Network Identifier Option", RFC 7563, DOI 10.17487/RFC7563, June 2015, <http://www.rfc-editor.org/info/rfc7563>.

8.2. Informative References

[RFC2460] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, DOI 10.17487/RFC2460, December 1998, <http://www.rfc-editor.org/info/rfc2460>.

[RFC6275] Perkins, C., Ed., Johnson, D., and J. Arkko, "Mobility Support in IPv6", RFC 6275, DOI 10.17487/RFC6275, July 2011, <http://www.rfc-editor.org/info/rfc6275>.

Authors' Addresses

Dhananjay Patki
Cisco
Cessna Business Park SEZ, Kadubeesanahalli
Bangalore, Karnataka 560087
India

Email: dhpatki@cisco.com

Sri Gundavelli
Cisco
170 West Tasman Drive
San Jose, CA 95134
USA

Email: sgundave@cisco.com

Jong-Hyouk Lee
Sangmyung University
31, Sangmyeongdae-gil, Dongnam-gu
Cheonan 330-720
Republic of Korea

Email: jonghyouk@smu.ac.kr

Qiao Fu
China Mobile
Xuanwumenxi Ave. No.32
Beijing
China

Email: fuqiao1@outlook.com

Lyle T Bertz
Sprint
Kansas
USA

Email: Lyle.T.Bertz@sprint.com

Distributed Mobility Management [dmm]
Internet-Draft
Intended status: Standards Track
Expires: September 19, 2018

C. Perkins
Futurewei
V. Devarapalli
Vasona Networks

March 18, 2018

MN Identifier Types for RFC 4283 Mobile Node Identifier Option
draft-ietf-dmm-4283mnids-08.txt

Abstract

Additional Identifier Type Numbers are defined for use with the Mobile Node Identifier Option for MIPv6 (RFC 4283).

Status of This Memo

This Internet-Draft will expire on September 19, 2018.

Copyright Notice

Table of Contents

1. Introduction

The Mobile Node Identifier Option for MIPv6 [RFC4283] has proved to be a popular design tool for providing identifiers for mobile nodes during authentication procedures with AAA protocols such as Diameter [RFC3588]. To date, only a single type of identifier has been specified, namely the MN NAI. Other types of identifiers are in common use, and even referenced in RFC 4283. In this document, we propose adding some basic types that are defined in various telecommunications standards, including types for IMSI [ThreeGPP-IDS], P-TMSI [ThreeGPP-IDS], IMEI [ThreeGPP-IDS], and GUTI [ThreeGPP-IDS]. In addition, we specify the IPv6 address itself and IEEE MAC-layer addresses as mobile node identifiers. Defining identifiers that are tied to the physical elements of the device (

MAC address etc.) help in deployment of Mobile IP because in many cases such identifiers are the most natural means for uniquely identifying the device, and will avoid additional look-up steps that might be needed if other identifiers were used.

2. Terminology

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].

3. New Mobile Node Identifier Types

The following types of identifiers are commonly used to identify mobile nodes. For each type, references are provided with full details on the format of the type of identifer.

Mobile Node Identifier Description

Identifier Type	Description	Reference
IPv6 Address		[RFC4291]
IMSI	International Mobile Subscriber Identity	[ThreeGPP-IDS]
P-TMSI	Packet-Temporary Mobile Subscriber Identity	[ThreeGPP-IDS]
GUTI	Globally Unique Temporary ID	[ThreeGPP-IDS]
EUI-48 address	48-bit Extended Unique Identifier	[IEEE802]
EUI-64 address	64-bit Extended Unique Identifier-64 bit	[IEEE802]
DUID	DHCPv6 Unique Identifier	[RFC3315]

Table 1

4. Descriptions of MNID types

In this section descriptions for the various MNID types are provided.

4.1. Description of the IPv6 address type

The IPv6 address [RFC4291] is encoded as a 16 octet string containing a full IPv6 address which has been assigned to the mobile node. The IPv6 address MUST be a unicast routable IPv6 address. Multicast addresses, link-local addresses, and the unspecified IPv6 address MUST NOT be used. IPv6 Unique Local Addresses (ULAs) MAY be used, as long as any security operations making use of the ULA also take into account the domain in which the ULA is guaranteed to be unique.

4.2. Description of the IMSI MNID type

The International Mobile Subscriber Identity (IMSI) [ThreeGPP-IDS] is at most 15 decimal digits (i.e., digits from 0 through 9). The IMSI MUST be encoded as a string of octets in network order (i.e., high-to-low for all digits), where each digit occupies 4 bits. If needed for full octet size, the last digit MUST be padded with 0xf. For example an example IMSI 123456123456789 would be encoded as follows:

```
0x12, 0x34, 0x56, 0x12, 0x34, 0x56, 0x78, 0x9f
```

4.3. Description of the EUI-48 address type

The IEEE EUI-48 address [IEEE802-eui48] is encoded as 6 octets containing the IEEE EUI-48 address.

4.4. Description of the EUI-64 address type

The IEEE EUI-64 address [IEEE802-eui64] is encoded as 8 octets containing the full IEEE EUI-64 address.

4.5. Description of the DUID type

The DUID is the DHCPv6 Unique Identifier (DUID) [RFC3315]. There are various types of DUID, which are distinguished by an initial two-octet type field. Clients and servers MUST treat DUIDs as opaque values and MUST only compare DUIDs for equality.

5. Security Considerations

This document does not introduce any security mechanisms, and does not have any impact on existing security mechanisms.

Mobile Node Identifiers such as those described in this document are considered to be private information. If used in the MNID extension as defined in [RFC4283], the packet including the MNID extension MUST be encrypted so that no personal information or trackable identifiers is inadvertently disclosed to passive observers. Operators can potentially apply IPsec Encapsulating Security Payload (ESP) [RFC4303], in transport mode, with confidentiality and integrity protection for protecting the identity and location information in Mobile IPv6 signaling messages.

Some MNIDs contain sensitive identifiers which, as used in protocols specified by other SDOs, are only used for signaling during initial network entry. In such protocols, subsequent exchanges then rely on a temporary identifier allocated during the initial network entry. Managing the association between long-lived and temporary identifiers is outside the scope of this document.

6. IANA Considerations

The new mobile node identifier types defined in the document should be assigned values from the "Mobile Node Identifier Option Subtypes" registry. The following values should be assigned.

New Mobile Node Identifier Types

Identifier Type	Identifier Type Number
IPv6 Address	2
IMSI	3
P-TMSI	4
EUI-48 address	5
EUI-64 address	6
GUTI	7
DUID-LLT	8
DUID-EN	9
DUID-LL	10
DUID-UUID	11
	12-15 reserved
	16-255 unassigned

Table 2

See Section 4 for additional information about the identifier types. Future new assignments are to be made only after Expert Review [RFC8126]. The expert must ascertain that the identifier type allows unique identification of the mobile device; since all MNIDs require encryption there is no additional privacy exposure attendent to the use of new types.

7. Acknowledgements

The authors wish to acknowledge Hakima Chaouchi, Tatuya Jinmei, Jouni Korhonen, Sri Gundavelli, Suresh Krishnan, Dapeng Liu, Dale Worley, Joseph Salowey, Linda Dunbar, and Mirja Kuehlewind for their helpful comments.

8. References

8.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC3315] Droms, R., Ed., Bound, J., Volz, B., Lemon, T., Perkins, C., and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3315, DOI 10.17487/RFC3315, July 2003, <https://www.rfc-editor.org/info/rfc3315>.

[RFC4283] Patel, A., Leung, K., Khalil, M., Akhtar, H., and K. Chowdhury, "Mobile Node Identifier Option for Mobile IPv6 (MIPv6)", RFC 4283, DOI 10.17487/RFC4283, November 2005, <https://www.rfc-editor.org/info/rfc4283>.

[RFC4291] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, DOI 10.17487/RFC4291, February 2006, <https://www.rfc-editor.org/info/rfc4291>.

[RFC4303] Kent, S., "IP Encapsulating Security Payload (ESP)", RFC 4303, DOI 10.17487/RFC4303, December 2005, <https://www.rfc-editor.org/info/rfc4303>.

[RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <https://www.rfc-editor.org/info/rfc8126>.

8.2. Informative References

[EANUCCGS]
EAN International and the Uniform Code Council, "General EAN.UCC Specifications Version 5.0", Jan 2004.

[EPC-Tag-Data]
EPCglobal Inc., "EPC(TM) Generation 1 Tag Data Standards Version 1.1 Rev.1.27 http://www.gs1.org/gsmp/kc/epcglobal/tds/tds_1_1_rev_1_27-standard-20050510.pdf", January 2005.

[IEEE802] IEEE, "IEEE Std 802: IEEE Standards for Local and Metropolitan Networks: Overview and Architecture", 2001.

[IEEE802-eui48]
IEEE, "Guidelines for 48-Bit Global Identifier (EUI-48) https://standards.ieee.org/develop/regauth/tut/eui48.pdf", 2001.

[IEEE802-eui64]
IEEE, "Guidelines for 64-Bit Global Identifier (EUI-64) https://standards.ieee.org/develop/regauth/tut/eui.pdf64", 2001.

[RFC3588] Calhoun, P., Loughney, J., Guttman, E., Zorn, G., and J. Arkko, "Diameter Base Protocol", RFC 3588, DOI 10.17487/RFC3588, September 2003, <https://www.rfc-editor.org/info/rfc3588>.

[RFID-DoD-spec]
Department of Defense, "United States Department of Defense Suppliers Passive RFID Information Guide (Version 15.0)", January 2010.

[RFID-framework]
Institut National des Telecommunication, ""Heterogeneous RFID framework design, analysis and evaluation"", July 2012.

[ThreeGPP-IDS]
3rd Generation Partnership Project, "3GPP Technical Specification 23.003 V8.4.0: Technical Specification Group Core Network and Terminals; Numbering, addressing and identification (Release 8)", March 2009.

[TRACK-IoT]
IPv6.com, ""Heterogeneous IoT Network : TRACK-IoT"", March 2012.

[Using-RFID-IPv6]
IPv6.com, ""Using RFID & IPv6"", September 2006.

Appendix A. RFID types

The material in this non-normative appendix was originally composed for inclusion in the main body of the specification, but was moved into an appendix because there was insufficient support for allocating RFID types at this time. It was observed that RFID-based mobile devices may create privacy exposures unless confidentiality is assured for signaling. A specification for eliminating unauthorized RFID tracking based on layer-2 addresses would be helpful.

Much of the following text is due to contributions from Hakima Chaouchi. For an overview and some initial suggestions about using RFID with IPv6 on mobile devices, see [Using-RFID-IPv6].

In the context of IoT and industry 4.0 vertical domain, efficient inventory and tracking items is of major interest, and RFID technology is the identification technology in the hardware design of many such items.

The "TRACKIOT: Heterogeneous IoT control" project ([TRACK-IoT], [RFID-framework]) explored Mobile IPv6 as a mobility management protocol for RFID-based mobile devices.

1. Passive RFID tags (that have no processing resources) need to be handled by the gateway (likely also the RFID Reader), which is then the end point of the mobility protocol. It is also the point where the CoA will be created based on some combination such as the RFID tag and the prefix of that gateway. The point here is to offer the possibility to passive RFID items to get an IPv6 address and take advantage of the mobility framework to follow the mobile device (passive tag on the item). One example scenario that has been proposed, showing the need for mobility management of passive RFID items, would be pieces of art tagged with passive tags that need to be monitored while transported.
2. Using active RFID tags (where processing resource is available on the tag), the end point of the mobility protocol can be pushed up to the RFID Active tag. We name it also an identification sensor. Use cases include active RFID tags for traceability of cold food respect during mobility (transport) of food. Mobility of cars equiped with active RFID tags that we already use for toll payement can be added with mobility management.

One major effort of connecting IETF efforts to the EPCGlobal (RFID standardisation) led to the ONS (DNS version applied for RFID logical names and page information retrieval). Attempts have tried to connect IPv6 on the address space to RFID identifier format. Other initiatives started working on gateways to map tag identifiers with IPv6 addresses and build signaling protocols for the application level. For instance tracking of mobile items equipped with a tag can be triggered remotely by a remote correspondent node until a visiting area where a mobile item equipped with an RFID tag is located. An RFID reader will be added with an IPv6 to RFID tag translation. One option is to build a Home IPv6 address of that tagged item by using the prefix of the Home agent combined with the tag RFID identifier of the mobile item; as the tag ID is unique, the home IPv6 address of that item will be also unique. Then the visiting RFID reader will compose the IPV6 care of address of the tagged mobile item by combining the prefix of the RFID reader with the tag ID of the item).

MIPv6 can then provide normally the mobility management of that RFID tagged item. A different useful example of tagged items involves items of a factory that can be tracked while they are transported, especially for real time localisation and tracking of precious items transported without GPS. An automotive car manufacturer can assign IPv6 addresses corresponding to RFID tagged cars or mechanical car parts, and build a tracking dataset of the mobility not only of the cars, but also of the mechanical pieces.

The Tag Data standard promoted by Electronic Product Code(TM) (abbreviated EPC) [EPC-Tag-Data] supports several encoding systems or schemes, which are commonly used in RFID (radio-frequency identification) applications, including

- RFID-GID (Global Identifier),
- RFID-SGTIN (Serialized Global Trade Item Number),
- RFID-SSCC (Serial Shipping Container),
- RFID-SGLN (Global Location Number),
- RFID-GRAI (Global Returnable Asset Identifier),
- RFID-DOD (Department of Defense ID), and
- RFID-GIAI (Global Individual Asset Identifier).

For each RFID scheme except GID, there are three representations:

- a 64-bit binary representation (for example, SGLN-64) (except for GID)
- a 96-bit binary representation (SGLN-96)
- a representation as a URI

The URI representation for the RFID is actually a URN. The EPC document has the following language:

> All categories of URIs are represented as Uniform Reference Names (URNs) as defined by [RFC2141], where the URN Namespace is epc.

The following list includes the above RFID types.

Mobile Node RFID Identifier Description

Identifier Type	Description	Reference
RFID-SGTIN-64	64-bit Serialized Global Trade Item Number	[EPC-Tag-Data]
RFID-SSCC-64	64-bit Serial Shipping Container	[EPC-Tag-Data]
RFID-SGLN-64	64-bit Serialized Global Location Number	[EPC-Tag-Data]
RFID-GRAI-64	64-bit Global Returnable Asset Identifier	[EPC-Tag-Data]
RFID-DOD-64	64-bit Department of Defense ID	[RFID-DoD-spec]
RFID-GIAI-64	64-bit Global Individual Asset Identifier	[EPC-Tag-Data]
RFID-GID-96	96-bit Global Identifier	[EPC-Tag-Data]
RFID-SGTIN-96	96-bit Serialized Global Trade Item Number	[EPC-Tag-Data]
RFID-SSCC-96	96-bit Serial Shipping Container	[EPC-Tag-Data]
RFID-SGLN-96	96-bit Serialized Global Location Number	[EPC-Tag-Data]
RFID-GRAI-96	96-bit Global Returnable Asset Identifier	[EPC-Tag-Data]
RFID-DOD-96	96-bit Department of Defense ID	[RFID-DoD-spec]
RFID-GIAI-96	96-bit Global Individual Asset Identifier	[EPC-Tag-Data]
RFID-GID-URI	Global Identifier represented as URI	[EPC-Tag-Data]
RFID-SGTIN-URI	Serialized Global Trade Item Number represented as URI	[EPC-Tag-Data]
RFID-SSCC-URI	Serial Shipping Container represented as URI	[EPC-Tag-Data]
RFID-SGLN-URI	Global Location Number represented as URI	[EPC-Tag-Data]
RFID-GRAI-URI	Global Returnable Asset Identifier represented as URI	[EPC-Tag-Data]
RFID-DOD-URI	Department of Defense ID represented as URI	[RFID-DoD-spec]
RFID-GIAI-URI	Global Individual Asset Identifier represented as URI	[EPC-Tag-Data]

Table 3

A.1. Description of the RFID types

The General Identifier (GID) that is used with RFID is composed of three fields - the General Manager Number, Object Class and Serial Number. The General Manager Number identifies an organizational entity that is responsible for maintaining the numbers in subsequent fields. GID encodings include a fourth field, the header, to guarantee uniqueness in the namespace defined by EPC.

Some of the RFID types depend on the Global Trade Item Number (GTIN) code defined in the General EAN.UCC Specifications [EANUCCGS]. A GTIN identifies a particular class of object, such as a particular kind of product or SKU.

The EPC encoding scheme for SGTIN permits the direct embedding of EAN.UCC System standard GTIN and Serial Number codes on EPC tags. In all cases, the check digit is not encoded. Two encoding schemes are specified, SGTIN-64 (64 bits) and SGTIN-96 (96 bits).

The Serial Shipping Container Code (SSCC) is defined by the EAN.UCC Specifications. Unlike the GTIN, the SSCC is already intended for assignment to individual objects and therefore does not require additional fields to serve as an EPC pure identity. Two encoding schemes are specified, SSCC-64 (64 bits) and SSCC-96 (96 bits).

The Global Location Number (GLN) is defined by the EAN.UCC Specifications. A GLN can represent either a discrete, unique physical location such as a warehouse slot, or an aggregate physical location such as an entire warehouse. In addition, a GLN can represent a logical entity that performs a business function such as placing an order. The Serialized Global Location Number (SGLN) includes the Company Prefix, Location Reference, and Serial Number.

The Global Returnable Asset Identifier (GRAI) is defined by the General EAN.UCC Specifications. Unlike the GTIN, the GRAI is already intended for assignment to individual objects and therefore does not require any additional fields to serve as an EPC pure identity. The GRAI includes the Company Prefix, Asset Type, and Serial Number.

The Global Individual Asset Identifier (GIAI) is defined by the General EAN.UCC Specifications. Unlike the GTIN, the GIAI is already intended for assignment to individual objects and therefore does not require any additional fields to serve as an EPC pure identity. The GRAI includes the Company Prefix, and Individual Asset Reference.

The DoD Construct identifier is defined by the United States Department of Defense (DoD). This tag data construct may be used to encode tags for shipping goods to the DoD by a supplier who has already been assigned a CAGE (Commercial and Government Entity) code.

A.1.1. Description of the RFID-SGTIN-64 type

The RFID-SGTIN-64 is encoded as specified in [EPC-Tag-Data]. The SGTIN-64 includes five fields: Header, Filter Value (additional data that is used for fast filtering and pre-selection), Company Prefix Index, Item Reference, and Serial Number. Only a limited number of Company Prefixes can be represented in the 64-bit tag.

A.1.2. Description of the RFID-SGTIN-96 type

The RFID-SGTIN-96 is encoded as specified in [EPC-Tag-Data]. The SGTIN-96 includes six fields: Header, Filter Value, Partition (an indication of where the subsequent Company Prefix and Item Reference numbers are divided), Company Prefix Index, Item Reference, and Serial Number.

A.1.3. Description of the RFID-SSCC-64 type

The RFID-SSCC-64 is encoded as specified in [EPC-Tag-Data]. The SSCC-64 includes four fields: Header, Filter Value, Company Prefix Index, and Serial Reference. Only a limited number of Company Prefixes can be represented in the 64-bit tag.

A.1.4. Description of the RFID-SSCC-96 type

The RFID-SSCC-96 is encoded as specified in [EPC-Tag-Data]. The SSCC-96 includes six fields: Header, Filter Value, Partition, Company Prefix, and Serial Reference, as well as 24 bits that remain Unallocated and must be zero.

A.1.5. Description of the RFID-SGLN-64 type

The RFID-SGLN-64 type is encoded as specified in [EPC-Tag-Data]. The SGLN-64 includes five fields: Header, Filter Value, Company Prefix Index, Location Reference, and Serial Number.

A.1.6. Description of the RFID-SGLN-96 type

The RFID-SGLN-96 type is encoded as specified in [EPC-Tag-Data]. The SGLN-96 includes six fields: Header, Filter Value, Partition, Company Prefix, Location Reference, and Serial Number.

A.1.7. Description of the RFID-GRAI-64 type

The RFID-GRAI-64 type is encoded as specified in [EPC-Tag-Data]. The GRAI-64 includes five fields: Header, Filter Value, Company Prefix Index, Asset Type, and Serial Number.

A.1.8. Description of the RFID-GRAI-96 type

The RFID-GRAI-96 type is encoded as specified in [EPC-Tag-Data]. The GRAI-96 includes six fields: Header, Filter Value, Partition, Company Prefix, Asset Type, and Serial Number.

A.1.9. Description of the RFID-GIAI-64 type

The RFID-GIAI-64 type is encoded as specified in [EPC-Tag-Data]. The GIAI-64 includes four fields: Header, Filter Value, Company Prefix Index, and Individual Asset Reference.

A.1.10. Description of the RFID-GIAI-96 type

The RFID-GIAI-96 type is encoded as specified in [EPC-Tag-Data]. The GIAI-96 includes five fields: Header, Filter Value, Partition, Company Prefix, and Individual Asset Reference.

A.1.11. Description of the RFID-DoD-64 type

The RFID-DoD-64 type is encoded as specified in [RFID-DoD-spec]. The DoD-64 type includes four fields: Header, Filter Value, Government Managed Identifier, and Serial Number.

A.1.12. Description of the RFID-DoD-96 type

The RFID-DoD-96 type is encoded as specified in [RFID-DoD-spec]. The DoD-96 type includes four fields: Header, Filter Value, Government Managed Identifier, and Serial Number.

A.1.13. Description of the RFID URI types

In some cases, it is desirable to encode in URI form a specific encoding of an RFID tag. For example, an application may prefer a URI representation for report preparation. Applications that wish to manipulate any additional data fields on tags may need some representation other than the pure identity forms.

For this purpose, the fields as represented the previous sections are associated with specified fields in the various URI types. For instance, the URI may have fields such as CompanyPrefix, ItemReference, or SerialNumber. For details and encoding specifics, consult [EPC-Tag-Data].

Authors’ Addresses

Charles E. Perkins
Futurewei Inc.
2330 Central Expressway
Santa Clara, CA 95050
USA

Phone: +1-408-330-4586
Email: charliep@computer.org

Vijay Devarapalli
Vasona Networks
2900 Lakeside Drive, Suite 180
Santa Clara, CA 95054
USA

Email: dvijay@gmail.com

DMM Working Group
Internet-Draft
Intended status: Standards Track
Expires: December 20, 2018

S. Matsushima
SoftBank
L. Bertz
Sprint
M. Liebsch
NEC
S. Gundavelli
Cisco
D. Moses
Intel Corporation
C. Perkins
Futurewei

June 18, 2018

Protocol for Forwarding Policy Configuration (FPC) in DMM
draft-ietf-dmm-fpc-cpdp-12

Abstract

This document describes a way, called Forwarding Policy Configuration (FPC) to manage the separation of data-plane and control-plane. FPC defines a flexible mobility management system using FPC agent and FPC client functions. A FPC agent provides an abstract interface to the data-plane. The FPC client configures data-plane nodes by using the functions and abstractions provided by the FPC agent for the data-plane nodes. The data-plane abstractions presented in this document are extensible in order to support many different types of mobility management systems and data-plane functions.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

This document describes Forwarding Policy Configuration (FPC), a system for managing the separation of control-plane and data-plane. FPC enables flexible mobility management using FPC client and FPC agent functions. A FPC agent exports an abstract interface representing the data-plane. To configure data-plane nodes and functions, the FPC client uses the interface to the data-plane offered by the FPC agent.

Control planes of mobility management systems, or related applications which require data-plane control, can utilize the FPC client at various levels of abstraction. FPC operations are capable of directly configuring a single Data-Plane Node (DPN), as well as multiple DPNs, as determined by the data-plane models exported by the FPC agent.

A FPC agent represents the data-plane operation according to several basic information models. A FPC agent also provides access to Monitors, which produce reports when triggered by events or FPC Client requests regarding Mobility Contexts, DPNs or the Agent.

To manage mobility sessions, the FPC client assembles applicable sets of forwarding policies from the data model, and configures them on the appropriate FPC Agent. The Agent then renders those policies into specific configurations for each DPN at which mobile nodes are attached. The specific protocols and configurations to configure a DPN from a FPC Agent are outside the scope of this document.

A DPN is a logical entity that performs data-plane operations (packet movement and management). It may represent a physical DPN unit, a sub-function of a physical DPN or a collection of physical DPNs (i.e., a "virtual DPN"). A DPN may be virtual -- it may export the FPC DPN Agent interface, but be implemented as software that controls other data-plane hardware or modules that may or may not be FPC-compliant. In this document, DPNs are specified without regard for whether the implementation is virtual or physical. DPNs are connected to provide mobility management systems such as access networks, anchors and domains. The FPC agent interface enables establishment of a topology for the forwarding plane.

When a DPN is mapped to physical data-plane equipment, the FPC client can have complete knowledge of the DPN architecture, and use that information to perform DPN selection for specific sessions. On the other hand, when a virtual DPN is mapped to a collection of physical DPNs, the FPC client cannot select a specific physical DPN because it is hidden by the abstraction; only the FPC Agent can address the specific associated physical DPNs. Network architects have the flexibility to determine which DPN-selection capabilities are performed by the FPC Agent (distributed) and which by the FPC client (centralized). In this way, overlay networks can be configured without disclosing detailed knowledge of the underlying hardware to the FPC client and applications.

The abstractions in this document are designed to support many different mobility management systems and data-plane functions. The architecture and protocol design of FPC is not tied to specific types of access technologies and mobility protocols.

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].

Attribute Expression: The definition of a template Property. This includes setting the type, current value, default value and if the attribute is static, i.e. can no longer be changed.

Domain: One or more DPNs that form a logical partition of network resources (e.g., a data-plane network under common network administration). A FPC client (e.g., a mobility management system) may utilize a single or multiple domains.

DPN: A data-plane node (DPN) is capable of performing data-plane features. For example, DPNs may be switches or routers, regardless of whether they are realized as hardware or purely in software.

FPC Client: A FPC Client is integrated with a mobility management system or related application, enabling control over forwarding policy, mobility sessions and DPNs via a FPC Agent.

Mobility Context: A Mobility Context contains the data-plane information necessary to efficiently send and receive traffic from a mobile node. This includes policies that are created or modified during the network’s operation - in most cases, on a per-flow or per session basis. A Mobility-Context represents the mobility sessions (or flows) which are active on a mobile node. This includes associated runtime attributes, such as tunnel endpoints, tunnel identifiers, delegated prefix(es), routing information, etc. Mobility-Contexts are associated to specific DPNs. Some pre-defined Policies may apply during mobility signaling requests. The Mobility Context supplies information about the policy settings specific to a mobile node and its flows; this information is often quite dynamic.

Mobility Session: Traffic to/from a mobile node that is expected to survive reconnection events.

Monitor: A reporting mechanism for a list of events that trigger notification messages from a FPC Agent to a FPC Client.

Policy: A Policy determines the mechanisms for managing specific traffic flows or packets. Policies specify QoS, rewriting rules for packet processing, etc. A Policy consists of one or more rules. Each rule is composed of a Descriptor and Actions. The Descriptor in a rule identifies packets (e.g., traffic flows), and the Actions apply treatments to packets that match the Descriptor in the rule. Policies can apply to Domains, DPNs, Mobile Nodes, Service-Groups, or particular Flows on a Mobile Node.

Property: An attribute-value pair for an instance of a FPC entity.

Service-Group: A set of DPN interfaces that support a specific data-plane purpose, e.g. inbound/outbound, roaming, subnetwork with common specific configuration, etc.

Template: A recipe for instantiating FPC entities. Template definitions are accessible (by name or by a key) in an indexed set. A Template is used to create specific instances (e.g., specific policies) by assigning appropriate values into the Template definition via Attribute Expression.

Template Configuration The process by which a Template is referenced (by name or by key) and Attribute Expressions are created that change the value, default value or static nature of the Attribute, if permitted. If the Template is Extensible, new attributes MAY be added.

Tenant: An operational entity that manages mobility management systems or applications which require data-plane functions. A Tenant defines a global namespace for all entities owned by the Tenant enabling its entities to be used by multiple FPC Clients across multiple FPC Agents.

Topology: The DPNs and the links between them. For example, access nodes may be assigned to a Service-Group which peers to a Service-Group of anchor nodes.

3. FPC Design Objectives and Deployment

Using FPC, mobility control-planes and applications can configure DPNs to perform various mobility management roles as described in [I-D.ietf-dmm-deployment-models]. This fulfills the requirements described in [RFC7333].

This document defines FPC Agent and FPC Client, as well as the information models that they use. The attributes defining those models serve as the protocol elements for the interface between the FPC Agent and the FPC Client.

Mobility control-plane applications integrate features offered by the FPC Client. The FPC Client connects to FPC Agent functions. The Client and the Agent communicate based on information models described in Section 4. The models allow the control-plane to configure forwarding policies on the Agent for data-plane communications with mobile nodes.

Once the Topology of DPN(s) and domains are defined on an Agent for a data plane, the DPNs in the topology are available for further configuration. The FPC Agent connects those DPNs to manage their configurations.

A FPC Agent configures and manages its DPN(s) according to forwarding policies requested and Attributes provided by the FPC Client. Configuration commands used by the FPC agent to configure its DPN node(s) may be specific to the DPN implementation; consequently the method by which the FPC Agent carries out the specific configuration for its DPN(s) is out of scope for this document. Along with the data models, the FPC Client (on behalf of control-plane and applications) requests that the Agent configures Policies prior to the time when the DPNs start forwarding data for their mobility sessions.

This architecture is illustrated in Figure 1. A FPC Agent may be implemented in a network controller that handles multiple DPNs, or (more simply) an FPC Agent may itself be integrated into a DPN.

This document does not specify a protocol for the FPC interface; it is out of scope. However, an implementation must support the FPC transactions described in Section 5.

```
                           +-------------------------+
                           | Mobility Control-Plane  |
                           |          and            |
                           |      Applications       |
                           |+-----------------------+|
                           ||      FPC Client       ||
                           |+----------^------------+|
                           +-----------|-------------+
               FPC interface protocol  |
                       +---------------+-----------------+
                       |                                 |
         Network       |                                 |
         Controller    |                      DPN        |
           +-----------|-------------+        +----------|---------+
           |+----------v------------+|        |+---------v--------+|
           ||   [Data-plane model]  ||        ||[Data-plane model]||
           ||       FPC Agent       ||        ||    FPC Agent     ||
           |+-----------------------+|        |+------------------+|
           |+------------+----------+|        |                    |
           ||SB Protocol |FPC Client||        |  DPN Configuration |
           ||   Modules  |  Module  ||        +--------------------+
           |+------^-----+----^-----+|
           +-------|----------|------+
                   |          |
         Other     |          | FPC interface
        southbound |          | protocol
         protocols |          |
                   |          +-----------------+
                   |                            |
       DPN         |                 DPN        |
        +----------|---------+       +----------|---------+
        |+---------v--------+|       |+---------v--------+|
        ||  Configuration   ||       ||[Data-plane model]||
        || Protocol module  ||       ||     FPC Agent    ||
        |+------------------+|       |+------------------+|
        |                    |       |                    |
        | DPN Configuration  |       |  DPN Configuration |
        +--------------------+       +--------------------+
```

Figure 1: Reference Forwarding Policy Configuration (FPC) Architecture

The FPC architecture supports multi-tenancy; a FPC enabled data-plane supports tenants of multiple mobile operator networks and/or applications. It means that the FPC Client of each tenant connects to the FPC Agent and it MUST partition namespace and data for their data-planes. DPNs on the data-plane may fulfill multiple data-plane roles which are defined per session, domain and tenant.

Multi-tenancy permits the paritioning of data-plane entities as well as a common namespace requirement upon FPC Agents and Clients when they use the same Tenant for a common data-plane entity.

FPC information models often configuration to fit the specific needs for DPN management of a mobile node’s traffic. The FPC interfaces in Figure 1 are the only interfaces required to handle runtime data in a Mobility Context. The Topology and some Policy FPC models MAY be pre-configured; in that case real-time protocol exchanges are not required for them.

The information model provides an extensibility mechanism through Templates that permits specialization for the needs of a particular vendor’s equipment or future extension of the model presented in this specification.

4. FPC Mobility Information Model

The FPC information model includes the following components:

DPN Information Model,
Topology Information Model,
Policy Information Model,
Mobility-Context, and
Monitor, as illustrated in Figure 2.

```
                   :
                   |
                   +-[FPC Mobility Information Model]
                   |          |
                   |          +-[Topology Information Model]
                   |          |
                   |          +-[Policy Information Model]
                   |          |
                   |          +-[Mobility-Context]
                   |          |
                   |          +-[Monitor]
                   |
```

Figure 2: FPC Information Model structure

4.1. Model Notation and Conventions

The following conventions are used to describe the FPC information models.

Information model entities (e.g. DPNs, Rules, etc.) are defined in a hierarchical notation where all entities at the same hierarchical level are located on the same left-justified vertical position sequentially. When entities are composed of sub-entities, the sub-entities appear shifted to the right, as shown in Figure 3.

```
                          |
                          +-[entity2]
                          |         +-[entity2.1]
                          |         +-[entity2.2]
```

Figure 3: Model Notation - An Example

Some entities have one or more qualifiers placed on the right hand side of the element definition in angle-brackets. Common types include:

List: A collection of entities (some could be duplicated)

Set: A nonempty collection of entities without duplications

Name: A human-readable string

Key: A unique value. We distinguish 3 types of keys:

U-Key: A key unique across all Tenants. U-Key spaces typically involve the use of registries or language specific mechanisms that guarantee universal uniqueness of values.

G-Key: A key unique within a Tenant

L-Key: A key unique within a local namespace. For example, there may exist interfaces with the same name, e.g. "if0", in two different DPNs but there can only be one "if0" within each DPN (i.e. its local Interface-Key L-Key space).

Each entity or attribute may be optional (O) or mandatory (M). Entities that are not marked as optional are mandatory.

```
       The following example shows 3 entities:
           -- Entity1 is a globally unique key, and optionally can have
                      an associated Name
           -- Entity2 is a list
           -- Entity3 is a set and is optional
                      +
                      |
                      +-[entity1] <G-Key> (M), <Name> (O)
                      +-[entity2] <List>
                      +-[entity3] <Set> (O)
                      |
                      +
```

Figure 4

When expanding entity1 into a modeling language such as YANG it would result in two values: entity1-Key and entity1-Name.

To encourage re-use, FPC defines indexed sets of various entity Templates. Other model elements that need access to an indexed model entity contain an attribute which is always denoted as "entity-Key". When a Key attribute is encountered, the referencing model element may supply attribute values for use when the referenced entity model is instantiated. For example: Figure 5 shows 2 entities:

EntityA definition references an entityB model element.

EntityB model elements are indexed by entityB-Key.

Each EntityB model element has an entityB-Key which allows it to be uniquely identified, and a list of Attributes (or, alternatively, a Type) which specifies its form. This allows a referencing entity to create an instance by supplying entityB-Values to be inserted, in a Settings container.

```
                           .
                           .
                           |
                           +-[entityA]
                           |      +-[entityB-Key]
                           |      +-[entityB-Values]
                           .
                           .
                           |
                           +-[entityB] <L-Key> (M) <Set>
                           |      +-[entityB-Type]
                           .
                           .
```

Figure 5: Indexed sets of entities

Indexed sets are specified for each of the following kinds of entities:

Domain (See Section 4.9.3)
DPN (See Section 4.9.4)
Policy (See Section 4.9.5)
Rule (See Section 4.9.5)
Descriptor (See Figure 12)
Action (See Figure 12)
Service-Group (See Section 4.9.2, and
Mobility-Context (See Section 4.9.6)

As an example, for a Domain entity, there is a corresponding attribute denoted as "Domain-Key" whose value can be used to determine a reference to the Domain.

4.2. Templates and Attributes

In order to simplify development and maintenance of the needed policies and other objects used by FPC, the Information Models which are presented often have attributes that are not initialized with their final values. When an FPC entity is instantiated according to a template definition, specific values need to be configured for each such attribute. For instance, suppose an entity Template has an Attribute named "IPv4-Address", and also suppose that a FPC Client instantiates the entity and requests that it be installed on a DPN. An IPv4 address will be needed for the value of that Attribute before the entity can be used.

```
+-[Template] <U-Key, Name> (M) <Set>
|      +-[Attributes] <Set> (M)
|      +-[Extensible ˜ FALSE]
|      +-[Entity-State ˜ Initial]
|      +-[Version]
```

Figure 6: Template entities

Attributes: A set of Attribute names MAY be included when defining a Template for instantiating FPC entities.

Extensible: Determines whether or not entities instantiated from the Template can be extended with new non-mandatory Attributes not originally defined for the Template. Default value is FALSE. If a Template does not explicitly specify this attribute, the default value is considered to be in effect.

Entity-State: Either Initial, PartiallyConfigured, Configured, or Active. Default value is Initial. See Section 4.6 for more information about how the Entity-Status changes during the configuration steps of the Entity.

Version: Provides a version tag for the Template.

The Attributes in an Entity Template may be either mandatory or non-mandatory. Attribute values may also be associated with the attributes in the Entity Template. If supplied, the value may be either assigned with a default value that can be reconfigured later, or the value can be assigned with a static value that cannot be reconfigured later (see Section 4.3).

It is possible for a Template to provide values for all of its Attributes, so that no additional values are needed before the entity can made Active. Any instantiation from a Template MUST have at least one Attribute in order to be a useful entity unless the Template has none.

4.3. Attribute-Expressions

The syntax of the Attribute definition is formatted to make it clear. For every Attribute in the Entity Template, six possibilities are specified as follows:

’[Att-Name:]’ Mandatory Attribute is defined, but template does not provide any configured value.

’[Att-Name: Att-Value]’ Mandatory Attribute is defined, and has a statically configured value.

'[Att-Name: ˜ Att-Value]' Mandatory Attribute is defined, and has a default value.

'[Att-Name]' Non-mandatory Attribute may be included but template does not provide any configured value.

'[Att-Name = Att-Value]' Non-mandatory Attribute may be included and has a statically configured value.

'[Att-Name ˜ Att-Value]' Non-mandatory Attribute may be included and has a default value.

So, for example, a default value for a non-mandatory IPv4-Address attribute would be denoted by [IPv4-Address ˜ 127.0.0.1].

After a FPC Client identifies which additional Attributes have been configured to be included in an instantiated entity, those configured Attributes MUST NOT be deleted by the FPC Agent. Similarly, any statically configured value for an entity Attribute MUST NOT be changed by the FPC Agent.

Whenever there is danger of confusion, the fully qualified Attribute name MUST be used when supplying needed Attribute Values for a structured Attribute.

4.4. Attribute Value Types

For situations in which the type of an attribute value is required, the following syntax is recommended. To declare than an attribute has data type "foo", typecast the attribute name by using the parenthesized data type (foo). So, for instance, [(float) Max-Latency-in-ms:] would indicate that the mandatory Attribute "Max-Latency-in-ms" requires to be configured with a floating point value before the instantiated entity could be used. Similarly, [(float) Max-Latency-in-ms: 9.5] would statically configure a floating point value of 9.5 to the mandatory Attribute "Max-Latency-in-ms".

4.5. Namespace and Format

The identifiers and names in FPC models which reside in the same Tenant must be unique. That uniqueness must be maintained by all Clients, Agents and DPNs that support the Tenant. The Tenant namespace uniqueness MUST be applied to all elements of the tenant model, i.e. Topology, Policy and Mobility models.

When a Policy needs to be applied to Mobility-Contexts in all Tenants on an Agent, the Agent SHOULD define that policy to be visible by all Tenants. In this case, the Agent assigns a unique identifier in the

Agent namespace and copies the values to each Tenant. This effectively creates a U-Key although only a G-Key is required within the Tenant.

The notation for identifiers can utilize any format with agreement between data-plane agent and client operators. The formats include but are not limited to Globally Unique IDentifiers (GUIDs), Universally Unique IDentifiers (UUIDs), Fully Qualified Domain Names (FQDNs), Fully Qualified Path Names (FQPNs) and Uniform Resource Identifiers (URIs). The FPC model does not limit the format, which could dictate the choice of FPC protocol. Nevertheless, the identifiers which are used in a Mobility model should be considered to efficiently handle runtime parameters.

There are identifiers reserved for Protocol Operation. See Section 5.1.1.5 for details.

4.6. Configuring Attribute Values

Attributes of Information Model components such as policy templates are configured with values as part of FPC configuration operations. There may be several such configuration operations before the template instantiation is fully configured.

Entity-Status indicates when an Entity is usable within a DPN. This permits DPN design tradeoffs amongst local storage (or other resources), over the wire request size and the speed of request processing. For example, DPN designers with constrained systems MAY only house entities whose status is Active which may result in sending over all policy information with a Mobility-Context request. Storing information elements with an entity status of "PartiallyConfigured" on the DPN requires more resources but can result in smaller over the wire FPC communication and request processing efficiency.

When the FPC Client instantiates a Policy from a Template, the Policy-Status is "Initial". When the FPC Client sends the policy to a FPC Agent for installation on a DPN, the Client often will configure appropriate attribute values for the installation, and accordingly changes the Policy-Status to "PartiallyConfigured" or "Configured". The FPC Agent will also configure Domain-specific policies and DPN-specific policies on the DPN. When configured to provide particular services for mobile nodes, the FPC Agent will apply whatever service-specific policies are needed on the DPN. When a mobile node attaches to the network data-plane within the topology under the jurisdiction of a FPC Agent, the Agent may apply policies and settings as appropriate for that mobile node. Finally, when the mobile node launches new flows, or quenches existing flows, the FPC

Agent, on behalf of the FPC Client, applies or deactivates whatever policies and attribute values are appropriate for managing the flows of the mobile node. When a "Configured" policy is de-activated, Policy-Status is changed to be "Active". When an "Active" policy is activated, Policy-Status is changed to be "Configured".

Attribute values in DPN resident Policies may be configured by the FPC Agent as follows:

Domain-Policy-Configuration: Values for Policy attributes that are required for every DPN in the domain.

DPN-Policy-Configuration: Values for Policy attributes that are required for every policy configured on this DPN.

Service-Group-Policy-Configuration: Values for Policy attributes that are required to carry out the intended Service of the Service Group.

MN-Policy-Configuration: Values for Policy attributes that are required for all traffic to/from a particular mobile node.

Service-Data-Flow-Policy-Configuration: Values for Policy attributes that are required for traffic belonging to a particular set of flows on the mobile node.

Any configuration changes MAY also supply updated values for existing default attribute values that may have been previously configured on the DPN resident policy.

Entity blocks describe the format of the policy configurations.

4.7. Entity Configuration Blocks

As described in Section 4.6, a Policy Template may be configured in several stages by configuring default or missing values for Attributes that do not already have statically configured values. A Policy-Configuration is the combination of a Policy-Key (to identify the Policy Template defining the Attributes) and the currently configured Attribute Values to be applied to the Policy Template. Policy-Configurations MAY add attributes to a Template if Extensible is True. They MAY also refine existing attributes by:

assign new values if the Attribute is not static

make attributes static if they were not

make an attribute mandatory

A Policy-Configuration MUST NOT define or refine an attribute twice. More generally, an Entity-Configuration can be defined for any configurable Indexed Set to be the combination of the Entity-Key along with a set of Attribute-Expressions that supply configuration information for the entity’s Attributes. Figure 7 shows a schematic representation for such Entity Configuration Blocks.

```
[Entity Configuration Block]
|       +-[Entity-Key] (M)
|       +-[Attribute-Expression] <Set> (M)
```

Figure 7: Entity Configuration Block

This document makes use of the following kinds of Entity Configuration Blocks:

Descriptor-Configuration

Action-Configuration

Rule-Configuration

Interface-Configuration

Service-Group-Configuration

Domain-Policy-Configuration

DPN-Policy-Configuration

Policy-Configuration

MN-Policy-Configuration

Service-Data-Flow-Policy-Configuration

4.8. Information Model Checkpoint

The Information Model Checkpoint permits Clients and Tenants with common scopes, referred to in this specification as Checkpoint BaseNames, to track the state of provisioned information on an Agent. The Agent records the Checkpoint BaseName and Checkpoint value set by a Client. When a Client attaches to the Agent it can query to determine the amount of work that must be executed to configure the Agent to a specific BaseName / checkpoint revision.

Checkpoints are defined for the following information model components:

Service-Group

DPN Information Model

Domain Information Model

Policy Information Model

4.9. Information Model Components

4.9.1. Topology Information Model

The Topology structure specifies DPNs and the communication paths between them. A network management system can use the Topology to select the most appropriate DPN resources for handling specific session flows.

The Topology structure is illustrated in Figure 8 (for definitions see Section 2):

```
|
+-[Topology Information Model]
|          +-[Extensible: FALSE]
|          +-[Service-Group]
|          +-[DPN] <Set>
|          +-[Domain] <Set>
```

Figure 8: Topology Structure

4.9.2. Service-Group

Service-Group-Set is collection of DPN interfaces serving some data-plane purpose including but not limited to DPN Interface selection to fulfill a Mobility-Context. Each Group contains a list of DPNs (referenced by DPN-Key) and selected interfaces (referenced by Interface-Key). The Interfaces are listed explicitly (rather than referred implicitly by its specific DPN) so that every Interface of a DPN is not required to be part of a Group. The information provided is sufficient to ensure that the Protocol, Settings (stored in the Service-Group-Configuration) and Features relevant to successful interface selection is present in the model.

```
   |
   +-[Service-Group] <G-Key>, <Name> (O) <Set>
   |           +-[Extensible: FALSE]
   |           +-[Role] <U-Key>
   |           +-[Protocol] <Set>
   |           +-[Feature] <Set> (O)
   |           +-[Service-Group-Configuration] <Set> (O)
   |           +-[DPN-Key] <Set>
   |           |           +-[Referenced-Interface] <Set>
   |           |           |       +-[Interface-Key] <L-Key>
   |           |           |       +-[Peer-Service-Group-Key] <Set> (O)
```

Figure 9: Service Group

Each Service-Group element contains the following information:

Service-Group-Key: A unique ID of the Service-Group.

Service-Group-Name: A human-readable display string.

Role: The role (MAG, LMA, etc.) of the device hosting the interfaces of the DPN Group.

Protocol-Set: The set of protocols supported by this interface (e.g., PMIP, S5-GTP, S5-PMIP etc.). The protocol MAY be only its name, e.g. ’gtp’, but many protocols implement specific message sets, e.g. s5-pmip, s8-pmip. When the Service-Group supports specific protocol message sub-subsets the Protocol value MUST include this information.

Feature-Set: An optional set of static features which further determine the suitability of the interface to the desired operation.

Service-Group-Configuration-Set: An optional set of configurations that further determine the suitability of an interface for the specific request. For example: SequenceNumber=ON/OFF.

DPN-Key-Set: A key used to identify the DPN.

Referenced-Interface-Set: The DPN Interfaces and peer Service-Groups associated with them. Each entry contains

> Interface-Key: A key that is used together with the DPN-Key, to create a key that is refers to a specific DPN interface definition.

Peer-Service-Group-Key: Enables location of the peer Service-Group for this Interface.

4.9.3. Domain Information Model

A Domain-Set represents a group of heterogeneous Topology resources typically sharing a common administrative authority. Other models, outside of the scope of this specification, provide the details for the Domain.

```
             |
             +-[Domain] <G-Key>, <Name> (O) <Set>
             |       +-[Domain-Policy-Configuration] (O) <Set>
             |
```

Figure 10: Domain Information Model

Each Domain entry contains the following information:

Domain-Key: Identifies and enables reference to the Domain.

Domain-Name: A human-readable display string naming the Domain.

4.9.4. DPN Information Model

A DPN-Set contains some or all of the DPNs in the Tenant’s network. Some of the DPNs in the Set may be identical in functionality and only differ by their Key.

```
               |
               +-[DPN] <G-Key>, <Name> (O) <Set>
               |     +-[Extensible: FALSE]
               |     +-[Interface] <L-Key> <Set>
               |     |      +-[Role] <U-Key>
               |     |      +-[Protocol] <Set>
               |     |      +-[Interface-Configuration] <Set> (O)
               |     +-[Domain-Key]
               |     +-[Service-Group-Key] <Set> (O)
               |     +-[DPN-Policy-Configuration] <List> (M)
               |     +-[DPN-Resource-Mapping-Reference] (O)
```

Figure 11: DPN Information Model

Each DPN entry contains the following information:

DPN-Key: A unique Identifier of the DPN.

DPN-Name: A human-readable display string.

Domain-Key: A Key providing access to the Domain information about the Domain in which the DPN resides.

Interface-Set: The Interface-Set references all interfaces (through which data packets are received and transmitted) available on the DPN. Each Interface makes use of attribute values that are specific to that interface, for example, the MTU size. These do not affect the DPN selection of active or enabled interfaces. Interfaces contain the following information:

Role: The role (MAG, LMA, PGW, AMF, etc.) of the DPN.

Protocol (Set): The set of protocols supported by this interface (e.g., PMIP, S5-GTP, S5-PMIP etc.). The protocol MAY implement specific message sets, e.g. s5-pmip, s8-pmip. When a protocol implements such message sub-subsets the Protocol value MUST include this information.

Interface-Configuration-Set: Configurable settings that further determine the suitability of an interface for the specific request. For example: SequenceNumber=ON/OFF.

Service-Group-Set: The Service-Group-Set references all of the Service-Groups which have been configured using Interfaces hosted on this DPN. The purpose of a Service-Group is not to describe each interface of each DPN, but rather to indicate interface types for use during the DPN selection process, when a DPN with specific interface capabilities is required.

DPN-Policy-Configuration: A list of Policies that have been configured on this DPN. Some may have values for all attributes, and some may require further configuration. Each Policy-Configuration has a key to enable reference to its Policy-Template. Each Policy-Configuration also has been configured to supply missing and non-default values to the desired Attributes defined within the Policy-Template.

DPN-Resource-Mapping-Reference (O): A reference to the underlying implementation, e.g. physical node, software module, etc. that supports this DPN. Further specification of this attribute is out of scope for this document.

4.9.5. Policy Information Model

The Policy Information Model defines and identifies Rules for enforcement at DPNs. A Policy is basically a set of Rules that are to be applied to each incoming or outgoing packet at a DPN interface. Rules comprise Descriptors and a set of Actions. The Descriptors, when evaluated, determine whether or not a set of Actions will be performed on the packet. The Policy structure is independent of a policy context.

In addition to the Policy structure, the Information Model (per Section 4.9.6) defines Mobility-Context. Each Mobility-Context may be configured with appropriate Attribute values, for example depending on the identity of a mobile node.

Traffic descriptions are defined in Descriptors, and treatments are defined separately in Actions. A Rule-Set binds Descriptors and associated Actions by reference, using Descriptor-Key and Action-Key. A Rule-Set is bound to a policy in the Policy-Set (using Policy-Key), and the Policy references the Rule definitions (using Rule-Key).

```
              |
              +-[Policy Information Model]
              |      +-[Extensible:]
              |      +-[Policy-Template] <G-Key> (M) <Set>
              |      |       +-[Policy-Configuration] <Set> (O)
              |      |       +-[Rule-Template-Key] <List> (M)
              |      |       |       +-[Precedence] (M)
              |      +-[Rule-Template] <L-Key> (M) <Set>
              |      |       +-[Descriptor-Match-Type] (M)
              |      |       +-[Descriptor-Configuration] <Set> (M)
              |      |       |       +-[Direction] (O)
              |      |       +-[Action-Configuration] <Set> (M)
              |      |       |       +-[Action-Order] (M)
              |      |       +-[Rule-Configuration] (O)
              |      +-[Descriptor-Template] <L-Key> (M) <Set>
              |      |       +-[Descriptor-Type] (O)
              |      |       +-[Attribute-Expression] <Set> (M)
              |      +-[Action-Template] <L-Key> (M) <Set>
              |              +-[Action-Type] (O)
              |      |       +-[Attribute-Expression] <Set> (M)
```

Figure 12: Policy Information Model

The Policy structure defines Policy-Set, Rule-Set, Descriptor-Set, and Action-Set, as follows:

Policy-Template: <Set> A set of Policy structures, indexed by Policy-Key, each of which is determined by a list of Rules referenced by their Rule-Key. Each Policy structure contains the following:

Policy-Key: Identifies and enables reference to this Policy definition.

Rule-Template-Key: Enables reference to a Rule template definition.

Rule-Precedence: For each Rule identified by a Rule-Template-Key in the Policy, specifies the order in which that Rule must be applied. The lower the numerical value of Precedence, the higher the rule precedence. Rules with equal precedence MAY be executed in parallel if supported by the DPN. If this value is absent, the rules SHOULD be applied in the order in which they appear in the Policy.

Rule-Template-Set: A set of Rule Template definitions indexed by Rule-Key. Each Rule is defined by a list of Descriptors (located by Descriptor-Key) and a list of Actions (located by Action-Key) as follows:

Rule-Template-Key: Identifies and enables reference to this Rule definition.

Descriptor-Match-Type Indicates whether the evaluation of the Rule proceeds by using conditional-AND, or conditional-OR, on the list of Descriptors.

Descriptor-Configuration: References a Descriptor template definition, along with an expression which names the Attributes for this instantiation from the Descriptor-Template and also specifies whether each Attribute of the Descriptor has a default value or a statically configured value, according to the syntax specified in Section 4.2.

Direction: Indicates if a rule applies to uplink traffic, to downlink traffic, or to both uplink and downlink traffic. Applying a rule to both uplink and downlink traffic, in case of symmetric rules, eliminates the requirement for a separate entry for each direction. When not present, the direction is implied by the Descriptor’s values.

Action-Configuration: References an Action Template definition, along with an expression which names the Attributes for this instantiation from the Action-Template and also specifies whether each Attribute of the Action has a default value or a statically configured value, according to the syntax specified in Section 4.2.

Action-Order: Defines the order in which actions are executed when the associated traffic descriptor selects the packet.

Descriptor-Template-Set: A set of traffic Descriptor Templates, each of which can be evaluated on the incoming or outgoing packet, returning a TRUE or FALSE value, defined as follows:

Descriptor-Template-Key: Identifies and enables reference to this descriptor template definition.

Attribute-Expression: An expression which defines an Attribute in the Descriptor-Template and also specifies whether the Template also defines a default value or a statically configured value for the Attribute of the Descriptor has, according to the syntax specified in Section 4.2.

Descriptor-Type: Identifies the type of descriptor, e.g. an IPv6 traffic selector per [RFC6088].

Action-Template-Set: A set of Action Templates defined as follows:

Action-Template-Key: Identifies and enables reference to this action template definition.

Attribute-Expression: An expression which defines an Attribute in the Action-Template and also specifies whether the Template also defines a default value or a statically configured value for the Attribute of the Action has, according to the syntax specified in Section 4.2.

Action-Type: Identifies the type of an action for unambiguous interpretation of an Action-Value entry.

4.9.6. Mobility-Context Information Model

The Mobility-Context structure holds entries associated with a mobile node and its mobility sessions (flows). It is created on a DPN during the mobile node’s registration to manage the mobile node’s flows. Flow information is added or deleted from the Mobility-Context as needed to support new flows or to deallocate resources for flows that are deactivated. Descriptors are used to characterize the nature and resource requirement for each flow.

Termination of a Mobility-Context implies termination of all flows represented in the Mobility-Context, e.g. after deregistration of a mobile node. If any Child-Contexts are defined, they are also terminated.

```
        +-[Mobility-Context] <G-Key> <Set>
        |            +-[Extensible:˜ FALSE]
        |            +-[Delegating-IP-Prefix:] <Set> (O)
        |            +-[Parent-Context] (O)
        |            +-[Child-Context] <Set> (O)
        |            +-[Service-Group-Key] <Set> (O)
        |            +-[Mobile-Node]
        |            |       +-[IP-Address] <Set> (O))
        |            |       +-[MN-Policy-Configuration] <Set>
        |            +-[Domain-Key]
        |            |       +-[Domain-Policy-Configuration] <Set>
        |            +-[DPN-Key] <Set>
        |            |       +-[Role]
        |            |       +-[DPN-Policy-Configuration] <Set>
        |            |       +-[ServiceDataFlow] <L-Key> <Set> (O)
        |            |       |       +-[Service-Group-Key] (O)
        |            |       |       +-[Interface-Key] <Set>
        |            |       |       +-[ServiceDataFlow-Policy-
                                           Configuration] <Set> (O)
        |            |       |       |       +-[Direction]
```

Figure 13: Mobility-Context Information Model

The Mobility-Context Substructure holds the following entries:

Mobility-Context-Key: Identifies a Mobility-Context

Delegating-IP-Prefix-Set: Delegated IP Prefixes assigned to the Mobility-Context

Parent-Context: If present, a Mobility Context from which the Attributes and Attribute Values of this Mobility Context are inherited.

Child-Context-Set: A set of Mobility Contexts which inherit the Attributes and Attribute Values of this Mobility Context.

Service-Group-Key: Service-Group(s) used during DPN assignment and re-assignment.

Mobile-Node: Attributes specific to the Mobile Node. It contains the following

IP-Address-Set IP addresses assigned to the Mobile Node.

MN-Policy-Configuration-Set For each MN-Policy in the set, a key and relevant information for the Policy Attributes.

Domain-Key: Enables access to a Domain instance.

Domain-Policy-Configuration-Set: For each Domain-Policy in the set, a key and relevant information for the Policy Attributes.

DPN-Key-Set: Enables access to a DPN instance assigned to a specific role, i.e. this is a Set that uses DPN-Key and Role as a compound key to access specific set instances.

Role: Role this DPN fulfills in the Mobility-Context.

DPN-Policy-Configuration-Set: For each DPN-Policy in the set, a key and relevant information for the Policy Attributes.

ServiceDataFlow-Key-Set: Characterizes a traffic flow that has been configured (and provided resources) on the DPN to support data-plane traffic to and from the mobile device.

Service-Group-Key: Enables access to a Service-Group instance.

Interface-Key-Set: Assigns the selected interface of the DPN.

ServiceDataFlow-Policy-Configuration-Set: For each Policy in the set, a key and relevant information for the Policy Attributes.

Direction: Indicates if the reference Policy applies to uplink or downlink traffic, or to both, uplink- and downlink traffic. Applying a rule to both, uplink- and downlink traffic, in case of symmetric rules, allows omitting a separate entry for each direction. When not present the value is assumed to apply to both directions.

4.9.7. Monitor Information Model

Monitors provide a mechanism to produce reports when events occur. A Monitor will have a target that specifies what is to be watched.

The attribute/entity to be monitored places certain constraints on the configuration that can be specified. For example, a Monitor using a Threshold configuration cannot be applied to a Mobility-Context, because it does not have a threshold. Such a monitor configuration could be applied to a numeric threshold property of a Context.

```
                           |
                           +-[Monitor] <G-Key> <List>
                           |          +-[Extensible:]
                           |          +-[Target:]
                           |          +-[Deferrable]
                           |          +-[Configuration]
```

Figure 14: Monitor Substructure

Monitor-Key: Identifies the Monitor.

Target: Description of what is to be monitored. This can be a Service Data Flow, a Policy installed upon a DPN, values of a Mobility-Context, etc. The target name is the absolute information model path (separated by '/') to the attribute / entity to be monitored.

Deferrable: Indicates that a monitoring report can be delayed up to a defined maximum delay, set in the Agent, for possible bundling with other reports.

Configuration: Determined by the Monitor subtype. The monitor report is specified by the Configuration. Four report types are defined:

- "Periodic" reporting specifies an interval by which a notification is sent.

- "Event-List" reporting specifies a list of event types that, if they occur and are related to the monitored attribute, will result in sending a notification.

- "Scheduled" reporting specifies the time (in seconds since Jan 1, 1970) when a notification for the monitor should be sent. Once this Monitor's notification is completed the Monitor is automatically de-registered.

- "Threshold" reporting specifies one or both of a low and high threshold. When these values are crossed a corresponding notification is sent.

5. Protocol

5.1. Protocol Messages and Semantics

Four Client to Agent messages are supported.

Message	Description
Configure	A Configure message includes multiple edits to one or more information model entities. Edits are executed according to their Edit-Id in ascending order. The global status of the operation and the status of individual edits are returned. Partial failures, i.e. individual edit failures, are allowed.
Register-Monitors	Register monitors at an Agent. The message includes the Monitor information as specified in Section 4.9.7.
Deregister-Monitors	Deregister monitors from an Agent. An optional boolean, Send-Data, indicates if a successful deregistration triggers a Notify with final data from the Agent for the corresponding Monitor.
Probe	Probe the status of registered monitors. This triggers a Notify with current data from the Agent for the corresponding Monitors.

Table 1: Client to Agent Messages

Each message contains a header with the following information:

Client Identifier: An Identifier used by the Agent to associate specific configuration characteristics, e.g. options used by the Client when communicating with the Agent, the association of the Client and tenant in the information model as well as tracking operations and notifications.

Delay: An optional time (in ms) to delay the execution of the operation on the DPN once it is received by the Agent.

Operation Identifier: A unique identifier created by the Client to correlate responses and notifications

An Agent will respond with an ERROR, indicating one or more Errors have occurred, or an OK.

For Configure messages, an OK status for an edit MAY include subsequent edits in the response that were required to properly execute the edit. It MAY also indicate that the final status and any final edits required to fulfill the request will be sent via a

Configure Result Notification from the Agent to the Client, see Section 5.1.1.4.2.

If errors occur, they MUST be returned as a list in responses and each Error contains the following information:

Error-type: The specific error type. Values are TRANSPORT (0), RPC (1), PROTOCOL(2) or APPLICATION (3).

Error-Tag: An error tag.

Error-App-Tag: Application specific error tag.

Error-Message: A message describing the error.

Error-Info: Any data required for the response.

```
|
+-[Errors] <List>
|         +-[(Enumeration) Error-Type ]
|         +-[(String) Error-Tag ]
|         +-[(String) Error-App-Tag ] (O)
|         +-[(String) Error-Message ] (O)
|         +-[Error-Info] (O)
```

Figure 15: Error Information Model

Two Agent to Client notifications are supported.

Message	Description
Configure-Result-Notification	An asynchronous notification from Agent to Client based upon a previous Configure request.
Notify	An asynchronous notification from Agent to Client based upon a registered Monitor’s configuration, a Monitor deregistration or Probe.

Table 2: Agent to Client Messages (notifications)

5.1.1. Configure Message

The Configure message follows edit formats proposed by [RFC8072] with more fields in each edit, an extra operation (clone) and a different response format.

5.1.1.1. Edit Operation Types

Operation	Description
create	Creates a new data resource or Entity. If the resource exists an error is returned.
delete	Deletes a resource. If it does not exist an error is returned.
insert	Inserts data in a list or user ordered list.
merge	Merges the edit value with the target data resource; the resource is created if it does not exist.
move	Moves the target data resource.
replace	Replace the target data resource with the edit value.
remove	Removes a data resource if it already exists.
clone	Clones a data resource and places the copy at the new location. If the resource does not exist an error is returned.

Table 3: Configure Edit Operations

5.1.1.2. Edit Operation

Each Configure includes one or more edits. These edits include the following information:

Edit-Id: Uniquely specifies the identifier of the edit within the operation.

Edit-Type: Specifies the type of operation (see Section 5.1.1.1).

Command-Set: The Command-Set is a technology-specific bitset that allows for a single entity to be sent in an edit with multiple requested, technology specific sub-transactions to be completed. It can also provide clarity for a request. For example, a Mobility-Context could have the Home Network Prefix absent but it is unclear if the Client would like the address to be assigned by the Agent or if this is an error. Rather than creating a specific command for assigning the IP, a bit position in a Command-Set can be used to indicate Agent based IP assignment requests.

Reference-Scope: If supported, specifies the Reference Scope (see Section 5.1.1.3)

Target: Specifies the Target node (Data node path or FPC Identity) for the edit operation. This MAY be a resource, e.g. Mobility-Context, Descriptor-Template, etc., or a data node within a resource as specified by its path.

Point: The absolute URL path for the data node that is being used as the insertion point, clone point or move point for the target of this ’edit’ entry.

Where: Identifies where a data resource will be inserted, cloned to or moved. Only allowed these for lists and lists of data nodes that are ’ordered-by user’. The values are ’before’, ’after’, ’first’, ’last’ (default value).

Value The value used for this edit operation. In this message it MUST NOT be a MONITOR entity.

```
|
+-[Configure]
|         +-[Client-Id:]
|         +-[(Unsigned 32) Execution-Delay]
|         +-[Operation-Id:]
|         +-[Edit:] <List>
|         |    +-[Edit-Id:] <L-Key>
|         |    +-[(Enumeration) Edit-Type:]
|         |    +-[(BitSet) Command-Set]
|         |    +-[(Enumeration) Reference-Scope]
|         |    +-[Target:]
|         |    +-[Point]
|         |    +-[(Enumeration) Where]
|         |    +-[Value]
```

Figure 16: Configure Request

Edits sent to the Agent provided in an operation SHOULD be sent in the following order to avoid errors:

1. Action Templates

2. Descriptor Templates

3. Rule Templates

4. Policy Templates

5. DPN Templates

6. Mobility Contexts

5.1.1.3. Reference Scope

The Reference Scope is an optional feature that provides the scope of references used in a configuration command. These scopes are defined as:

- none - All edits have no references to other entities or within edits.

- edit - All references are contained within each edit body (intra-edit/intra-operation)

- operation - All references exist in the operation (inter-edit/ intra-operation).

- storage - One or more references exist outside of the operation. A lookup to cache / storage is required.

- unknown - the location of the references are unknown. This is treated as a ’storage’ type.

An Agent that only accepts ’edit’ or ’operation’ reference scope messages is referred to as ’stateless’ as it has no direct memory of references outside messages themselves. This permits low memory footprint Agents/DPNs. Even when an Agent supports all message types an ’edit’ or ’operation’ scoped message can be processed quickly by the Agent/DPN as it does not require storage access.

Figure 17 shows an example containment hierarchy provided for all caches.

```
                         +---------------+
                         | Global Cache  |
                         |  (storage)    |
                         +------+--------+
                                |
                                +----------------+
                                |                |
                +------+-----------+      +------+-----------+
                | Operation Cache  |      | Operation Cache  |
                |   (operation)    | .... |   (operation)    |
                +------+-----------+      +--------+---------+
                       |                           |
                   +---+-----------+               |
                   |               |               |
            +------+------+ +------+------+ +------+------+
            | Edit Cache  | | Edit Cache  | | Edit Cache  |
            |   (edit)    | |   (edit)    | |   (edit)    |
            +-------------+ +-------------+ +-------------+

                                (no cache)
```

Figure 17: Example Hierarchical Cache

5.1.1.4. Operation Response

5.1.1.4.1. Immediate Response

The Response MUST include the following:

Operation Identifier of the corresponding request.

Global Status for the operation (see Table 1).

A list of Edit results (described below).

An edit response, Edit-Status, is comprised of the following:

Edit-Id: Edit Identifier.

Edit-Status: OK.

When the Edit-Status is OK the following values MAY be present

Notify-Follows - A boolean indicator that the edit has been accepted by the Agent but further processing is required. A Configure-Result-Notification will be sent once the processing has succeeded or failed.

Subsequent-Edits-List: This is a list of Edits that were required to fulfill the request. It follows the edit request semantics (see Section 5.1.1.2).

Errors-List: When the Edit-Status is ERROR the following values are present. See Table 1 for details.

The response will minimally contain an Edit-Status implying ’OK’ or a list of errors.

```
             |
             +-[Operation-Id:]
             +-[Result-Status:]
             +-[Errors] <List>
             |         +-[(Enumeration) Error-Type:]
             |         +-[(String) Error-Tag:]
             |         +-[(String) Error-App-Tag]
             |         +-[(String) Error-Message]
             |         +-[Error-Info]
             +-[Edit-Status]
             |         +-[Edit-Id:]
             |         +-[Edit-Status: ~ OK]
             |         +-[Notify-Follows]
             |         +-[Subsequent-Edits] <List>
             |         |    +-[Edit-Id:] <L-Key>
             |         |    +-[(Enumeration) Edit-Type:]
             |         |    +-[Target:]
             |         |    +-[Point]
             |         |    +-[(Enumeration) Where]
             |         |    +-[Value]
             |         +-[Errors] <List>
             |         |         +-[(Enumeration) Error-Type:]
             |         |         +-[(String) Error-Tag:]
             |         |         +-[(String) Error-App-Tag]
             |         |         +-[(String) Error-Message]
             |         |         +-[Error-Info]
             |
```

Figure 18: Configure Operation Response

5.1.1.4.2. Asynchronous Notification

A Configure-Result-Notification occurs after the Agent has completed processing related to a Configure request. It is an asynchronous communication from the Agent to the Client.

It is identical to the immediate response with the exception that the Notify-Follows, if present, MUST be false. As this value is unnecessary it SHOULD be omitted.

5.1.1.5. Reserved Identities

Several identities are reserved in the Policy Information Model and Mobility-Context to facilitate specific uses cases.

Agents and tenants express their support for descriptors and actions using the following Key patterns

supported-<descriptor template name> indicates a support for the descriptor template as defined in its original specification. For example "supported-rfc5777classifier" is a Descriptor Template that conforms to the rfc5777-classifier (Figure 31) as defined in this document.

supported-<action template name> indicates a support for the action template as defined in its original specification.

"base-rule" is comprised of all base descriptors using an ’or’ Descriptor-Match-Type and all Actions in no specific order.

"base-template" is comprised of the base rule.

"base-template" can be used to determine supported Action and Descriptor Templates. It can also be used to support an open template where any specific Descriptors and Actions can be applied, however, depending upon the Order of Actions it is likely to produce undesirable results.

One use case is supported via reservation of specific DPN-Keys:

Requested policies are those that the Client would like to be assigned to a DPN within a Mobility-Context. The naming convention is similar to those used for DPN Assignment via an Agent.

"Requested" is a Key that represents requested policies which have not been assigned to a specific DPN. No Role is assigned to the DPN.

"Requested-<Role>" represents requested policies that have not been assigned to a DPN and can only be assigned to DPNs that fulfill the specified Role.

It is possible to have policies in the "Requested" DPN that do not appear in other entries which reflects the inability to successfully assign the policy.

5.1.2. Monitor Messages

An Agent may reject a registration if it or the DPN has insufficient resources.

An Agent or DPN MAY temporarily suspend monitoring if insufficient resources exist. In such a case the Agent MUST notify the Client.

When a monitor has a reporting configuration of SCHEDULED it is automatically de-registered after the last Notify occurs.

If a SCHEDULED or PERIODIC configuration is provided during registration with the time related value (time or period respectively) of 0 a Notify is sent and the monitor is immediately de-registered. This method should, when a Monitor has not been installed, result in an immediate Notify sufficient for the Client’s needs and lets the Agent realize the Client has no further need for the monitor to be registered.

Probe messages are used by a Client to retrieve information about a previously installed monitor. The Probe message SHOULD identify one or more monitors by means of including the associated monitor identifier. An Agent receiving a Probe message sends the requested information in a single or multiple Notify messages.

If the Monitor configuration associated with a Notify can be deferred, then the Notify MAY be bundled with other messages back to the Agent even if this results in a delay of the Notify.

The Monitor messages use the following data:

Monitor-Key: Monitor Key.

Monitor: A Monitor configuration (see Section 4.9.7).

Send-Data: An indicator that specifies that the final value MUST be sent as a notification from the Agent.

```
             |
             +-[Register-Monitor]
             |         +-[Client-Id:]
             |         +-[(Unsigned 32) Execution-Delay]
             |         +-[Operation-Id:]
             |         +-[Monitor:] <List>
             |         |         +-[Extensible:]
             |         |         +-[Monitor-Key:] <U-Key>
             |         |         +-[Target:]
             |         |         +-[Deferrable]
             |         |         +-[Configuration:]

             |
             +-[Deregister-Monitor]
             |         +-[Client-Id:]
             |         +-[(Unsigned 32) Execution-Delay]
             |         +-[Operation-Id:]
             |         +-[Monitor:] <List>
             |         |         +-[Monitor-Key:] <U-Key>
             |         |         +-[(Boolean) Send-Data ~ False]

             |
             +-[Probe]
             |         +-[Client-Id:]
             |         +-[(Unsigned 32) Execution-Delay]
             |         +-[Operation-Id:]
             |         +-[Monitor-Key:] <List>
```

Figure 19: Monitor Messages

5.1.2.1. Asynchronous Notification

A Monitor Report can be sent as part of de-registration, a trigger based upon a Monitor Configuration or a Probe. A Report is comprised of the Monitor Key the report applies to, the Trigger for the report, a timestamp of when the report’s associated event occurs and data, Report-Value, that is specific to the monitored value’s type.

Triggers include but are not limited to

- Subscribed Event occurred

- Low Threshold Crossed

- High Threshold Crossed

- Periodic Report

- Scheduled Report

- Probe

- Deregistration Final Value

- Monitoring Suspended

- Monitoring Resumed

- DPN Available

- DPN Unavailable

Multiple Reports are sent in a Notify message. Each Notify is comprised of unique Notification Identifier from the Agent and timestamp indicating when the notification was created.

```
           |
           +-[ Notify ]
           |         +-[(Unsigned 32) Notification-Identifier:]
           |         +-[Timestamp:]
           |         +-[Report:] <List>
           |         |    +-[Trigger:]
           |         |    +-[Monitor-Key:]
           |         |    +-[Report-Value]
```

Figure 20: Monitor Messages

5.2. Protocol Operation

Please note that JSON is used to represent the information in Figures in this section but any over the wire representation that accurately reflects the information model MAY be used.

5.2.1. DPN Selection

In order to assign a DPN to a Mobility Context, the Client or Agent requires topology information. The Service-Group provides information, e.g. function, role, protocol, features and configuration, to determine suitable DPN interfaces.

Consider a Client attempting to select DPN interfaces that are served by a single Agent. In this example interfaces are present with different protocols, settings and features as shown in the following figure.

```
    "topology-information-model" : {
     "dpn" : [ {
      "dpn-key" : "dpn1",
      "interface" : [ {
        "interface-key" : "ifc1",
        "role" : "lma",
        "protocol" : [ "pmip" ],
        "interface-configuration" : [ {
            "index" : 0,
            "setting" : [ "optionA" : "OFF" ]
        } ]
      },{
        "interface-key" : "ifc2",
        "role" : "lma",
        "protocol" : [ "pmip" ],
        "interface-configuration" : [ {
            "index" : 0,
            "setting" : [ "optionC" : "OFF"  ]
         } ]
       },{
        "interface-key" : "ifc2-b",
        "role" : "mag",
        "protocol" : [ "pmip" ]
        } ] },
      {
      "dpn-key" : "dpn2",
      "interface" : [ {
        "interface-key" : "ifc1",
        "role" : "mag",
        "protocol" : [ "pmip" ],
        "interface-configuration" : [ {
          "index" : 0,
          "settings" : [ "optionA" : "OFF", "optionB" : "ON" ]
        } ]
      } ] }
     ],
     ...
     },
     "service-group" : [
     { "service-group-key" : "group1",
       "service-group-name" : "Anchors-OptionA-OFF",
       "role-key" : "lma",
       "protocol" : [ "pmip" ],
       "service-group-configuration" : [ {
            "index" : 0,
            "setting" : [ "optionA" : "OFF" ]
       } ],
       "dpn" : [
       { "dpn-key" : "dpn1",
         "referenced-interface" : [ { "interface-key" : "ifc1" } ] }
       ]
     },{ "service-group-key" : "group2",
       "service-group-name" : "Anchors",
       "role-role" : "lma",
       "protocol" : [ "pmip" ],
       "dpn" : [
       { "dpn-key" : "dpn1",
         "referenced-interface" : [ { "interface-key" : "ifc2" } ] }
       ]
      },{ "service-group-key" : "group3",
       "service-group-name" : "MAGs",
       "role-role" : "mag",
       "protocol" : [ "pmip" ],
       "dpn" : [
       { "dpn-key" : "dpn2",
         "referenced-interface" : [ { "interface-key" : "ifc1" } ] },
       { "dpn-key" : "dpn1",
         "referenced-interface" : [ { "interface-key" : "ifc2-b" } ] }
       ]
      }
    ]
```

NOTE - A Setting is, in this example, a list of string attributes in a Configuration.

Figure 21: Monitor Messages

Two DPNs are present. The first, dpn1, has 3 interfaces. Two support the LMA role and both have settings. The third supports the MAG function. The second DPN, dpn2, provides a single interface with the MAG function.

Three ServiceGroups are presented. The first provides the PMIP protocol and LMA role. It also has a setting, OptionA, that is OFF and only contains ifc1 from dpn1.

The second group is comprised of interfaces that support the PMIP protocol and LMA function. It only contains ifc2 from dpn1. An interface that has setting(s) or feature(s) that must appear in a ServiceGroup SHOULD NOT appear in ServiceGroups that do not have those setting(s) or feature(s) present. Thus, ifc1 of dpn1 should not be present in this second Service-Group.

A third group is comprised of interfaces that support the MAG function of the LMA protocol. It contains the MAG interfaces form both dpn1 and dpn2.

Given the task to find a LMA that supports the PMIP protocol the Client can determine that dpn1 is its only option and, depending on its requirement of OptionA, can appropriately determine which interface to select.

5.2.2. Policy Creation and Installation

A Policy must be installed upon an Agent in order to install policies on the selected dpn(s). This requires construction of the Action(s), Descriptor(s) and Rule(s) used by the Policy.

The CONFIGURE message permits editing all information elements except monitors. The following figure shows use of a CONFIGURE message to install policy information on the Agent.

```
                                                 +-------Router--------+
                            +-----------+        |+-------+ +---------+|
    +------+ +------+     +-----+ FPC   |         | FPC   | |  Anchor |
    |MAG-C1| |MAG-C2|     |LMA-C| Client|         | Agent | |   DPN   |
    +------+ +------+     +-----+-------+         +-------+ +---------+
    |         |            |---(1)--Configure-------->|           |
    |         |          "configure" : {              |           |
    |         |           "client-id" : 0,            |           |
    |         |           "operation-id" : 0,         |           |
    |         |           "edit" : [ {                |           |
    |         |            "edit-id" : 0,             |           |
    |         |            "edit-type" : "create",    |           |
    |         |            "target" : "/policy-information-model/
                              /descriptor-template",              |
    |         |            "value" : {                            |
    |         |              "descriptor-template-key" : "desc1", |
    |         |              "descriptor-type" : "all-traffic" }  |
    |         |             }, {                                  |
    |         |            "edit-id" : 1,             |           |
    |         |            "edit-type" : "create",    |           |
    |         |            "target" : "/policy-information-model/
                              /action-template",                  |
    |         |            "value" : {                            |
    |         |              "action-template-key" : "action1",   |
    |         |              "action-type" : "drop" }             |
    |         |             }, {                                  |
    |         |            "edit-id" : 2,             |           |
    |         |            "edit-type" : "create",    |           |
    |         |            "target" : "/policy-information-model/
                              /rule-template",                    |
    |         |            "value" : {                            |
    |         |              "rule-template-key" : "deny-all",    |
    |         |              "descriptor-match-type" : "and",     |
    |         |              "descriptor-configuration" : [{      |
    |         |              "descriptor-template-key" : "all" }],|
    |         |              "action-configuration" : [{          |
    |         |                "action-template-key" : "deny",    |
    |         |                "action-order" : 0 }]              |
    |         |             }, {                                  |
    |         |            "edit-id" : 3,            |            |
    |         |            "edit-type" : "create",   |            |
    |         |            "target" : "/policy-information-model/
                              /policy-template",                  |
    |         |            "value" : {                            |
    |         |              "policy-template-key" : "policy1",   |
    |         |              "entity-state" : "configured",       |
    |         |              "rule-template" : [ {                |
    |         |              "rule-template-key" : "deny-all",    |
    |         |              "precedence" : 0 } ]                 |
    |         |             } } ] }                               |
    |         |            |<---(2)- Response --------|           |
    |         |            | {                        |           |
    |         |            |  "agent-id" : "agent1"," |           |
    |         |            |  "operation-id" : 0,     |           |
    |         |            |  "result-status" : "ok"  |           |
    |         |            | }                        |           |
    |         |            |                          |           |
```

Figure 22: Example Policy Installation (focus on FPC reference point)

In this example a Descriptor "all-traffic" Template and an Action, "drop", Template are both empty Templates. The "deny-all" Rule Template is comprised of the action and descriptor. The Rule is included in "policy1". The policy’s status is "Configured" as it is a complete policy ready for immediate use. The policy could be set as "Active" if the Client intends to use it upon immediate installation in a DPN.

Installation of the policy on dpn1 is shown in the following Figure. The Policy-Status is set to "Active" to make it immediately usable. Leaving the status as Configured would permit its installation on the DPN without an ability to use it in a Mobility Context. Such a use case is often referred to as policy pre-configuration.

```
                                                 +-------Router--------+
                            +-----------+        |+-------+ +---------+|
    +------+ +------+     +-----+ FPC   |         | FPC   | |  Anchor |
    |MAG-C1| |MAG-C2|     |LMA-C| Client|         | Agent | |   DPN   |
    +------+ +------+     +-----+-------+         +-------+ +---------+
    |         |            |---(1)--Configure-------->|           |
    |         |          "configure" : {              |           |
    |         |           "client-id" : 0,            |           |
    |         |           "operation-id" : 1,         |           |
    |         |           "edit" : [ {                |           |
    |         |            "edit-id" : 0,             |           |
    |         |            "edit-type" : "create",    |           |
    |         |            "target" : "/topology-information-model/
                             /dpn/dpn1/dpn-policy-configuration", |
    |         |            "value" : {                            |
    |         |              "policy-template-key" : "policy1",   |
    |         |              "policy-status" : "active" } } ] }   |
    |         |            |<---(2)- Response --------|-policy--->|
    |         |            | {                        | install   |
    |         |            |  "agent-id" : "agent1"," |           |
    |         |            |  "operation-id" : 1,     |           |
    |         |            |  "result-status" : "ok"  |           |
    |         |            | }                        |           |
    |         |            |                          |           |
```

Figure 23: Example Policy Installation (focus on FPC reference point)

This message uses an edit type of "create" to add the policy template directly to the installed DPN policy set.

5.2.3. Simple RPC Operation

A Client and Agent MUST identify themselves using the Client Identifier and Agent Identifier respectively to ensure that, for all transactions, a recipient of a FPC message can unambiguously identify the sender of the FPC message.

A Client MAY direct the Agent to enforce a rule in a particular DPN by including a DPN Key value in a Mobility Context. Otherwise the Agent selects a suitable DPN to enforce one or more portions of a Mobility Context and notifies the Client about the selected DPN(s) using DPN Identifier(s).

All messages sent from a Client to an Agent MUST be acknowledged by the Agent. The response must include all edit status as well as subsequent edits, which indicates the result of processing the message, as part of the Configure response. In case the processing of the message results in a failure, the Agent sets the global status, Error-Type and Error-Tag accordingly and MAY clear the entity, e.g. Mobility-Context, which caused the failure, in the response.

If based upon Agent configuration or the processing of the request possibly taking a significant amount of time the Agent MAY respond with a Notify-Follows indication with optional Subsequent-Edit(s) containing the partially completed entity modifications. When a Notify-Follows indication is sent in a response, the Agent will, upon completion or failure of the operation, respond with an asynchronous Configuration-Result-Notification to the Client.

A Client MAY add a property to a Mobility-Context without providing all required details of the attribute’s value. In such case the Agent SHOULD determine the missing details and provide the completed property description, via Subsequent-Edit(s), back to the Client. If the processing will take too long or based upon Agent configuration, the Agent MAY respond with an OK for the Edit that indicates a Notify-Follows and also includes Subsequent-Edit(s) containing the partially completed entity edits.

In case the Agent cannot determine the missing value of an attribute’s value per the Client’s request, it leaves the attribute’s value cleared, sets the Edit Result to Error and provides an Error-Type and Error-Tag. As example, the Control-Plane needs to setup a tunnel configuration in the Data-Plane but has to rely on the Agent to determine the tunnel endpoint which is associated with the DPN that supports the Mobility-Context. The Client adds the tunnel property attribute to the FPC message and clears the value of the attribute (e.g. IP address of the local tunnel endpoint). The Agent determines the tunnel endpoint and includes the completed tunnel property in its response to the Client in a Subsequent-Edit entry.

Figure 24 illustrates an exemplary session life-cycle based on Proxy Mobile IPv6 registration via MAG Control-Plane function 1 (MAG-C1) and handover to MAG Control-Plane function 2 (MAG-C2). Edge DPN1 represents the Proxy CoA after attachment, whereas Edge DPN2 serves as Proxy CoA after handover. As exemplary architecture, the FPC Agent and the network control function are assumed to be co-located with the Anchor-DPN, e.g. a Router.

The Target of the second request uses the Mobility-Context by name. Alternatively, the Target could have included the DPN-Key and Policy-Key to further reduce the amount of information exchanged. Setting the Target’s value to the most specific node SHOULD be followed whenever practical.

```
                                         +-------Router--------+
                           +-----------+        |+-------+ +---------+|
   +------+ +------+     +-----+ FPC   |         | FPC   | |  Anchor |
   |MAG-C1| |MAG-C2|     |LMA-C| Client|         | Agent | |   DPN1  |
   +------+ +------+     +-----+-------+         +-------+ +---------+
   [MN attach]  |            |                          |           |
      |-------------PBU----->|                          |           |
      |         |            |---(1)--Configure-------->|           |
      |         |  "configure" : {                      |           |
      |         |    "client-id" : 0,                   |           |
      |         |    "operation-id" : 3,                |           |
      |         |    "edit" : [                         |           |
      |         |      "edit-id" : 0,                   |           |
      |         |      "edit-type" : "create",          |           |
      |         |      "target" : "/mobility-context",              |
      |         |      "value" : {                                  |
      |         |        "mobility-context-key" : "ctxt1",          |
      |         |        "delegating-ip-prefix" : [ <HNP> ],        |
      |         |        "dpn" : [ {                                |
      |         |          "dpn-key" : "DPN1",                      |
      |         |          "role" : "lma",                          |
      |         |          "service-data-flow" : [ {
      |         |            "identifier" : 0,
      |         |            "interface" : [ "interface-key" : "ifc1" ],
      |         |            "service-data-flow-policy-configuration" :[
      |         |              {"policy-template-key" :
                                   "dl-tunnel-with-qos",
      |         |                "policy-status" : "active",
      |         |                "policy-configuration" : [
      |         |                  {"index" : 0,
      |         |                   "qos-template" : <QOS Settings...>},
      |         |                  {"index" : 1,
      |         |                   "tunnel" : <DL tunnel info...>},
      |         |               {"policy-template-key" : "ul-tunnel",
      |         |                   "policy-status" : "active",
      |         |                "policy-configuration" : [
      |         |                  {"index" : 1,
      |         |                   "tunnel" : <UL tunnel info...>}] }]
      |         |           } ] } ] } ] } ] }           |           |
      |         |            |                          |--tun1 up->|
      |         |            |                          |           |
      |         |            |                          |--tc qos-->|
      |         |            |                          |           |
      |         |            |<---(2)- Response --------|-route add>|
      |         |            | {                        |           |
      |         |            |  "agent-id" : "agent1"," |           |
      |         |            |  "operation-id" : 3,     |           |
      |         |            |  "result-status" : "ok", |           |
      |         |            |  }                       |           |
      |         |            |                          |           |
      |<------------PBA------|                          |           |
      |         |            |                          |           |
      | +----+  |            |                          |           |
      | |Edge|  |            |                          |           |
      | |DPN1|  |            |                          |           |
      | +----+  |            |                          |           |
      |   |                                                         |
      |   |-=======================================================-|
      |                      |                          |           |
      |   [MN handover]      |                          |           |
      |         |---PBU ---->|                          |           |
      |         |            |--(3)- CONFIG(MODIFY)---->|           |
      |         |  "configure" : {                      |-tun1 mod->|
      |         |    "client-id" : 0,                   |           |
      |         |    "operation-id" : 4,                |           |
      |         |    "edit" : [                         |           |
      |         |      "edit-id" : 0,                   |           |
      |         |      "edit-type" : "merge",           |           |
      |         |      "target" : "/mobility-context/ctxt1",        |
      |         |      "value" : {                                  |
      |         |        "mobility-context-key" : "ctxt1",          |
      |         |        "dpn" : "[ {                               |
      |         |          "dpn-key" : "DPN1",                      |
      |         |          "service-data-flow" : [ {
      |         |            "identifier" : 0,
      |         |            "service-data-flow-policy-configuration":[
      |         |                {"policy-template-key" :
                                        "dl-tunnel-with-qos",
      |         |                "policy-configuration" : [
      |         |                  {"index" : 1,
      |         |                   "tunnel" : <NEW tunnel info...>}]}]
      |         |           } ] } ] } ] }               |           |
      |         |<--PBA------|                          |           |
      |         |            |                          |-tun1 mod->|
      |         |            |<---(4)- OK --------------|           |
      |         |            | {                        |           |
      |         |            |  "agent-id" : "agent1"," |           |
      |         |            |  "operation-id" : 4,     |           |
      |         |            |  "result-status" : "ok", |           |
      |         |            |  }                       |           |
      |         |  +----+    |                          |           |
      |         |  |Edge|    |                          |           |
      |         |  |DPN2|    |                          |           |
      |         |  +----+    |                          |           |
      |         |    |       |                          |           |
      |         |    |-============================================-|
      |         |            |                          |           |
```

Figure 24: Single Agent with Handover (focus on FPC reference point)

After reception of the Proxy Binding Update (PBU) at the LMA Control-Plane function (LMA-C), the LMA-C selects a suitable DPN, which serves as Data-Plane anchor to the mobile node’s (MN) traffic. The LMA-C adds a new logical Mobility-Context to the DPN to treat the MN’s traffic (1) and includes a Mobility-Context-Key (ctxt1) in the Configure command. The LMA-C identifies the selected Anchor DPN by including the associated DPN identifier.

The LMA-C adds policy template properties during the creation of the new Mobility-Context. One policy, "dl-tunnel-with-qos", is an example template that permits tunnel forwarding of traffic destined to the MN’s HNP, i.e. downlink traffic, with optional QoS parameters. Another policy, "ul-tunnel", provides a simple uplink anchor termination template where uplink tunnel information is provided.

The downlink tunnel information specifies the destination endpoint (Edge DPN1).

Upon reception of the Mobility-Context, the FPC Agent utilizes local configuration commands to create the tunnel (tun1) as well as the traffic control (tc) to enable QoS differentiation. After configuration has been completed, the Agent applies a new route to forward all traffic destined to the MN’s HNP specified as a property in the Mobility-Context and applied the configured tunnel interface (tun1).

During handover, the LMA-C receives an updating PBU from the handover target MAG-C2. The PBU refers to a new Data-Plane node (Edge DPN2) to represent the new tunnel endpoint in the downlink as required. The LMA-C sends a Configure message (3) to the Agent to modify the existing tunnel property of the existing Mobility-Context and to update the downlink tunnel endpoint from Edge DPN1 to Edge DPN2. Upon reception of the Configure message, the Agent applies updated tunnel property to the local configuration and responds to the Client (4).

```
                                                 +-------Router--------+
                           +-----------+         |+-------+ +---------+|
   +------+ +------+     +-----+ FPC   |          | FPC   | |  Anchor |
   |MAG-C1| |MAG-C2|     |LMA-C| Client|          | Agent | |   DPN   |
   +------+ +------+     +-----+-------+          +-------+ +---------+
   [MN detach]  |            |                           |           |
      |-------------PBU----->|                           |           |
      |         |            |---(1)--Configure--------->|           |
      |         |          "configure" : {               |           |
      |         |           "client-identifier" : 0,    |           |
      |         |           "operation-id" : 5,         |           |
      |         |           "edits" : [                 |           |
      |         |            "edit-id" : 0,             |           |
      |         |            "edit-type" : "merge",     |           |
      |         |            "target" : "/mobility-context/ctxt1    |
      |         |                /dpn/DPN1/service-data-flow/0      |
      |         |                /service-data-flow-policy-
                                  configuration/dl-tunnel-with-qos/1"
      |         |            "value" : {                |           |
      |         |                  "tunnel" : null      |           |
      |         |             } ] }                     |           |
      |<------------PBA------|                          |--tun1   ->|
      |         |            |                          |    down   |
      |         |            |                          |           |
      |         |            |<---(2)- Response --------|           |
      |         |            | {                        |           |
      |         |            |  "agent-id" : "agent1"," |           |
      |         |            |  "operation-id" : 5,     |           |
      |         |            |  "result-status" : "ok", |           |
      |         |            | }                        |           |
      |         |            |                          |           |
      |         |  [ MinDelayBeforeBCEDelete expires ]  |           |
      |         |            |                          |           |
      |         |            |---(3)--Configure-------->|-- tun1 -->|
      |         |          "configure" : {              |  delete   |
      |         |           "client-identifier" : 0,    |           |
      |         |           "operation-id" : 6,         |           |
      |         |           "edits" : [                 |           |
      |         |            "edit-id" : 0,             |           |
      |         |            "edit-type" : "delete",    |           |
      |         |            "target" : "/mobility-context/ctxt1"   |
      |         |            ] }                        |           |
      |         |            |                          |           |
      |         |            |<---(4)- Response --------|           |
      |         |            | {                        |           |
      |         |            |  "agent-id" : "agent1"," |           |
      |         |            |  "operation-id" : 6,     |           |
      |         |            |  "result-status" : "ok", |           |
      |         |            | }                        |           |
      |         |            |                          |-- route ->|
      |         |            |                          |   remove  |
      |         |            |                          |           |
```

Figure 25: Single Agent with Deletion (focus on FPC reference point)

When a teardown of the session occurs, MAG-C1 will send a PBU with a lifetime value of zero. The LMA-C sends a Configure message (1) to the Agent to modify the existing tunnel property of the existing

Mobility-Context to delete the tunnel information. Upon reception of the Configure message, the Agent removes the tunnel configuration and responds to the Client (2). Per [RFC5213], the PBA is sent back immediately after the PBA is received.

If no valid PBA is received after the expiration of the MinDelayBeforeBCEDelete timer (see [RFC5213]), the LMA-C will send a Configure (3) message with a deletion request for the Context. Upon reception of the message, the Agent deletes the tunnel and route on the DPN and responds to the Client (4).

When a multi-DPN Agent is used the DPN list permits several DPNs to be provisioned in a single message for the single Mobility-Context.

```
                           +-----------+           +-------+ +---------+
   +------+ +------+     +-----+ FPC   |           | FPC   | |  Anchor |
   |MAG-C1| |MAG-C2|     |LMA-C| Client|           | Agent | |   DPN1  |
   +------+ +------+     +-----+-------+           +-------+ +---------+
   [MN attach]  |            |                          |           |
      |-------------PBU----->|                          |           |
      |         |            |---(1)--Configure-------->|           |
      |         |  "configure" : {                      |--tun1 up->|
      |         |    "client-id" : 0,                   |           |
      |         |    "operation-id" : 0,                |           |
      |         |    "edit" : [                         |--tc qos-->|
      |         |      "edit-id" : 0,                   |           |
      |         |      "edit-type" : "create",          |           |
      |         |      "target" : "/mobility-context",              |
      |         |      "value" : {                                  |
      |         |        "mobility-context-key" : "ctxt1",          |
      |         |        "delegating-ip-prefix" : [ <HNP> ],        |
      |         |        "dpn" : [ {                                |
      |         |          "role" : "lma",                          |
      |         |          "dpn-key" : "DPN1",                      |
      |         |          "service-data-flow" : [ {
      |         |            "identifier" : 0,
      |         |            "interface" : [ "interface-key" : "ifc1" ],
      |         |            "service-data-flow-policy-configuration" :[
      |         |              {"policy-template-key" :
                                        "dl-tunnel-with-qos",
      |         |                "policy-status" : "active",
      |         |                "policy-configuration" : [
      |         |                  {"index" : 0,
      |         |                   "qos-template" : <QOS Settings...>},
      |         |                  {"index" : 1,
      |         |                   "tunnel" : <DL tunnel info...>},
      |         |               {"policy-template-key" : "ul-tunnel",
      |         |                   "policy-status" : "active",
      |         |                "policy-configuration" : [
      |         |                  {"index" : 1,
      |         |                   "tunnel" : <UL tunnel info...>}] }]
      |         |              } ] } ] }, {
      |         |          "dpn-key" : "DPN2",                         |
      |         |          "role" : "mag",                             |
      |         |          "service-data-flow" : [ {
      |         |            "identifier" : 0,
      |         |            "interface" : [ "interface-key" : "ifc2" ],
      |         |            "service-data-flow-policy-configuration" :[
      |         |              {"policy-template-key" :
      "dl-tunnel-with-qos",
      |         |                "policy-status" : "active",
      |         |                "policy-configuration" : [
      |         |                  {"index" : 0,
      |         |                   "qos-template" : <QOS Settings...>},
      |         |                  {"index" : 1,
      |         |                   "tunnel" : <DL tunnel info...>},
      |         |               {"policy-template-key" : "ul-tunnel",
      |         |                   "policy-status" : "active",
      |         |                "policy-configuration" : [
      |         |                  {"index" : 1,
      |         |                   "tunnel" : <UL tunnel info...>}] }]
      |         |              } ] } ] }                |           |
      |         |              ] } ] }                  |           |
      |         |            |                          |           |
      |         |            |<---(2)- Response --------|           |
      |         |            | {                        |-route add>|
      |         |            |  "agent-id" : "agent1"," |           |
      |         |            |  "operation-id" : 0,     |           |
      |         |            |  "result-status" : "ok", |           |
      |         |            |  "notify-follows" : "true",          |
      |         |            | }                        |           |
      |         |            |                          |           |
      |<------------PBA------|                          |           |
      |         |            |                          |           |
      | +----+               |                          |           |
      | |Edge|               |                          |           |
      | |DPN2|               |                          |           |
      | +----+               |                          |           |
      |   |<---------------------- tun1 up -------------|           |
      |   |<---------------------- tc qos --------------|           |
      |   |<---------------------- route add -----------|           |
      |   |                  |                          |           |
      |         |            |<(3) Configure-Result-    |           |
      |         |            |       Notification       |           |
      |         |            | {                        |-route add>|
      |         |            |  "agent-id" : "agent1"," |           |
      |         |            |  "operation-id" : 3,     |           |
      |         |            |  "result-status" : "ok", |           |
      |         |            |  "notify-follows" : "true",          |
      |         |            |  "edit-status" : [       |           |
      |         |            |    "edit-id" : 0,        |           |
      |         |            |    "edit-status" : "ok"  |           |
      |         |            | ] }                      |           |
      |         |            |                          |           |
      |   |                  |                          |           |
```

Figure 26: Exemplary Message Sequence for Multi-DPN Agent

Figure 26 shows how the first 2 messages in Figure 24 are supported when a multi-DPN Agent communicates with both Anchor DPN1 and Edge DPN2. In such a case, the FPC Client sends the downlink and uplink for both DPNs in the DPN Reference List of the same Mobility-Context. Message 1 shows the DPN Set with all entries. Each entry identifies the DPN.

The Agent responds with an OK and Notify-Follows indication while it simultaneously provisions both DPNs. Upon successful completion, the Agent responds to the Client with a Configuration-Result-Notification indicating the operation status.

5.2.4. Policy and Mobility on the Agent

A Client may build Policy and Topology using Configure messages.

The Client may add, modify or delete many DPN Policies as DPN Policy Configurations and Mobility-Contexts in a single FPC message. This includes linking Mobility-Contexts to DPN Policies as well as creating the Policy, Rules Actions and Descriptors. As example, a Rule which performs re-writing of an arriving packet’s destination IP address from IP_A to IP_B matching an associated Descriptor, can be enforced in the Data-Plane via an Agent to implicitly consider matching arriving packet’s source IP address against IP_B and re-write the source IP address to IP_A.

Figure 27 illustrates the generic policy configuration model as used between a FPC Client and a FPC Agent.

```
      Descriptor_1 -+          +- Action_1
                    |          |
      Descriptor_2 -+--<Rule>--+- Action_2
                      +-----------+
                      /Precendent#/--------+
                      +----------+         |
                                           |
      Descriptor_3 -+          +- Action_3 +-<Policy>
                    |          |           |  ^
      Descriptor_4 -+--<Rule>--+- Action_4 |  |
                      +-----------+        |  |
                      /Precendent#/--------+  |
                      +----------+            |
                                             <DPN-Policy-Configuration>

      +---------------------+     +----------------------+
      | Bind 1..M traffic   |     |  Bind 1..N traffic   |
      |  Descriptors to     | --> |  treatment actions   |
      |  to a Policy        |     |   to a Policy        |
      +---------------------+     +----------------------+

     |                                                 |
     +-------------- Data-Plane Rule ------------------+
```

Figure 27: Structure of Configurable Policies

As depicted in Figure 27, the DPN Settings represents the anchor of Rules through the Policy / Rule hierarchy. A Client and Agent use the identifier of the associated Policy to directly access the Rule and perform modifications of traffic Descriptors or Action references. Arriving packets are matched against traffic according to Rule precedence and Descriptors. If a Rule is applicable the packet is treated according to the ordered Action values.

A Client associates a Precedence value for the Rule’s Descriptors, to allow unambiguous traffic matching on the Data-Plane.

Figure 28 illustrates the generic context configuration model as used between a Client and an Agent.

```
             <Policy 1>
                   ^
                   |
             <Service-Data-Flow 0> <--- <Mobility-Context-ID2>
                                                  ^
                                                  |
             <Policy 1>                           |
             ^                                    |
             |                                    |
             <Service-Data-Flow 0> <--- <Mobility-Context-ID1>

             +-------------------+     +---------------------+
             | Bind 1..M traffic |     |  Bind 1..N traffic  |
             |    selectors to   | --> |  treatment / qos    |
             |     a Context     |     |  actions to a       |
             |                   |     |       Context       |
             +-------------------+     +---------------------+

            |                                                 |
            +-------------- Data-Plane Rule ------------------+
```

Figure 28: Mobility Context Hierarchy

Figure 28 represents a mobility session hierarchy. A Client and Agent directly assigns values such as downlink traffic descriptors, QoS information, etc. A Client and Agent use the context identifiers to access the descriptors, qos information, etc. to perform modifications. From the viewpoint of packet processing, arriving packets are matched against traffic Descriptors and processed according to the qos or other mobility profile related Actions specified in the Mobility-Context’s and Service-Data-Flow’s’ properties. If present, a Policy could contain tunnel information to encapsulate and forward the packet.

A second Mobility-Context also references Mobility-Context-ID1 in the figure. Based upon the technology a property in a parent context (parent mobility-context-id reference) MAY be inherited by its descendants. This permits concise over the wire representation. When a Client deletes a parent Context all children are also deleted.

5.2.5. Monitor Example

The following example shows the installation of a DPN level monitor (1) to observe ifc1 status, a property that is either "up" or "down", and another monitor to watch for interface events. The interface experiences an outage which is reported to the Client via a Notify (3) message. At a later time a Probe (4) and corresponding Notify (5) is sent. Finally, the monitors are de-registered (6).

Note, specific event identifiers and types are out of scope.

```
                                                 +-------Router--------+
                            +-----------+        |+-------+ +---------+|
    +------+ +------+     +-----+ FPC   |         | FPC   | |  Anchor |
    |MAG-C1| |MAG-C2|     |LMA-C| Client|         | Agent | |   DPN   |
    +------+ +------+     +-----+-------+         +-------+ +---------+
    |         |            |---(1)--Configure-------->|           |
    |         |          "register-monitor" : {       |           |
    |         |           "client-id" : 0,            |           |
    |         |           "operation-id" : 0,         |           |
    |         |           "monitor" : [ {             |           |
    |         |            "monitor-key" : "ifc1-status",         |
    |         |            "target" : "/dpn/dpn1/interface/ifc1/status"
    |         |            "deferrable" : false                   |
    |         |            }, {                                   |
    |         |            "monitor-key" : "ifc1-events",         |
    |         |            "target" : "/dpn/dpn1/interface/ifc1"  |
    |         |            "deferrable" : false,                  |
    |         |            "configuration" : {                    |
    |         |                "target-event-configuration" : [ 0,
    |         |                  1, 3, .. ] } } ] }               |
    |         |            |<---(2)- Response --------|           |
    |         |            | {                        |           |
    |         |            |  "agent-id" : "agent1"," |           |
    |         |            |  "operation-id" : 0,     |           |
    |         |            |  "result-status" : "ok", |           |
    |         |            | }                        |           |
    |         |            |                          |           |
             [ ifc1 goes down which is reported as event type 3 ]
    |         |            |<---(3)-- NOTIFY ---------|           |
    |         |          "notify" : {                 |           |
    |         |           "notification-id" : 0,      |           |
    |         |           "timestamp" : ...,          |           |
    |         |           "report" : [ {              |           |
    |         |            "monitor-key" : "ifc1-events",         |
    |         |            "trigger" : "subscribed-event-occurred",
    |         |            "report-value" : { 3 } } ] }           |
    |         |            |                          |           |
    |         |            |---(4)--  Probe  -------->|           |
    |         |          "probe" : {      |           |
    |         |           "client-id" : 0,            |           |
    |         |           "operation-id" : 1,         |           |
    |         |           "monitor" : [               |           |
    |         |            "monitor-key" : "ifc1-status" ] }      |
    |         |            |<---(5)- Response --------|           |
    |         |            | {                        |           |
    |         |            |  "agent-id" : "agent1"," |           |
    |         |            |  "operation-id" : 1,     |           |
    |         |            |  "result-status" : "ok", |           |
    |         |            | }                        |           |
    |         |            |                          |           |
    |         |            |<---(6)-- NOTIFY ---------|           |
    |         |          "notify" : {                 |           |
    |         |           "notification-id" : 1,      |           |
    |         |           "timestamp" : ...,          |           |
    |         |           "report" : [ {              |           |
    |         |            "monitor-key" : "ifc1-status",         |
    |         |            "trigger" : "probe",
    |         |            "report-valuerporte" : { "up" } } ] }  |
    |         |            |                          |           |
    |         |            |---(7)- Deregister ------>|           |
    |         |          "deregister-monitor" : {     |           |
    |         |           "client-id" : 0,            |           |
    |         |           "operation-id" : 2,         |           |
    |         |           "monitor" : [               |           |
    |         |            { "monitor-key" : "ifc1-events" },     |
    |         |            { "monitor-key" : "ifc1-status",       |
    |         |              "send-data" : true } ] }             |
    |         |            |<---(8)- Response --------|           |
    |         |            | {                        |           |
    |         |            |  "agent-id" : "agent1"," |           |
    |         |            |  "operation-id" : 2,     |           |
    |         |            |  "result-status" : "ok", |           |
    |         |            | }                        |           |
    |         |            |                          |           |
    |         |            |<---(9)-- NOTIFY ---------|           |
    |         |          "notify" : {                 |           |
    |         |           "notification-id" : 2,      |           |
    |         |           "timestamp" : ...,          |           |
    |         |           "report" : [ {              |           |
    |         |            "monitor-key" : "ifc1-status",         |
    |         |            "trigger" : "deregistration-final-value",
    |         |            "report-value" : { "up" } } ] }        |
```

Figure 29: Monitor Example (focus on FPC reference point)

6. Templates and Command Sets

Configuration templates are shown below.

6.1. Monitor Configuration Templates

A periodic configuration specifies a time interval (ms) for reporting.

A scheduled configuration specifies a time for reporting.

A threshold configuration MUST have at least one hi or low threshold and MAY have both.

A Target-Events-Configuration is a list of Events that, when generated by the Target, results in a Monitor notification.

```
     |
     +-[Monitor] <List>
     ...
     |         +-[Configuration]
     |         |     +-[Periodic-Configuration]
     |         |     |       +-[(Unsigned32) Period:]
     ...
     |         +-[Configuration]
     |         |     +-[Schedule-Configuration]
     |         |     |       +-[(Unsigned32) Schedule:]
     ...
     |         +-[Configuration]
     |         |     +-[Threshold-Configuration]
     |         |     |       +-[(Unsigned32) Low]
     |         |     |       +-[(Unsigned32) Hi]
     ...
     |         +-[Configuration]
     |         |     +-[Target-Events-Configuration]
     |         |     |       +-[(Unsigned32) Event-Key:] <List>
```

Figure 30: Monitor Configuration Templates

6.2. Descriptor Templates

A IP-Prefix-Template MUST have at least the To or From IP Prefix / Length populated. The IP Prefix specifies and Address and Length.

The PMIP Traffic Selector template is mapped according to [RFC6088]

The RFC 5777 Classifier is a structured version of common filter rules and follows the format specified in [RFC5777]. The Flow-Label, Flow-Label range and ECN-IP-Codepoint specified in [RFC7660] are added to the Descriptor as well.

```
  |
  +-[ip-prefix-template]
  |       +-[(IP Prefix / Length) To-IP-Prefix]
  |       +-[(IP Prefix / Length) From-IP-Prefix]
  ...
  +-[pmip-traffic-selector]
```

```
     |       +-[(Enumerated - IPv4 or IPv6) ts-format]
     |       +-[ipsec-spi-range]
     |       |   +-[ (ipsec-spi) start-spi: ]
     |       |   +-[ (ipsec-spi) end-spi ]
     |       +-[source-port-range]
     |       |   +-[ (port-number) start-port: ]
     |       |   +-[ (port-number) end-port ]
     |       +-[destination-port-range]
     |       |   +-[ (port-number) start-port: ]
     |       |   +-[ (port-number) end-port ]
     |       +-[source-address-range-v4]
     |       |   +-[ (ipv4-address) start-address: ]
     |       |   +-[ (ipv4-address) end-address ]
     |       +-[destination-address-range-v4]
     |       |   +-[ (ipv4-address) start-address: ]
     |       |   +-[ (ipv4-address) end-address ]
     |       +-[ds-range]
     |       |   +-[ (dscp) start-ds: ]
     |       |   +-[ (dscp) end-ds ]
     |       +-[protocol-range]
     |       |   +-[ (uint8) start-protocol: ]
     |       |   +-[ (uint8) end-protocol ]
     |       +-[source-address-range-v6]
     |       |   +-[(ipv6-address) start-address: ]
     |       |   +-[(ipv6-address) end-address ]
     |       +-[destination-address-range-v6]
     |       |   +-[(ipv6-address) start-address: ]
     |       |   +-[(ipv6-address) end-address ]
     |       +-[flow-label-range]
     |       |   +-[(ipv6-flow-label) start-flow-label ]
     |       |   +-[(ipv6-flow-label) end-flow-label ]
     |       +-[traffic-class-range]
     |       |   +-[ (dscp) start-traffic-class ]
     |       |   +-[ (dscp) end-traffic-class ]
     |       +-[next-header-range]
     |       |   +-[ (uint8) start-next-header ]
     |       |   +-[ (uint8) end-next-header ]
     ...
     +-[rfc5777-classifier]
     |       +-[Extensible: True]
     |       +-[(uint8) protocol]
     |       +-[(Enumerated - In/Out/Both) Direction]
     |       +-[From-Spec] <List>
     |       |    +-[(ip-address) IP-Address] <List>
     |       |    +-[IP-Address-Range] <List>
     |       |    |    +-[(ip-address) IP-Address-Start]
     |       |    |    +-[(ip-address) IP-Address-End]
     |       |    +-[IP-Address-Mask] <List>
     |       |    |    +-[(ip-address) IP-Address:]
     |       |    |    +-[(Unsigned 32) IP-Bit-Mask-Width:]
     |       |    +-[(mac-address) MAC-Address] <List>
     |       |    +-[MAC-Address-Mask] <List>
     |       |    |    +-[(mac-address) MAC-Address:]
     |       |    |    +-[(mac-address) MAC-Address-Mask-Pattern:]
     |       |    +-[(eui64-address) EUI64-Address] <List>
     |       |    +-[EUI64-Address-Mask] <List>
     |       |    |    +-[(eui64-address) EUI64-Address:]
     |       |    |    +-[(eui64-address) EUI64-Address-Mask-Pattern:]
     |       |    +-[(Integer 32) Port] <List>
     |       |    +-[Port-Range] <List>
     |       |    |    +-[(Integer 32) Port-Start]
     |       |    |    +-[(Integer 32) Port-End]
     |       |    +-[(Boolean) Negated]
     |       |    +-[(Boolean) Use-Assigned-Address]
     |       +-[To-Spec] <List> (O)
     |       |    +-[(ip-address) IP-Address] <List>
     |       |    +-[IP-Address-Range] <List>
     |       |    |    +-[(ip-address) IP-Address-Start]
     |       |    |    +-[(ip-address) IP-Address-End]
     |       |    +-[IP-Address-Mask] <List>
     |       |    |    +-[(ip-address) IP-Address:]
     |       |    |    +-[(Unsigned 32) IP-Bit-Mask-Width:]
     |       |    +-[(mac-address) MAC-Address] <List>
     |       |    +-[MAC-Address-Mask] <List>
     |       |    |    +-[(mac-address) MAC-Address:]
     |       |    |    +-[(mac-address) MAC-Address-Mask-Pattern:]
     |       |    +-[(eui64-address) EUI64-Address] <List>
     |       |    +-[EUI64-Address-Mask] <List>
     |       |    |    +-[(eui64-address) EUI64-Address:]
     |       |    |    +-[(eui64-address) EUI64-Address-Mask-Pattern:]
     |       |    +-[(Integer 32) Port] <List>
     |       |    +-[Port-Range] <List>
     |       |    |    +-[(Integer 32) Port-Start]
     |       |    |    +-[(Integer 32) Port-End]
     |       |    +-[(Boolean) Negated]
     |       |    +-[(Boolean) Use-Assigned-Address]
     |       +-[(dscp) Diffserv-Code-Point] <List>
     |       +-[(Boolean) Fragmentation-Flag ~ False]
     |       +-[IP-Option] <List>
     |       +-[TCP-Option] <List>
     |       +-[TCP-Flags]
     |       +-[ICMP-Type] <List>
     |       +-[ETH-Option] <List>
     |       +-[ecn-ip-codepoint] <List>
     |       +-[(flowlabel) flow-label] <List>
     |       +-[flow-label-range] <List>
     |       |    +-[(flowlabel) flow-label-start]
     |       |    +-[(flowlabel) flow-label-end]
```

Figure 31: Descriptor Templates

6.3. Tunnel Templates

The Network Service Header is specified in [RFC8300].

The MPLS SR Stack is specified in [I-D.ietf-spring-segment-routing-mpls].

The IPv6 SR Stack is specified in [I-D.ietf-6man-segment-routing-header].

A tunnel MUST have the local-address or remote-address (or both) populated.

For GRE, the gre-key MUST be present.

For GTP (GPRS Tunneling Protocol), the following attributes MAY be present

local tunnel endpoint identifier (teid) - MUST be present if local-address is nonempty

remote tunnel endpoint identifier (teid) - MUST be present if remote-address is nonempty

sequence-numbers-on - Indicates that sequence numbers will be used

Tunnels can be used as Next Hop and Descriptor values.

```
     |
     +-[next-hop-template]
     |     +-[Extensible: True]
     |     +-[(ip-address) address]
     |     +-[(mac-address) mac-address]
     |     +-[(service-path-id) service-path]
     |     +-[(mpls-label) mpls-path]
     |     +-[(network service header) nsh]
     |     +-[(Unsigned Integer) interface]
     |     +-[(Unsigned 128) segment-identifier]
     |     +-[(MPLS Stack) mpls-label-stack]
     |     +-[(MPLS SR Stack) mpls-sr-stack]
     |     +-[(IPv6 SR Stack) srv6-stack]
     |     +-[tunnel-template]
     ...
     |
     +-[tunnel-template]
     |     +-[Extensible: True]
     |     +-[(address) local-address]
     |     +-[(address) remote-address]
     |     +-[mtu]
     |     +-[(Enumeration - ipv4(0), ipv6(1), dual(2) payload_type:]
     |     +-[(Enumeration - ip-in-ip(0),
                   udp(1), gre(2), gtpv1(3), gtpv2(4)) type:]
     |     +-[interface]
     |     +-[next-hop]
     |     +-[gre-key:] (type == gre)
     |     +-[gtp-info] (type == gtpv1 or type == gtpv2 )
     |     |    +-[(Unsigned 32) local-teid]
     |     |    +-[(Unsigned 32) remote-teid]
     |     |    +-[(Boolean) sequence-numbers-on] (type == gtpv1)
```

Figure 32: Tunnel Templates

6.4. Action Templates

The following figure shows common next-hop (set next-hop) and tunnel templates for Actions.

Drop action has no values.

Rewrite uses a Descriptor to set the values of the packet. Exactly one Descriptor MUST be present. Only the Destination and Source port fields, if present, are used from the Descriptor.

Copy-Forward creates a copy of the packet and then forwards it in accordance to the nexthop value.

```
                      |
                      +-[drop-template]
                      ...
                      |
                      +-[rewrite-template]
                      |     +-[Extensible: True]
                      |     +-[ip-prefix-template]
                      |     +-[pmip-traffic-selector]
                      |     +-[rfc5777-classifier]
                      ...
                      |
                      +-[copy-forward-template]
                      |     +-[Extensible: True]
                      |     +-[next-hop:]
```

Figure 33: Action Templates

6.5. Quality of Service Action Templates

PMIP QoS is specified in [RFC7222].

```
       |
       +-[qos-template]
       |     +-[Extensible: True]
       |     +-[(dscp) trafficclass]
       |     +-[pmip-qos]
       |     |     +-[(Unsigned 32) per-mn-agg-max-dl]
       |     |     +-[(Unsigned 32) per-mn-agg-max-ul]
       |     |     +-[per-session-agg-max-dl]
       |     |     |    +-[(Unsigned 32) max-rate:]
       |     |     |    +-[(Boolean) service-flag:]
       |     |     |    +-[(Boolean) exclude-flag:]
       |     |     +-[per-session-agg-max-ul]
       |     |     |    +-[(Unsigned 32) max-rate:]
       |     |     |    +-[(Boolean) service-flag:]
       |     |     |    +-[(Boolean) exclude-flag:]
       |     |     +-[allocation-retention-priority]
       |     |     |    +-[(Unsigned 8) priority-level:]
       |     |     |    +-[(Enumeration) preemption-capability:]
       |     |     |    +-[(Enumeration) preemption-vulnerability:]
       |     |     +-[(Unsigned 32) agg-max-dl]
       |     |     +-[(Unsigned 32) agg-max-ul]
       |     |     +-[(Unsigned 32) gbr-dl]
       |     |     +-[(Unsigned 32) gbr-ul]
```

Figure 34: QoS Templates

6.6. PMIP Command-Set

The following Command Set values are supported for IETF PMIP.

- assign-ip - Assign the IP Address for the mobile session.
- assign-dpn - Assign the Data-plane Node.
- session - Assign values for the Session Level.
- uplink - Command applies to uplink.
- downlink - Command applies to downlink.

6.7. 3GPP Specific Templates and Command-Set

3GPP support is optional and detailed in this section. The following acronyms are used:

APN-AMBR: Access Point Name Aggregate Maximum Bit Rate

UE-AMBR: User Equipment Aggregate Maximum Bit Rate

QCI: QoS Class Identifier

EBI: EPS Bearer Identity

LBI: Linked Bearer Identity

IMSI: International Mobile Subscriber Identity

TFT: Traffic Flow Template (TFT)

Generally, 3GPP QoS values should use the qos-template. Note: User Equipment Aggregate Maximum Bit Rate (UE-AMBR) maps to the per-mn-agg-max-dl and per-mn-agg-max-ul.

```
                      |
                      +-[ MN-Policy-Template ]
                      |     +-[(Unsigned 64) imsi:]
                      ...
                      +-[tunnel-template]
                      |     +-[Extensible: True]
                      |     +-[(unsigned 4) ebi:]
                      |     +-[(unsigned 4) lbi]
                      ...
                      +-[qos-template]
                      |     +-[Extensible: True]
                      |     +-[(unsigned 4) qos-class-identifier]
                      |     +-[(Unsigned 32) ue-agg-max-bitrate]
                      |     +-[(Unsigned 32) apn-agg-max-bitrate]
                      ...
```

Figure 35: 3GPP Mobility Templates

```
            |
            +-[ packet-filter ]
            |     +-[Extensible: True]
            |     +-[(Unsigned 8) identifier:]
            |     +-[Contents:] <List>
            |     |     +-[(ip-address) ipv4-ipv6-local]
            |     |     +-[(ipv6-prefix) ipv6-prefix-local]
            |     |     +-[(ip-address) ipv4-ipv6-remote]
            |     |     +-[(ipv6-prefix) ipv6-prefix-remote]
            |     |     +-[(Unsigned 8) protocol-next-header]
            |     |     +-[(Unsigned 16) local-port]
            |     |     +-[local-port-range]
            |     |     |      +-[(Unsigned 16) local-port-lo]
            |     |     |      +-[(Unsigned 16) local-port-hi]
            |     |     +-[(Unsigned 16) remote-port]
            |     |     +-[remote-port-range]
            |     |     |      +-[(Unsigned 16) remote-port-lo]
            |     |     |      +-[(Unsigned 16) remote-port-hi]
            |     |     +-[(Unsigned 32) sec-parameter-index]
            |     |     +-[(dscp) traffic-class]
            |     |     +-[traffic-class-range]
            |     |     |      +-[(dscp) traffic-class-lo]
            |     |     |      +-[(dscp) traffic-class-hi]
            |     |     +-[(dscp) flow-label]
            ...
```

Figure 36: 3GPP Packet Filter Template (Descriptor)

The following Command Set values are supported for 3GPP.

- o assign-ip - Assign the IP Address for the mobile session.

- o assign-fteid-ip - Assign the Fully Qualified TEID (F-TEID) LOCAL IP address.

- o assign-fteid-teid - Assign the Fully Qualified TEID (F-TEID) LOCAL TEID.

- o session - Assign values for the Session Level. When this involves ’assign-fteid-ip’ and ’assign-fteid-teid’, the values are part of the default bearer.

- o uplink - Command applies to uplink.

- o downlink - Command applies to downlink.

- o assign-dpn - Assign the Data-plane Node.

7. Implementation Status

Three FPC Agent implementations have been made to date. The first was based upon Version 03 of the draft and followed Model 1. The second follows Version 04 of the document. Both implementations were OpenDaylight plug-ins developed in Java by Sprint. Version 04 is now primarily enhanced by GS Labs. Version 03 was known as fpcagent and version 04’s implementation is simply referred to as ’fpc’. A third has been developed on an ONOS Controller for use in MCORD projects.

fpcagent’s intent was to provide a proof of concept for FPC Version 03 Model 1 in January 2016 and research various errors, corrections and optimizations that the Agent could make when supporting multiple DPNs.

As the code developed to support OpenFlow and a proprietary DPN from a 3rd party, several of the advantages of a multi-DPN Agent became obvious including the use of machine learning to reduce the number of Flows and Policy entities placed on the DPN. This work has driven new efforts in the DIME WG, namely Diameter Policy Groups [I-D.bertz-dime-policygroups].

A throughput performance of tens per second using various NetConf based solutions in OpenDaylight made fpcagent, based on version 03, undesirable for call processing. The RPC implementation improved throughput by an order of magnitude but was not useful based upon FPC’s Version 03 design using two information models. During this time the features of version 04 and its converged model became attractive and the fpcagent project was closed in August 2016.

fpcagent will no longer be developed and will remain a proprietary implementation.

The learnings of fpcagent has influenced the second project, fpc. Fpc is also an OpenDaylight project but is an open source release as the Opendaylight FpcAgent plugin (https://wiki.opendaylight.org/view/Project_Proposals:FpcAgent). This project is scoped to be a fully compliant FPC Agent that supports multiple DPNs including those that communicate via OpenFlow. The following features present in this draft and others developed by the FPC development team have already led to an order of magnitude improvement.

> Migration of non-realtime provisioning of entities such as topology and policy allowed the implementation to focus only on the rpc.
>
> Using only 5 messages and 2 notifications has also reduced implementation time.
>
> Command Sets, an optional feature in this specification, have eliminated 80% of the time spent determining what needs to be done with a Context during a Create or Update operation.
>
> Op Reference is an optional feature modeled after video delivery. It has reduced unnecessary cache lookups. It also has the additional benefit of allowing an Agent to become cacheless and effectively act as a FPC protocol adapter remotely with multi-DPN support or co-located on the DPN in a single-DPN support model.
>
> Multi-tenant support allows for Cache searches to be partitioned for clustering and performance improvements. This has not been capitalized upon by the current implementation but is part of the development roadmap.
>
> Use of Contexts to pre-provision policy has also eliminated any processing of Ports for DPNs which permitted the code for CONFIGURE and CONF_BUNDLE to be implemented as a simple nested FOR loops (see below).

Initial v04 performance results without code optimizations or tuning allow reliable provisioning of 1K FPC Mobility-Contexts processed per second on a 12 core server. This results in 2x the number of transactions on the southbound interface to a proprietary DPN API on the same machine.

fpc currently supports the following:

1 proprietary DPN API

Policy and Topology as defined in this specification using OpenDaylight North Bound Interfaces such as NetConf and RestConf

CONFIG and CONF_BUNDLE (all operations)

DPN assignment, Tunnel allocations and IPv4 address assignment by the Agent or Client.

Immediate Response is always an OK_NOTIFY_FOLLOWS.

```
assignment system (receives rpc call):
  perform basic operation integrity check
  if CONFIG then
    goto assignments
    if assignments was ok then
      send request to activation system
      respond back to client with assignment data
    else
      send back error
    end if
  else if CONF_BUNDLE then
    for each operation in bundles
    goto assignments
    if assignments was ok then
      hold onto data
    else
      return error with the assignments that occurred in
        prior operations (best effort)
    end if
    end for
    send bundles to activation systems
  end if

assignments:
  assign DPN, IPv4 Address and/or tunnel info as required
  if an error occurs undo all assignments in this operation
  return result

activation system:
  build cache according to op-ref and operation type
  for each operation
    for each Context
      for each DPN / direction in Context
        perform actions on DPN according to Command Set
      end for
    end for
  end for
  commit changes to in memory cache
  log transaction for tracking and notification
                                (CONFIG_RESULT_NOTIFY)
```

Figure 37: fpc pseudo code

For further information please contact Lyle Bertz who is also a co-author of this document.

NOTE: Tenant support requires binding a Client ID to a Tenant ID (it is a one to many relation) but that is outside of the scope of this specification. Otherwise, the specification is complete in terms of providing sufficient information to implement an Agent.

8. Security Considerations

Detailed protocol implementations for DMM Forwarding Policy Configuration must ensure integrity of the information exchanged between a FPC Client and a FPC Agent. Required Security Associations may be derived from co-located functions, which utilize the FPC Client and FPC Agent respectively.

The YANG modules defined in this memo are designed to be accessed via the NETCONF [RFC6241] or RESTCONF [RFC8040] protocol. The lowest NETCONF layer is the secure transport layer and the mandatory-to-implement secure transport is SSH [RFC6242].

The information model defined in the memo is designed to be access by protocols specified in extensions to this document or, if using the YANG modules, as described above.

There are a number of data nodes defined which are writable/creatable/deletable. These data nodes may be considered sensitive or vulnerable in some network environments. Write operations (e.g., a NETCONF 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:

> Nodes under the Policy tree provide generic policy enforcement and traffic classification. They can be used to block or permit traffic. If this portion of the model was to be compromised it may be used to block, identify or permit traffic that was not intended by the Tenant or FPC Client.
>
> Nodes under the Topology tree provide definition of the Tenant’s forwarding topology. Any compromise of this information will provide topology information that could be used for subsequent attack vectors. Removal of topology can limit services.
>
> Mobility-Context provides runtime only information and manipulated by remote procedure calls. The unwanted deletion or removal of such information would deny users service or provide services to unauthorized parties.

Some of the readable data nodes defined 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:

IP address assignments in the Mobility-Context along with their associated tunnel configurations/identifiers (from the FPC base module)

Internaitonal Mobile Subscriber Identity (IMSI) and bearer identifiers in the Context when using the FPC base model

Some of the RPC operations defined may be considered sensitive or vulnerable in some network environments. It is thus important to control access to these operations. These are the operations and their sensitivity/vulnerability:

Configure sends Mobility-Context information which can include information of a sensitive or vulnerable nature in some network environments as described above.

Monitor related RPC operations do not specifically provide sensitive or vulnerable information but care must be taken by users to avoid identifier values that expose sensitive or vulnerable information.

Notifications MUST be treated with same level of protection and scrutiny as the operations they correspond to. For example, a Configure-Result-Notification provides the same information that is sent as part of the input and output of the Configure RPC operation.

General usage of FPC MUST consider the following:

FPC Naming Section 4.5 permits arbitrary string values but a user MUST avoid placing sensitive or vulnerable information in those values.

Policies that are very narrow and permit the identification of specific traffic, e.g. that of a single user, SHOULD be avoided.

9. IANA Considerations

This document registers six URIs in the "IETF XML Registry" [RFC3688]. Following the format in RFC 3688, the following registrations have been made.

URI: urn:ietf:params:xml:ns:yang:ietf-dmm-fpc
Registrant Contact: The DMM WG of the IETF.
XML: N/A, the requested URI is an XML namespace.

URI: urn:ietf:params:xml:ns:yang:ietf-dmm-pmip-qos
Registrant Contact: The DMM WG of the IETF.
XML: N/A, the requested URI is an XML namespace.

URI: urn:ietf:params:xml:ns:yang:ietf-dmm-traffic-selector-types
Registrant Contact: The DMM WG of the IETF.
XML: N/A, the requested URI is an XML namespace.

URI: urn:ietf:params:xml:ns:yang:ietf-dmm-fpc-settingsext
Registrant Contact: The DMM WG of the IETF.
XML: N/A, the requested URI is an XML namespace.

URI: urn:ietf:params:xml:ns:yang:ietf-diam-trafficclassifier
Registrant Contact: The DMM WG of the IETF.
XML: N/A, the requested URI is an XML namespace.

This document registers the following YANG modules in the "YANG Module Names" registry [RFC6020].

```
     name:         ietf-dmm-fpc
     namespace:    urn:ietf:params:xml:ns:yang:ietf-dmm-fpc
     prefix:       fpc
     reference:    TBD1

     name:         ietf-dmm-pmip-qos
     namespace:    urn:ietf:params:xml:ns:yang:ietf-dmm-pmip-qos
     prefix:       qos-pmip
     reference:    TBD2

     name:         ietf-dmm-traffic-selector-types
     namespace:    urn:ietf:params:xml:ns:yang:
       ietf-dmm-traffic-selector-types
     prefix:       traffic-selectors
     reference:    TBD3

     name:         ietf-dmm-fpc-settingsext
     namespace:    urn:ietf:params:xml:ns:yang:
       ietf-dmm-fpc-settingsext
     prefix:       fpcbase
     reference:    TBD4

       name:         ietf-diam-trafficclassifier
       namespace:    urn:ietf:params:xml:ns:yang:
       ietf-diam-trafficclassifier
       prefix:       diamclassifier
       reference:    TBD5
```

10. Work Team Participants

Participants in the FPSM work team discussion include Satoru Matsushima, Danny Moses, Sri Gundavelli, Marco Liebsch, Pierrick Seite, Alper Yegin, Carlos Bernardos, Charles Perkins and Fred Templin.

11. References

11.1. Normative References

[I-D.ietf-6man-segment-routing-header]
Previdi, S., Filsfils, C., Leddy, J., Matsushima, S., and d. daniel.voyer@bell.ca, "IPv6 Segment Routing Header (SRH)", draft-ietf-6man-segment-routing-header-13 (work in progress), May 2018.

[I-D.ietf-spring-segment-routing-mpls]
Bashandy, A., Filsfils, C., Previdi, S., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing with MPLS data plane", draft-ietf-spring-segment-routing-mpls-14 (work in progress), June 2018.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC5777] Korhonen, J., Tschofenig, H., Arumaithurai, M., Jones, M., Ed., and A. Lior, "Traffic Classification and Quality of Service (QoS) Attributes for Diameter", RFC 5777, DOI 10.17487/RFC5777, February 2010, <https://www.rfc-editor.org/info/rfc5777>.

[RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <https://www.rfc-editor.org/info/rfc6020>.

[RFC6088] Tsirtsis, G., Giarreta, G., Soliman, H., and N. Montavont, "Traffic Selectors for Flow Bindings", RFC 6088, DOI 10.17487/RFC6088, January 2011, <https://www.rfc-editor.org/info/rfc6088>.

[RFC6991] Schoenwaelder, J., Ed., "Common YANG Data Types", RFC 6991, DOI 10.17487/RFC6991, July 2013, <https://www.rfc-editor.org/info/rfc6991>.

[RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", RFC 8040, DOI 10.17487/RFC8040, January 2017, <https://www.rfc-editor.org/info/rfc8040>.

[RFC8072] Bierman, A., Bjorklund, M., and K. Watsen, "YANG Patch Media Type", RFC 8072, DOI 10.17487/RFC8072, February 2017, <https://www.rfc-editor.org/info/rfc8072>.

[RFC8300] Quinn, P., Ed., Elzur, U., Ed., and C. Pignataro, Ed., "Network Service Header (NSH)", RFC 8300, DOI 10.17487/RFC8300, January 2018, <https://www.rfc-editor.org/info/rfc8300>.

[RFC8342] Bjorklund, M., Schoenwaelder, J., Shafer, P., Watsen, K., and R. Wilton, "Network Management Datastore Architecture (NMDA)", RFC 8342, DOI 10.17487/RFC8342, March 2018, <https://www.rfc-editor.org/info/rfc8342>.

11.2. Informative References

[I-D.bertz-dime-policygroups]
Bertz, L. and M. Bales, "Diameter Policy Groups and Sets", draft-bertz-dime-policygroups-05 (work in progress), December 2017.

[I-D.ietf-dmm-deployment-models]
Gundavelli, S. and S. Jeon, "DMM Deployment Models and Architectural Considerations", draft-ietf-dmm-deployment-models-04 (work in progress), May 2018.

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, DOI 10.17487/RFC3688, January 2004, <https://www.rfc-editor.org/info/rfc3688>.

[RFC3958] Daigle, L. and A. Newton, "Domain-Based Application Service Location Using SRV RRs and the Dynamic Delegation Discovery Service (DDDS)", RFC 3958, DOI 10.17487/RFC3958, January 2005, <https://www.rfc-editor.org/info/rfc3958>.

[RFC5213] Gundavelli, S., Ed., Leung, K., Devarapalli, V., Chowdhury, K., and B. Patil, "Proxy Mobile IPv6", RFC 5213, DOI 10.17487/RFC5213, August 2008, <https://www.rfc-editor.org/info/rfc5213>.

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <https://www.rfc-editor.org/info/rfc6241>.

[RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure Shell (SSH)", RFC 6242, DOI 10.17487/RFC6242, June 2011, <https://www.rfc-editor.org/info/rfc6242>.

[RFC7222] Liebsch, M., Seite, P., Yokota, H., Korhonen, J., and S. Gundavelli, "Quality-of-Service Option for Proxy Mobile IPv6", RFC 7222, DOI 10.17487/RFC7222, May 2014, <https://www.rfc-editor.org/info/rfc7222>.

[RFC7333] Chan, H., Ed., Liu, D., Seite, P., Yokota, H., and J. Korhonen, "Requirements for Distributed Mobility Management", RFC 7333, DOI 10.17487/RFC7333, August 2014, <https://www.rfc-editor.org/info/rfc7333>.

[RFC7660] Bertz, L., Manning, S., and B. Hirschman, "Diameter Congestion and Filter Attributes", RFC 7660, DOI 10.17487/RFC7660, October 2015, <https://www.rfc-editor.org/info/rfc7660>.

Appendix A. YANG Data Model for the FPC protocol

This section provides a type mapping for FPC structures in YANG. When being mapped to a specific information such as YANG the data type MAY change.

Keys for Actions, Descriptors, Rules, Policies, DPNs, Domains and Mobility-Contexts are specified as FPC-Identity which follows rules according to Section 4.5.

Action and Descriptor Templates are mapped as choices. This was done to ensure no duplication of Types and avoid use of identityref for typing.

Policy Expressions are provided as default values. NOTE that a static value CANNOT be supported in YANG.

Mapping of templates to YANG are performed as follows:

> Value is defined as a choice statement for extensibility and therefore a type value is not necessary to discriminated types
>
> Generic attributes are distinguished by the "Settings" type and holds ANY value. It is an any data node under configurations.

The CONFIGURE and CONFIGURE-RESULT-NOTIFICATION use the yang-patch-status which is a container for edits. This was done to maximize YANG reuse.

In the configure rpc, operation-id is mapped to patch-id and in an edit the edit-type is mapped to operation.

The Result-Status attribute is mapped to the ’ok’ (empty leaf) or errors structure.

The Policy-Status is mapped to entity-state to reduce YANG size.

Five modules are defined:

o ietf-dmm-fpc (fpc) - Defines the base model and messages for FPC that are meant to be static in FPC.

o ietf-dmm-fpc-settingsext - A FPC module that defines the information model elements that are likely to be extended in FPC.

o ietf-pmip-qos (pmip-qos) - Defines proxy mobile IPv6 QoS parameters per RFC 7222

o ietf-trafficselectors-types (traffic-selectors) - Defines Traffic Selectors per [RFC6088]

o ietf-diam-trafficclassifier (diamclassifier) - Defines the Classifier per [RFC5777]

All modules defined in this specification make use of (import) ietf-inet-types as defined in [RFC6991].

ietf-dmm-fpc-settingsext and ietf-diam-trafficclassifier make use of (imports) ietf-yang-types as defined in [RFC6991].

ietf-dmm-fpc imports the restconf (ietf-restconf) [RFC8040] and yang patch (ietf-yang-patch) [RFC8072] modules.

ietf-pmip-qos and ietf-dmm-fpc-settings import the trafficselector from the ietf-traffic-selector-types module.

ietf-dmm-fpc-settings also imports the qosattribute (ietf-pmip-qos) and classifier (ietf-diam-trafficclassifier).

ietf-dmm-fpc-settingsext groups various settings, actions and descriptors and is used by the fpc module (ietf-dmm-fpc).

The following groupings are intended for reuse (import) by other modules.

o qosoption (ietf-qos-pmip module)

- qosattribute (ietf-qos-pmip module)

- qosoption (ietf-qos-pmip module)

- Allocation-Retention-Priority-Value (ietf-qos-pmip module)

- trafficselector (ietf-traffic-selector-types)

- classifier (ietf-diam-trafficclassifier)

- packet-filter (ietf-dmm-fpc-settingsext)

- instructions (ietf-dmm-fpc-settingsext)

- fpc-descriptor-value (ietf-dmm-fpc-settingsext)

- fpc-action-value (ietf-dmm-fpc-settingsext)

The YANG modules in this document conform to the Network Management Datastore Architecture (NMDA) defined in [RFC8342].

DPNs conformant to NMDA MAY only have policies, installed policies, topology, domains and mobility session information that has been assigned to it in its intended and operational datastores. What is housed in the operational datastore MAY be determined on a per DPN basis and using the Entity-Status as a guideline based upon tradeoffs described in Section 4.6.

ServiceGroups are not expected to appear in operational datastores of DPNs as they remain in and are used by FPC Agents and Clients. They MAY be operationally present in DNS when using the Dynamic Delegation and Discovery System (DDDS) as defined in [RFC3958] or the operational datastore of systems that provide equivalent functionality.

A.1. FPC YANG Model

This module defines the information model and protocol elements specified in this document.

This module references [RFC6991], [RFC8040] and the fpc-settingsext module defined in this document.

```
<CODE BEGINS> file "ietf-dmm-fpc@2018-05-17.yang"
module ietf-dmm-fpc {
  yang-version 1.1;
    namespace "urn:ietf:params:xml:ns:yang:ietf-dmm-fpc";
    prefix fpc;
```

```
       import ietf-inet-types { prefix inet;
           revision-date 2013-07-15; }
       import ietf-dmm-fpc-settingsext { prefix fpcbase;
           revision-date 2018-05-17; }
       import ietf-diam-trafficclassifier { prefix rfc5777;
           revision-date 2018-05-17; }
       import ietf-restconf { prefix rc;
           revision-date 2017-01-26; }
       import ietf-yang-patch { prefix ypatch;
           revision-date 2017-02-22; }

       organization "IETF Distributed Mobility Management (DMM)
         Working Group";

       contact
          "WG Web:   <http://tools.ietf.org/wg/netmod/>
           WG List:  <mailto:netmod@ietf.org>

           WG Chair: Dapeng Liu
                     <mailto:maxpassion@gmail.com>

           WG Chair: Jouni Korhonen
                     <mailto:jouni.nospam@gmail.com>

           Editor:   Satoru Matsushima
                     <mailto:satoru.matsushima@g.softbank.co.jp>

           Editor:   Lyle Bertz
                     <mailto:lylebe551144@gmail.com>";

       description
       "This module contains YANG definition for
        Forwarding Policy Configuration Protocol (FPCP).

        Copyright (c) 2016 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.";

       revision 2018-05-17 {
       description "Initial Revision.";
       reference "draft-ietf-dmm-fpc-cpdp-10";
     }

       //General Structures
       grouping templatedef {
           leaf extensible {
               type boolean;
               description "Indicates if the template is extensible";
           }
           leaf-list static-attributes {
               type string;
               description "Attribute (Name) whose value cannot
                   change";
           }
           leaf-list mandatory-attributes {
               type string;
               description "Attribute (Name) of optional attributes
                 that MUST be present in instances of this tempplate.";
           }
           leaf entity-state {
               type enumeration {
                   enum initial {
                       description "Inital Configuration";
                   }
                   enum partially-configured {
                       description "Partial Configuration";
                   }
                   enum configured {
                       description "Confgured";
                   }
                   enum active {
                       description "Active";
                   }
               }
               default initial;
               description "Entity State";
           }
           leaf version {
               type uint32;
               description "Template Version";
           }
           description "Teamplate Definition";
       }
       typedef fpc-identity {
           type union {
               type uint32;
               type instance-identifier;
```

```
                type string;
            }
            description "FPC Identity";
        }
        grouping index {
            leaf index {
                type uint16;
                description "Index";
            }
            description "Index Value";
        }

        // Policy Structures
        grouping descriptor-template-key {
            leaf descriptor-template-key {
                type fpc:fpc-identity;
                mandatory true;
                description "Descriptor Key";
            }
            description "Descriptor-Template Key";
        }
        grouping action-template-key {
            leaf action-template-key {
                type fpc:fpc-identity;
                mandatory true;
                description "Action Key";
            }
            description "Action-Template Key";
        }
        grouping rule-template-key {
            leaf rule-template-key {
                type fpc:fpc-identity;
                mandatory true;
                description "Rule Identifier";
            }
            description "Rule Key";
        }
        grouping policy-template-key {
            leaf policy-template-key {
                type fpc:fpc-identity;
                mandatory true;
                description "Rule Identifier";
            }
            description "Rule Key";
        }

        grouping fpc-setting-value {
            anydata setting;
```

```
           description "FPC Setting Value";
       }
       // Configuration / Settings
       grouping policy-configuration-choice {
              choice policy-configuration-value {
                   case descriptor-value {
                       uses fpcbase:fpc-descriptor-value;
                       description "Descriptor Value";
                   }
                   case action-value {
                       uses fpcbase:fpc-action-value;
                       description "Action Value";
                   }
                   case setting-value {
                       uses fpc:fpc-setting-value;
                       description "Setting";
                   }
                   description "Policy Attributes";
               }
               description "Policy Configuration Value Choice";
       }
       grouping policy-configuration {
           list policy-configuration {
               key index;
               uses fpc:index;
               uses fpc:policy-configuration-choice;
               description "Policy Configuration";
           }
           description "Policy Configuration Value";
       }
       grouping ref-configuration {
           uses fpc:policy-template-key;
           uses fpc:policy-configuration;
           uses fpc:templatedef;
           description "Policy-Configuration Entry";
       }

       // FPC Policy
       grouping policy-information-model {
         list action-template {
           key action-template-key;
           uses fpc:action-template-key;
           uses fpcbase:fpc-action-value;
           uses fpc:templatedef;
           description "Action Template";
         }
         list descriptor-template {
           key descriptor-template-key;
           uses fpc:descriptor-template-key;
           uses fpcbase:fpc-descriptor-value;
           uses fpc:templatedef;
           description "Descriptor Template";
         }
         list rule-template {
           key rule-template-key;
           uses fpc:rule-template-key;
           leaf descriptor-match-type {
               type enumeration {
                   enum or {
                       value 0;
                       description "OR logic";
                   }
                   enum and {
                       value 1;
                       description "AND logic";
                   }
               }
               mandatory true;
               description "Type of Match (OR or AND) applied
                   to the descriptor-configurations";
           }
           list descriptor-configuration {
               key "descriptor-template-key";
               uses fpc:descriptor-template-key;
               leaf direction {
                   type rfc5777:direction-type;
                   description "Direction";
               }
               list attribute-expression {
                   key index;
                   uses fpc:index;
                   uses fpcbase:fpc-descriptor-value;
                   description "Descriptor Attributes";
               }
               uses fpc:fpc-setting-value;
               description "A set of Descriptor references";
           }
           list action-configuration {
               key "action-order";
               leaf action-order {
                   type uint32;
                   mandatory true;
                   description "Action Execution Order";
               }
               uses fpc:action-template-key;
               list attribute-expression {
                   key index;
                   uses fpc:index;
                   uses fpcbase:fpc-action-value;
                   description "Action Attributes";
               }
               uses fpc:fpc-setting-value;
               description "A set of Action references";
           }
           uses fpc:templatedef;
           list rule-configuration {
               key index;
               uses fpc:index;
               uses fpc:policy-configuration-choice;
               description "Rule Configuration";
           }
           description "Rule Template";
         }
         list policy-template {
           key policy-template-key;
           uses fpc:policy-template-key;
           list rule-template {
               key "precedence";
               unique "rule-template-key";
               leaf precedence {
                   type uint32;
                   mandatory true;
                   description "Rule Precedence";
               }
               uses fpc:rule-template-key;
               description "Rule Entry";
           }
           uses fpc:templatedef;
           uses fpc:policy-configuration;
           description "Policy Template";
         }
         description "FPC Policy Structures";
       }

       // Topology Information Model
       identity role {
           description "Role";
       }
       grouping dpn-key {
           leaf dpn-key {
               type fpc:fpc-identity;
               description "DPN Key";
           }
           description "DPN Key";
       }
       grouping role-key {
           leaf role-key {
               type identityref {
                   base "fpc:role";
               }
               mandatory true;
               description "Access Technology Role";
           }
           description "Access Technology Role key";
       }
       grouping interface-key {
           leaf interface-key{
               type fpc:fpc-identity;
               mandatory true;
               description "interface identifier";
           }
           description "Interface Identifier key";
       }
       identity interface-protocols {
           description "Protocol supported by the interface";
       }
       identity features {
           description "Protocol features";
       }

     // Mobility Context
     grouping mobility-context {
       leaf mobility-context-key {
         type fpc:fpc-identity;
         mandatory true;
         description "Mobility Context Key";
       }
       leaf-list delegating-ip-prefix {
           type inet:ip-prefix;
           description "IP Prefix";
       }
       leaf parent-context {
         type fpc:fpc-identity;
         description "Parent Mobility Context";
       }
       leaf-list child-context {
           type fpc:fpc-identity;
           description "Child Mobility Context";
       }
       container mobile-node {
           leaf-list ip-address {
               type inet:ip-address;
               description "IP Address";
           }
           leaf imsi {
               type fpcbase:imsi-type;
               description "IMSI";
           }
           list mn-policy-configuration {
               key policy-template-key;
               uses fpc:ref-configuration;
               description "MN Policy Configuration";
           }
           description "Mobile Node";
       }
       container domain {
           leaf domain-key {
               type fpc:fpc-identity;
               description "Domain Key";
           }
           list domain-policy-settings {
               key policy-template-key;
               uses fpc:ref-configuration;
               description "MN Policy Configuration";
           }
           description "Domain";
       }
       list dpn {
           key dpn-key;
           uses fpc:dpn-key;
           list dpn-policy-configuration {
               key policy-template-key;
               uses fpc:ref-configuration;
               description "DPN Policy Configuration";
           }
           leaf role {
               type identityref {
                   base "fpc:role";
               }
               description "Role";
           }
           list service-data-flow {
               key identifier;
               leaf identifier {
                   type uint32;
                   description "Generic Identifier";
               }
               leaf service-group-key {
                   type fpc:fpc-identity;
                   description "Service Group Key";
               }
               list interface {
                   key interface-key;
                   uses fpc:interface-key;
                   description "interface assigned";
               }
               list service-data-flow-policy-configuration {
                   key policy-template-key;
                   uses fpc:ref-configuration;
                   description "Flow Policy Configuration";
               }
               description "Service Dataflow";
           }
           description "DPN";
       }
       description "Mobility Context";
     }

     // Events, Probes & Notifications
     identity event-type {
       description "Base Event Type";
     }
     typedef event-type-id {
       type uint32;
       description "Event ID Type";
     }
     grouping monitor-key {
       leaf monitor-key {
         type fpc:fpc-identity;
         mandatory true;
         description "Monitor Key";
       }
       description "Monitor Id";
     }
     grouping monitor-config {
       uses fpc:templatedef;
       uses fpc:monitor-key;
       leaf target {
         type string;
         description "target";
       }
       leaf deferrable {
         type boolean;
         description "Indicates reports related to this
           config can be delayed.";
       }
       choice configuration {
         mandatory true;
         leaf period {
             type uint32;
             description "Period";
         }
         case threshold-config {
           leaf low {
             type uint32;
             description "low threshold";
           }
           leaf hi {
             type uint32;
             description "high threshold";
           }
           description "Threshold Config Case";
         }
         leaf schedule {
             type uint32;
             description "Reporting Time";
         }
         leaf-list event-identities {
             type identityref {
               base "fpc:event-type";
             }
             description "Event Identities";
         }
         leaf-list event-ids {
             type uint32;
               description "Event IDs";
         }
         description "Event Config Value";
       }
       description "Monitor Configuration";
     }

     // Top Level Structures
     list tenant {
      key "tenant-key";
      leaf tenant-key {
           type fpc:fpc-identity;
           description "Tenant Key";
      }
      container topology-information-model {
        config false;
        list service-group {
           key "service-group-key role-key";
           leaf service-group-key {
               type fpc:fpc-identity;
               mandatory true;
               description "Service Group Key";
           }
           leaf service-group-name {
               type string;
               description "Service Group Name";
           }
           uses fpc:role-key;
           leaf role-name {
               type string;
               mandatory true;
               description "Role Name";
           }
           leaf-list protocol {
             type identityref {
               base "interface-protocols";
             }
             min-elements 1;
             description "Supported protocols";
           }
           leaf-list feature {
               type identityref {
                   base "interface-protocols";
               }
               description "Supported features";
           }
           list service-group-configuration {
               key index;
               uses fpc:index;
               uses fpc:policy-configuration-choice;
               description "Settings";
           }
           list dpn {
               key dpn-key;
               uses fpc:dpn-key;
               min-elements 1;
               list referenced-interface {
                   key interface-key;
                   uses fpc:interface-key;
                   leaf-list peer-service-group-key {
                       type fpc:fpc-identity;
                       description "Peer Service Group";
                   }
                   description "Referenced Interface";
               }
               description "DPN";
           }
           description "Service Group";
       }
       list dpn {
         key dpn-key;
         uses fpc:dpn-key;
         leaf dpn-name {
           type string;
           description "DPN name";
         }
         leaf dpn-resource-mapping-reference {
           type string;
           description "Reference to underlying DPN resource(s)";
         }
         leaf domain-key {
           type fpc:fpc-identity;
           description "Domains";
         }
         leaf-list service-group-key {
           type fpc:fpc-identity;
           description "Service Group";
         }
         list interface {
           key "interface-key";
           uses fpc:interface-key;
           leaf interface-name {
               type string;
               description "Service Endpoint Interface Name";
           }
           leaf role {
               type identityref {
                   base "fpc:role";
               }
               description "Roles supported";
           }
           leaf-list protocol {
             type identityref {
               base "interface-protocols";
             }
             description "Supported protocols";
           }
           list interface-configuration {
            key index;
            uses fpc:index;
            uses fpc:policy-configuration-choice;
            description "Interface settings";
          }
          description "DPN interfaces";
         }
         list dpn-policy-configuration {
           key policy-template-key;
           uses fpc:ref-configuration;
           description "DPN Policy Configuration";
         }
         description "Set of DPNs";
       }
       list domain {
         key domain-key;
         leaf domain-key {
           type fpc:fpc-identity;
           mandatory true;
           description "Domain Key";
         }
         leaf domain-name {
           type string;
           description "Domain displayname";
         }
         list domain-policy-configuration {
           key policy-template-key;
           uses fpc:ref-configuration;
           description "Domain Configuration";
         }
         description "List of Domains";
       }
       container dpn-checkpoint {
           uses fpc:basename-info;
           description "DPN Checkpoint information";
       }
       container service-group-checkpoint {
           uses fpc:basename-info;
           description "Service Group Checkpoint information";
       }
       container domain-checkpoint {
           uses fpc:basename-info;
           description "Domain Checkpoint information";
       }
       description "FPC Topology grouping";
      }
      container policy-information-model {
           config false;
           uses fpc:policy-information-model;
           uses fpc:basename-info;
           description "Policy";
      }
      list mobility-context {
           key "mobility-context-key";
           config false;
           uses fpc:mobility-context;
           description "Mobility Context";
      }
      list monitor {
           key monitor-key;
           config false;
           uses fpc:monitor-config;
           description "Monitor";
      }
      description "Tenant";
     }

       typedef agent-identifier {
           type fpc:fpc-identity;
           description "Agent Identifier";
       }
       typedef client-identifier {
           type fpc:fpc-identity;
           description "Client Identifier";
       }
       grouping basename-info {
             leaf basename {
               type fpc:fpc-identity;
               description "Rules Basename";
             }
             leaf base-checkpoint {
               type string;
               description "Checkpoint";
             }
             description "Basename Information";
       }

       // RPCs
       grouping client-id {
           leaf client-id {
               type fpc:client-identifier;
               mandatory true;
               description "Client Id";
           }
           description "Client Identifier";
       }
       grouping execution-delay {
           leaf execution-delay {
               type uint32;
               description "Execution Delay (ms)";
           }
           description "Execution Delay";
       }
       typedef ref-scope {
         type enumeration {
           enum none {
             value 0;
             description "no references";
           }
           enum op {
             value 1;
             description "All references are intra-operation";
           }
           enum bundle {
             value 2;
             description "All references in exist in bundle";
           }
           enum storage {
             value 3;
             description "One or more references exist in storage.";
           }
           enum unknown {
             value 4;
             description "The location of the references are unknown.";
           }
         }
         description "Search scope for references in the operation.";
       }
       rpc configure {
           description "Configure RPC";
           input {
               uses client-id;
               uses execution-delay;
               uses ypatch:yang-patch;
           }
           output {
               uses ypatch:yang-patch-status;
           }
       }
       augment "/configure/input/yang-patch/edit" {
           leaf reference-scope {
               type fpc:ref-scope;
               description "Reference Scope";
           }
           uses fpcbase:instructions;
           description "yang-patch edit augments for configure rpc";
       }
       grouping subsequent-edits {
            list subsequent-edit {
              key edit-id;
              ordered-by user;

              description "Edit list";
              leaf edit-id {
                type string;
                description "Arbitrary string index for the edit.";
              }

              leaf operation {
                type enumeration {
                  enum create {
                    description "Create";
                  }
                  enum delete {
                    description "Delete";
                  }
                  enum insert {
                    description "Insert";
                  }
                  enum merge {
                    description "Merge";
                  }
                  enum move {
                    description "Move";
                  }
                  enum replace {
                    description "Replace";
                  }
                  enum remove {
                    description
                      "Delete the target node if it currently exists.";
                  }
                }
                mandatory true;
                description
                  "The datastore operation requested";
              }

              leaf target {
                type ypatch:target-resource-offset;
                mandatory true;
                description
                  "Identifies the target data node";
              }

              leaf point {
               when "(../operation = 'insert' or ../operation = 'move')"
                + "and (../where = 'before' or ../where = 'after')" {
                  description
                    "This leaf only applies for 'insert' or 'move'
                     operations, before or after an existing entry.";
                }
                type ypatch:target-resource-offset;
                description
                  "The absolute URL path for the data node";
              }

              leaf where {
               when "../operation = 'insert' or ../operation = 'move'" {
                  description
                    "This leaf only applies for 'insert' or 'move'
                     operations.";
                }
                type enumeration {
                  enum before {
                    description
                      "Insert or move a data node before.";
                  }
                  enum after {
                    description
                      "Insert or move a data node after.";
                  }
                 enum first {
                    description
                      "Insert or move a data node so it becomes ordered
                       as the first entry.";
                  }
                  enum last {
                    description
                      "Insert or move a data node so it becomes ordered
                       as the last entry.";
                  }
                }
                default last;
                description
                  "Identifies where a data resource will be inserted
                   or moved.";
              }

              anydata value {
                when "../operation = 'create' "
                   + "or ../operation = 'merge' "
                   + "or ../operation = 'replace' "
                   + "or ../operation = 'insert'" {
                  description
                    "The anydata 'value' is only used for 'create',
                     'merge', 'replace', and 'insert' operations.";
                }
                description
```

```
                  "Value used for this edit operation.";
              }
            }
            description "Subsequent Edits";
       }
       augment "/configure/output/yang-patch-status/edit-status/edit/"
           + "edit-status-choice/ok" {
           leaf notify-follows {
               type boolean;
               description "Notify Follows Indication";
           }
           uses fpc:subsequent-edits;
           description "Configure output augments";
       }

       grouping op-header {
           uses client-id;
           uses execution-delay;
           leaf operation-id {
               type uint64;
               mandatory true;
               description "Operation Identifier";
           }
           description "Common Operation header";
       }
       grouping monitor-response {
           leaf operation-id {
               type uint64;
               mandatory true;
               description "Operation Identifier";
           }
           choice edit-status-choice {
                  description
                    "A choice between different types of status
                     responses for each 'edit' entry.";
                  leaf ok {
                    type empty;
                    description
                      "This 'edit' entry was invoked without any
                       errors detected by the server associated
                       with this edit.";
                  }
                  case errors {
                    uses rc:errors;
                    description
                      "The server detected errors associated with the
                       edit identified by the same 'edit-id' value.";
                  }
           }
           description "Monitor Response";
       }

     // Common RPCs
     rpc register_monitor {
       description "Used to register monitoring of parameters/events";
       input {
         uses fpc:op-header;
         list monitor {
           key monitor-key;
           uses fpc:monitor-config;
           description "Monitor Configuration";
         }
       }
       output {
           uses fpc:monitor-response;
       }
     }
     rpc deregister_monitor {
       description "Used to de-register monitoring of
         parameters/events";
       input {
         uses fpc:op-header;
         list monitor {
           key monitor-key;
           uses fpc:monitor-key;
           min-elements 1;
           leaf send_data {
               type boolean;
               description "Indicates if NOTIFY with final data
                   is desired upon deregistration";
           }
           description "Monitor Identifier";
         }
       }
       output {
           uses fpc:monitor-response;
       }
     }
     rpc probe {
       description "Probe the status of a registered monitor";
       input {
         uses fpc:op-header;
         list monitor {
           key monitor-key;
           uses fpc:monitor-key;
           min-elements 1;
```

```
              description "Monitor";
            }
          }
          output {
              uses fpc:monitor-response;
          }
        }

        // Notification Messages & Structures
        notification config-result-notification {
          uses ypatch:yang-patch-status;
          description "Configuration Result Notification";
        }
        augment "/config-result-notification" {
              uses fpc:subsequent-edits;
              description "config-result-notificatio augment";
        }

        identity notification-cause {
          description "Notification Cause";
        }
        identity subscribed-event-occurred {
          base "notification-cause";
          description "Subscribed Event Occurence";
        }
        identity low-threshold-crossed {
          base "notification-cause";
          description "Subscribed Event Occurence";
        }
        identity high-threshold-crossed {
          base "notification-cause";
          description "Subscribed Event Occurence";
        }
        identity periodic-report {
          base "notification-cause";
          description "Periodic Report";
        }
        identity scheduled-report {
          base "notification-cause";
          description "Scheduled Report";
        }
        identity probe {
          base "notification-cause";
          description "Probe";
        }
        identity deregistration-final-value {
          base "notification-cause";
          description "Probe";
```

```
     }
     identity monitoring-suspension {
       base "notification-cause";
       description "Indicates monitoring suspension";
     }
     identity monitoring-resumption {
       base "notification-cause";
       description "Indicates that monitoring has resumed";
     }
     identity dpn-available {
       base "notification-cause";
       description "DPN Candidate Available";
     }
     identity dpn-unavailable {
       base "notification-cause";
       description "DPN Unavailable";
     }
     notification notify {
       leaf notification-id {
         type uint32;
         description "Notification Identifier";
       }
       leaf timestamp {
         type uint32;
         description "timestamp";
       }
       list report {
           key monitor-key;
           uses fpc:monitor-key;
           min-elements 1;
           leaf trigger {
               type identityref {
                   base "notification-cause";
               }
               description "Notification Cause";
           }
           choice value {
               case dpn-candidate-available {
                   leaf node-id {
                       type inet:uri;
                       description "Topology URI";
                   }
                   list supported-interface-list {
                       key role-key;
                       uses fpc:role-key;
                       description "Support Intefaces";
                   }
                   description "DPN Candidate Information";
               }
               case dpn-unavailable {
                   leaf dpn-id {
                     type fpc:fpc-identity;
                     description "DPN Identifier for DPN Unavailable";
                   }
                   description "DPN Unavailable";
               }
               anydata report-value {
                   description "Any non integer report";
               }
               description "Report Value";
           }
           description "Report";
       }
       description "Notify Message";
     }
   }
   <CODE ENDS>
```

A.2. FPC YANG Settings and Extensions Model

This module defines the base data elements in FPC that are likely to be extended.

This module references [RFC6991], ietf-trafficselector-types and ietf-pmip-qos modules.

```
   <CODE BEGINS> file "ietf-dmm-fpc-settingsext@2018-05-17.yang"
   module ietf-dmm-fpc-settingsext {
     yang-version 1.1;
     namespace "urn:ietf:params:xml:ns:yang:ietf-dmm-fpc-settingsext";
     prefix fpcbase;

       import ietf-inet-types { prefix inet;
           revision-date 2013-07-15; }
       import ietf-trafficselector-types { prefix traffic-selectors;
           revision-date 2018-05-17; }
       import ietf-yang-types { prefix ytypes;
           revision-date 2013-07-15; }
       import ietf-pmip-qos { prefix pmipqos;
           revision-date 2018-05-17; }
       import ietf-diam-trafficclassifier { prefix rfc5777;
           revision-date 2018-05-17; }

       organization "IETF Distributed Mobility Management (DMM)
         Working Group";
       contact
          "WG Web:   <http://tools.ietf.org/wg/netmod/>
           WG List:  <mailto:netmod@ietf.org>

           WG Chair: Dapeng Liu
                     <mailto:maxpassion@gmail.com>

           WG Chair: Sri Gundavelli
                     <mailto:sgundave@cisco.com>

           Editor:   Satoru Matsushima
                     <mailto:satoru.matsushima@g.softbank.co.jp>

           Editor:   Lyle Bertz
                     <mailto:lylebe551144@gmail.com>";

       description
       "This module contains YANG definition for
        Forwarding Policy Configuration Protocol(FPCP).

         It contains Settings defintions as well as Descriptor and
         Action extensions.

        Copyright (c) 2016 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.";

   revision 2018-05-17 {
       description "Initial Revision.";
       reference "draft-ietf-dmm-fpc-cpdp-10";
   }

       //Tunnel Information
        identity tunnel-type {
             description "Tunnel Type";
         }
         identity grev1 {
             base "fpcbase:tunnel-type";
             description "GRE v1";
         }
         identity grev2 {
             base "fpcbase:tunnel-type";
             description "GRE v2";
         }
         identity ipinip {
             base "fpcbase:tunnel-type";
             description "IP in IP";
         }
         identity gtpv1 {
             base "fpcbase:tunnel-type";
             description "GTP version 1 Tunnel";
         }
         identity gtpv2 {
             base "fpcbase:tunnel-type";
             description "GTP version 2 Tunnel";
         }

       grouping tunnel-value {
         container tunnel-info {
           leaf tunnel-local-address {
               type inet:ip-address;
               description "local tunnel address";
           }
           leaf tunnel-remote-address {
               type inet:ip-address;
               description "remote tunnel address";
           }
           leaf mtu-size {
               type uint32;
               description "MTU size";
           }
           leaf tunnel {
               type identityref {
                   base "fpcbase:tunnel-type";
               }
           description "tunnel type";
           }
           leaf payload-type {
               type enumeration {
                   enum ipv4 {
                       value 0;
                       description "IPv4";
                   }
                   enum ipv6 {
                       value 1;
                       description "IPv6";
                   }
                   enum dual {
                       value 2;
                       description "IPv4 and IPv6";
                   }
               }
               description "Payload Type";
           }
           leaf gre-key {
               type uint32;
               description "GRE_KEY";
           }
           container gtp-tunnel-info {
               leaf local-tunnel-identifier {
                   type uint32;
                   description "Tunnel Endpoint IDentifier (TEID)";
               }
               leaf remote-tunnel-identifier {
                   type uint32;
                   description "Tunnel Endpoint IDentifier (TEID)";
               }
               leaf sequence-numbers-enabled {
                   type boolean;
                   description "Sequence No. Enabled";
               }
               description "GTP Tunnel Information";
           }
           leaf ebi {
               type fpcbase:ebi-type;
               description "EPS Bearier Identifier";
           }
           leaf lbi {
               type fpcbase:ebi-type;
               description "Linked Bearier Identifier";
           }
           description "Tunnel Information";
         }
         description "Tunnel Value";
       }

     //////////////////////////////
     // DESCRIPTOR DEFINITIONS

       // From 3GPP TS 24.008 version 13.5.0 Release 13
         typedef packet-filter-direction {
             type enumeration {
               enum preRel7Tft {
                 value 0;
                 description "Pre-Release 7 TFT";
               }
               enum uplink {
                 value 1;
                 description "uplink";
               }
               enum downlink {
                 value 2;
                 description "downlink";
               }
               enum bidirectional {
                 value 3;
                 description "bi-direcitonal";
               }
             }
             description "Packet Filter Direction";
         }
         typedef component-type-id {
             type uint8 {
               range "16 | 17 | 32 | 33 | 35 | 48 | 64 | 65 |"
               + " 80 | 81 | 96 | 112 | 128";
             }
             description "Specifies the Component Type";
         }
         grouping packet-filter {
           leaf direction {
               type fpcbase:packet-filter-direction;
               description "Filter Direction";
           }
           leaf identifier {
               type uint8 {
                 range "1..15";
               }
               description "Filter Identifier";
           }
           leaf evaluation-precedence {
               type uint8;
               description "Evaluation Precedence";
           }
           list contents {
             key component-type-identifier;
             description "Filter Contents";
             leaf component-type-identifier {
                 type fpcbase:component-type-id;
                 description "Component Type";
             }
             choice value {
               leaf ipv4-local {
                   type inet:ipv4-address;
                   description "IPv4 Local Address";
               }
               leaf ipv6-prefix-local {
                   type inet:ipv6-prefix;
                   description "IPv6 Local Prefix";
               }
               leaf ipv4-ipv6-remote {
                   type inet:ip-address;
                   description "Ipv4 Ipv6 remote address";
               }
               leaf ipv6-prefix-remote {
                   type inet:ipv6-prefix;
                   description "IPv6 Remote Prefix";
               }
               leaf next-header {
                   type uint8;
                   description "Next Header";
               }
               leaf local-port {
                   type inet:port-number;
                   description "Local Port";
               }
               case local-port-range {
                 leaf local-port-lo {
                   type inet:port-number;
                   description "Local Port Min Value";
                 }
                 leaf local-port-hi {
                   type inet:port-number;
                   description "Local Port Max Value";
                 }
               }
               leaf remote-port {
                   type inet:port-number;
                   description "Remote Port";
               }
               case remote-port-range {
                 leaf remote-port-lo {
                   type inet:port-number;
                   description "Remote Por Min Value";
                 }
                 leaf remote-port-hi {
                   type inet:port-number;
                   description "Remote Port Max Value";
                 }
               }
               leaf ipsec-index {
                   type traffic-selectors:ipsec-spi;
                   description "IPSec Index";
               }
               leaf traffic-class {
                   type inet:dscp;
                   description "Traffic Class";
               }
               case traffic-class-range {
                   leaf traffic-class-lo {
                     type inet:dscp;
                     description "Traffic Class Min Value";
                   }
                   leaf traffic-class-hi {
                     type inet:dscp;
                     description "Traffic Class Max Value";
                   }
               }
               leaf-list flow-label {
                   type inet:ipv6-flow-label;
                   description "Flow Label";
               }
               description "Component Value";
             }
           }
           description "Packet Filter";
         }

       grouping prefix-descriptor {
           leaf destination-ip {
             type inet:ip-prefix;
             description "Rule of destination IP";
           }
           leaf source-ip {
             type inet:ip-prefix;
             description "Rule of source IP";
           }
           description "Traffic descriptor based upon source/
             destination as IP prefixes";
       }

     grouping fpc-descriptor-value {
       choice descriptor-value {
         mandatory true;
         leaf all-traffic {
             type empty;
             description "admit any";
         }
         leaf no-traffic {
             type empty;
```

```
                description "deny any";
             }
             case prefix-descriptor {
               uses fpcbase:prefix-descriptor;
               description "IP Prefix descriptor";
             }
             case pmip-selector {
               uses traffic-selectors:traffic-selector;
               description "PMIP Selector";
             }
             container rfc5777-classifier-template {
                   uses rfc5777:classifier;
                   description "RFC 5777 Classifier";
             }
             container packet-filter {
                   uses fpcbase:packet-filter;
                   description "Packet Filter";
             }
             case tunnel-info {
               uses fpcbase:tunnel-value;
               description "Tunnel Descriptor (only
                   considers source info)";
             }
             description "Descriptor Value";
           }
           description "FPC Descriptor Values";
         }

           // Next Hop Structures
           typedef fpc-service-path-id {
               type uint32 {
                   range "0..33554431";
               }
               description "SERVICE_PATH_ID";
           }
           typedef fpc-mpls-label {
               type uint32 {
                 range "0..1048575";
               }
               description "MPLS label";
           }
           typedef segment-id {
               type string {
                   length "16";
               }
               description "SR Segement Identifier";
           }
           grouping fpc-nexthop {
```

```
           choice next-hop-value {
               leaf ip-address {
                   type inet:ip-address;
                   description "IP Value";
               }
               leaf mac-address {
                   type ytypes:mac-address;
                   description "MAC Address Value";
               }
               leaf service-path {
                   type fpcbase:fpc-service-path-id;
                   description "Service Path Value";
               }
               leaf mpls-path {
                   type fpcbase:fpc-mpls-label;
                   description "MPLS Value";
               }
               leaf nsh {
                   type string {
                       length "16";
                   }
                   description "Network Service Header";
               }
               leaf interface {
                   type uint16;
                   description "If (interface) Value";
               }
               leaf segment-identifier {
                   type fpcbase:segment-id;
                   description "Segment Id";
               }
               leaf-list mpls-label-stack {
                   type fpcbase:fpc-mpls-label;
                   description "MPLS Stack";
               }
               leaf-list mpls-sr-stack {
                   type fpcbase:fpc-mpls-label;
                   description "MPLS SR Stack";
               }
               leaf-list srv6-stack {
                   type fpcbase:segment-id;
                   description "Segment Id";
               }
               case tunnel-info {
                   uses fpcbase:tunnel-value;
                   description "Tunnel Descriptor (only
                   considers source info)";
               }
               description "Value";
           }
           description "Nexthop Value";
       }

       //////////////////////////////
       // PMIP Integration         //
         typedef pmip-commandset {
               type bits {
                   bit assign-ip {
                     position 0;
                     description "Assign IP";
                   }
                   bit assign-dpn {
                     position 1;
                     description "Assign DPN";
                   }
                   bit session {
                     position 2;
                     description "Session Level";
                   }
                   bit uplink {
                     position 3;
                     description "Uplink";
                   }
                   bit downlink {
                     position 4;
                     description "Downlink";
                   }
               }
               description "PMIP Instructions";
           }
       ///////////////////////////////
       // 3GPP Integration         //

         // Type Defs
         typedef fpc-qos-class-identifier {
             type uint8 {
                 range "1..9";
             }
             description "QoS Class Identifier (QCI)";
         }
         typedef ebi-type {
           type uint8 {
             range "0..15";
           }
           description "EUTRAN Bearere Identifier (EBI) Type";
         }
         typedef imsi-type {
             type uint64;
             description
                 "International Mobile Subscriber Identity (IMSI)
                   Value Type";
         }
         // Instructions
         typedef threegpp-instr {
           type bits {
             bit assign-ip {
               position 0;
               description "Assign IP Address/Prefix";
             }
             bit assign-fteid-ip {
               position 1;
               description "Assign FTEID-IP";
             }
             bit assign-fteid-teid {
               position 2;
               description "Assign FTEID-TEID";
             }
             bit session {
               position 3;
               description "Commands apply to the Session Level";
             }
             bit uplink {
               position 4;
               description "Commands apply to the Uplink";
             }
             bit downlink {
               position 5;
               description "Commands apply to the Downlink";
             }
             bit assign-dpn {
               position 6;
               description "Assign DPN";
             }
           }
           description "Instruction Set for 3GPP R11";
         }

       //////////////////////////////
       // ACTION VALUE AUGMENTS
       grouping fpc-action-value {
           choice action-value {
               mandatory true;
               leaf drop {
                   type empty;
                   description "Drop Traffic";
               }
               container rewrite {
                   choice rewrite-value {
                       case prefix-descriptor {
                           uses fpcbase:prefix-descriptor;
                           description "IP Prefix descriptor";
                       }
                       case pmip-selector {
                           uses traffic-selectors:traffic-selector;
                           description "PMIP Selector";
                       }
                       container rfc5777-classifier-template {
                           uses rfc5777:classifier;
                           description "RFC 5777 Classifier";
                       }
                       description "Rewrite Choice";
                   }
                   description "Rewrite/NAT value";
               }
               container copy-forward-nexthop {
                       uses fpcbase:fpc-nexthop;
                       description "Copy Forward Value";
               }
               container nexthop {
                       uses fpcbase:fpc-nexthop;
                       description "NextHop Value";
               }
               case qos {
                   leaf trafficclass {
                       type inet:dscp;
                       description "Traffic Class";
                   }
                   uses pmipqos:qosattribute;
                   leaf qci {
                       type fpcbase:fpc-qos-class-identifier;
                       description "QCI";
                   }
                   leaf ue-agg-max-bitrate {
                       type uint32;
                       description "UE Aggregate Max Bitrate";
                   }
                   leaf apn-ambr {
                       type uint32;
                       description
                        "Access Point Name Aggregate Max Bit Rate";
                   }
                   description "QoS Attributes";
               }
               description "Action Value";
           }
           description "FPC Action Value";
       }

     // Instructions
     grouping instructions {
       container command-set {
         choice instr-type {
           leaf instr-3gpp-mob {
               type fpcbase:threegpp-instr;
               description "3GPP GTP Mobility Instructions";
           }
           leaf instr-pmip {
               type pmip-commandset;
               description "PMIP Instructions";
           }
           description "Instruction Value Choice";
         }
         description "Instructions";
       }
       description "Instructions Value";
     }
   }
   <CODE ENDS>
```

A.3. PMIP QoS Model

This module defines the base protocol elements specified in this document.

This module references [RFC6991].

```
   <CODE BEGINS> file "ietf-pmip-qos@2018-05-17.yang"
   module ietf-pmip-qos {
       yang-version 1.1;

       namespace
         "urn:ietf:params:xml:ns:yang:ietf-pmip-qos";

       prefix "qos-pmip";

       import ietf-inet-types {
         prefix inet;
         revision-date 2013-07-15;
       }
       import ietf-trafficselector-types { prefix traffic-selectors;
```

```
           revision-date 2018-05-17; }

       organization "IETF Distributed Mobility Management (DMM)
         Working Group";

       contact
          "WG Web:   <http://tools.ietf.org/wg/netmod/>
           WG List:  <mailto:netmod@ietf.org>

           WG Chair: Dapeng Liu
                     <mailto:maxpassion@gmail.com>

           WG Chair: Sri Gundavelli
                     <mailto:sgundave@cisco.com>

           Editor:   Satoru Matsushima
                     <mailto:satoru.matsushima@g.softbank.co.jp>

           Editor:   Lyle Bertz
                     <mailto:lylebe551144@gmail.com>";

       description
         "This module contains a collection of YANG definitions for
        quality of service paramaters used in Proxy Mobile IPv6.

        Copyright (c) 2016 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.";

     revision 2018-05-17 {
       description "Initial Revision.";
       reference "RFC 6088: Traffic Selectors for Flow Bindings";
     }

       // Type Definitions

       // QoS Option Field Type Definitions
     typedef sr-id {
       type uint8;
```

```
         description
          "An 8-bit unsigned integer used for identifying the QoS
           Service Request.";
       }

       typedef traffic-class {
         type inet:dscp;
         description
           "Traffic Class consists of a 6-bit DSCP field followed by a
            2-bit reserved field.";
        reference
            "RFC 3289: Management Information Base for the
               Differentiated Services Architecture
             RFC 2474: Definition of the Differentiated Services Field
                       (DS Field) in the IPv4 and IPv6 Headers
             RFC 2780: IANA Allocation Guidelines For Values In
                       the Internet Protocol and Related Headers";
       }

       typedef operational-code {
         type enumeration {
           enum RESPONSE {
             value 0;
             description "Response to a QoS request";
           }
           enum ALLOCATE {
             value 1;
             description "Request to allocate QoS resources";
           }
           enum DE-ALLOCATE {
             value 2;
             description "Request to de-Allocate QoS resources";
           }
           enum MODIFY {
             value 3;
             description "Request to modify QoS parameters for a
                 previously negotiated QoS Service Request";
           }
           enum QUERY {
             value 4;
             description "Query to list the previously negotiated QoS
                 Service Requests that are still active";
           }
           enum NEGOTIATE {
             value 5;
             description "Response to a QoS Service Request with a
               counter QoS proposal";
           }
         }
         description
          "The type of QoS request. Reserved values:   (6) to (255)
                   Currently not used.  Receiver MUST ignore the option
                   received with any value in this range.";
       }

     //Value definitions
     typedef Per-MN-Agg-Max-DL-Bit-Rate-Value {
         type uint32;
         description
             "The aggregate maximum downlink bit rate that is
             requested/allocated for all the mobile node’s IP flows.
             The measurement units are bits per second.";
     }

      typedef Per-MN-Agg-Max-UL-Bit-Rate-Value {
         type uint32;
         description
           "The aggregate maximum uplink bit rate that is
               requested/allocated for the mobile node’s IP flows.  The
               measurement units are bits per second.";
      }

      // Generic Structure for the uplink and downlink
      grouping Per-Session-Agg-Max-Bit-Rate-Value {
        leaf max-rate {
          type uint32;
          mandatory true;
          description
          "The aggregate maximum bit rate that is requested/allocated
        for all the IP flows associated with that mobility session.
        The measurement units are bits per second.";
        }
        leaf service-flag {
         type boolean;
         mandatory true;
         description
          "This flag is used for extending the scope of the
           target flows for Per-Session-Agg-Max-UL/DL-Bit-Rate
           from(UL)/to(DL) the mobile node’s other mobility sessions
           sharing the same Service Identifier.";
         reference
           "RFC 5149 - Service Selection mobility option";
        }
        leaf exclude-flag {
          type boolean;
          mandatory true;
          description
           "This flag is used to request that the uplink/downlink
          flows for which the network is providing
               Guaranteed-Bit-Rate service be excluded from the
               target IP flows for which
               Per-Session-Agg-Max-UL/DL-Bit-Rate is measured.";
        }
       description "Per-Session-Agg-Max-Bit-Rate Value";
      }

      grouping Allocation-Retention-Priority-Value {
        leaf priority-level {
          type uint8 {
            range "0..15";
          }
          mandatory true;
          description
          "This is a 4-bit unsigned integer value. It is used to decide
           whether a mobility session establishment or modification
           request can be accepted; this is typically used for
           admission control of Guaranteed Bit Rate traffic in case of
           resource limitations.";
        }
        leaf preemption-capability {
          type enumeration {
           enum enabled {
             value 0;
             description "enabled";
           }
           enum disabled {
             value 1;
             description "disabled";
           }
           enum reserved1 {
             value 2;
             description "reserved1";
           }
           enum reserved2 {
             value 3;
             description "reserved2";
           }
          }
          mandatory true;
          description
          "This is a 2-bit unsigned integer value.  It defines whether a
           service data flow can get resources tha were already
           assigned to another service data flow with a lower priority
           level.";
        }
        leaf preemption-vulnerability {
          type enumeration {
           enum enabled {
             value 0;
             description "enabled";
           }
           enum disabled {
             value 1;
             description "disabled";
           }
           enum reserved1 {
             value 2;
             description "reserved1";
           }
           enum reserved2 {
             value 3;
             description "reserved2";
           }
          }
          mandatory true;
          description
          "This is a 2-bit unsigned integer value.  It defines whether a
            service data flow can lose the resources assigned to it in
            order to admit a service data flow with a higher priority
            level.";
        }
       description "Allocation-Retention-Priority Value";
      }

      typedef Aggregate-Max-DL-Bit-Rate-Value {
         type uint32;
         description
           "The aggregate maximum downlink bit rate that is
            requested/allocated for downlink IP flows.  The measurement
            units are bits per second.";
      }

       typedef Aggregate-Max-UL-Bit-Rate-Value {
         type uint32;
         description
           "The aggregate maximum downlink bit rate that is
            requested/allocated for downlink IP flows.  The measurement
            units are bits per second.";
       }

       typedef Guaranteed-DL-Bit-Rate-Value {
         type uint32;
         description
         "The guaranteed bandwidth in bits per second for downlink
           IP flows.  The measurement units are bits per second.";
       }

       typedef Guaranteed-UL-Bit-Rate-Value {
         type uint32;
         description
           "The guaranteed bandwidth in bits per second for uplink
            IP flows.  The measurement units are bits per second.";
       }

       grouping QoS-Vendor-Specific-Attribute-Value-Base {
         leaf vendorid {
           type uint32;
           mandatory true;
           description
            "The Vendor ID is the SMI (Structure of Management
             Information) Network Management Private Enterprise Code of
             the IANA-maintained ’Private Enterprise Numbers’
             registry.";
           reference
             "’PRIVATE ENTERPRISE NUMBERS’, SMI Network Management
               Private Enterprise Codes, April 2014,
                <http://www.iana.org/assignments/enterprise-numbers>";
         }
         leaf subtype {
           type uint8;
           mandatory true;
           description
             "An 8-bit field indicating the type of vendor-specific
              information carried in the option.  The namespace for this
              sub-type is managed by the vendor identified by the
              Vendor ID field.";
         }
         description
           "QoS Vendor-Specific Attribute.";
       }

       //Primary Structures (groupings)
       grouping qosattribute {
           leaf per-mn-agg-max-dl {
               type qos-pmip:Per-MN-Agg-Max-DL-Bit-Rate-Value;
               description "Per-MN-Agg-Max-DL-Bit-Rate Value";
           }
           leaf per-mn-agg-max-ul {
               type qos-pmip:Per-MN-Agg-Max-UL-Bit-Rate-Value;
               description "Per-MN-Agg-Max-UL-Bit-Rate Value";
           }
           container per-session-agg-max-dl {
               uses qos-pmip:Per-Session-Agg-Max-Bit-Rate-Value;
               description "Per-Session-Agg-Max-Bit-Rate Value";
           }
           container per-session-agg-max-ul {
               uses qos-pmip:Per-Session-Agg-Max-Bit-Rate-Value;
               description "Per-Session-Agg-Max-Bit-Rate Value";
           }
           uses qos-pmip:Allocation-Retention-Priority-Value;
           leaf agg-max-dl {
               type qos-pmip:Aggregate-Max-DL-Bit-Rate-Value;
               description "Aggregate-Max-DL-Bit-Rate Value";
           }
           leaf agg-max-ul {
               type qos-pmip:Aggregate-Max-UL-Bit-Rate-Value;
               description "Aggregate-Max-UL-Bit-Rate Value";
           }
           leaf gbr-dl {
               type qos-pmip:Guaranteed-DL-Bit-Rate-Value;
               description "Guaranteed-DL-Bit-Rate Value";
           }
           leaf gbr-ul {
               type qos-pmip:Guaranteed-UL-Bit-Rate-Value;
               description "Guaranteed-UL-Bit-Rate Value";
           }
           description "PMIP QoS Attributes. Note Vendor option
           is not a part of this grouping";
       }

       grouping qosoption {
           leaf srid {
               type sr-id;
               mandatory true;
               description "Service Request Identifier";
           }
           leaf trafficclass {
               type traffic-class;
               mandatory true;
               description "Traffic Class";
           }
           leaf operationcode {
               type operational-code;
               mandatory true;
               description "Operation Code";
           }
           uses qos-pmip:qosattribute;
           uses qos-pmip:QoS-Vendor-Specific-Attribute-Value-Base;
           container traffic-selector {
               uses traffic-selectors:traffic-selector;
               description "traffic selector";
           }
           description "PMIP QoS Option";
       }
   }
   <CODE ENDS>
```

A.4. Traffic Selectors YANG Model

This module defines traffic selector types commonly used in Proxy Mobile IP (PMIP).

This module references [RFC6991].

```
   <CODE BEGINS> file "ietf-trafficselector-types@2018-05-17.yang"
   module ietf-trafficselector-types {
    yang-version 1.1;

    namespace
    "urn:ietf:params:xml:ns:yang:ietf-trafficselector-types";

    prefix "traffic-selectors";

    import ietf-inet-types {
      prefix inet;
      revision-date 2013-07-15;
    }

    organization "IETF Distributed Mobility Management (DMM)
    Working Group";

    contact
    "WG Web: <http://tools.ietf.org/wg/netmod/>
    WG List: <mailto:netmod@ietf.org>

    WG Chair: Dapeng Liu
    <mailto:maxpassion@gmail.com>

    WG Chair: Sri Gundavelli
    <mailto:sgundave@cisco.com>

    Editor: Satoru Matsushima
    <mailto:satoru.matsushima@g.softbank.co.jp>

    Editor: Lyle Bertz
    <mailto:lylebe551144@gmail.com>";
```

```
description
"This module contains a collection of YANG definitions for
traffic selectors for flow bindings.

Copyright (c) 2016 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.";

 revision 2018-05-17 {
   description
    "Initial Revision.";
   reference
    "RFC 6088: Traffic Selectors for Flow Bindings";
 }

// Identities
  identity traffic-selector-format {
    description
    "The base type for Traffic-Selector Formats";
  }

  identity ipv4-binary-selector-format {
    base traffic-selector-format;
    description
      "IPv4 Binary Traffic Selector Format";
  }

  identity ipv6-binary-selector-format {
    base traffic-selector-format;
    description
      "IPv6 Binary Traffic Selector Format";
  }

  // Type definitions and groupings
  typedef ipsec-spi {
    type uint32;
    description
     "The first 32-bit IPsec Security Parameter Index (SPI)
     value on data. This field is defined in [RFC4303].";
```

```
          reference
          "RFC 4303: IP Encapsulating Security
          Payload (ESP)";
      }

      grouping traffic-selector-base {
        description "A grouping of the commen leaves between the
          v4 and v6 Traffic Selectors";
        container ipsec-spi-range {
          presence "Enables setting ipsec spi range";
          description
          "Inclusive range representing IPSec Security Parameter
          Indices to be used. When only start-spi is present, it
          represents a single spi.";
      leaf start-spi {
          type ipsec-spi;
          mandatory true;
          description
            "The first 32-bit IPsec SPI value on data.";
          }
      leaf end-spi {
            type ipsec-spi;
            must ". >= ../start-spi" {
              error-message
                "The end-spi must be greater than or equal
                 to start-spi";
          }
         description
           "If more than one contiguous SPI value needs to be matched,
           then this field indicates the end value of a range.";
          }
       }
       container source-port-range {
         presence "Enables setting source port range";
         description
          "Inclusive range representing source ports to be used.
           When only start-port is present, it represents a single
        port. These value(s) are from the range of port numbers
           defined by IANA (http://www.iana.org).";
         leaf start-port {
            type inet:port-number;
            mandatory true;
            description
            "The first 16-bit source port number to be matched";
         }
         leaf end-port {
            type inet:port-number;
            must ". >= ../start-port" {
```

```
            error-message
             "The end-port must be greater than or equal to start-port";
           }
           description
            "The last 16-bit source port number to be matched";
          }
       }
       container destination-port-range {
         presence "Enables setting destination port range";
         description
          "Inclusive range representing destination ports to be used.
          When only start-port is present, it represents a single
          port.";
           leaf start-port {
             type inet:port-number;
             mandatory true;
             description
             "The first 16-bit destination port number to be matched";
          }
          leaf end-port {
            type inet:port-number;
            must ". >= ../start-port" {
            error-message
              "The end-port must be greater than or equal to
             start-port";
           }
           description
        "The last 16-bit destination port number to be matched";
        }
      }
    }

    grouping ipv4-binary-traffic-selector {
      container source-address-range-v4 {
         presence "Enables setting source IPv4 address range";
         description
          "Inclusive range representing IPv4 addresses to be used. When
          only start-address is present, it represents a single
          address.";
         leaf start-address {
           type inet:ipv4-address;
           mandatory true;
          description
           "The first source address to be matched";
         }
         leaf end-address {
           type inet:ipv4-address;
           description
```

```
            "The last source address to be matched";
          }
      }
      container destination-address-range-v4 {
         presence "Enables setting destination IPv4 address range";
         description
           "Inclusive range representing IPv4 addresses to be used.
           When only start-address is present, it represents a
           single address.";
         leaf start-address {
           type inet:ipv4-address;
           mandatory true;
           description
            "The first destination address to be matched";
         }
         leaf end-address {
           type inet:ipv4-address;
           description
            "The last destination address to be matched";
         }
      }
      container ds-range {
         presence "Enables setting dscp range";
         description
          "Inclusive range representing DiffServ Codepoints to be used.
          When only start-ds is present, it represents a single
          Codepoint.";
         leaf start-ds {
           type inet:dscp;
           mandatory true;
           description
            "The first differential service value to be matched";
       }
       leaf end-ds {
         type inet:dscp;
         must ". >= ../start-ds" {
           error-message
             "The end-ds must be greater than or equal to start-ds";
         }
         description
           "The last differential service value to be matched";
      }
     }
     container protocol-range {
       presence "Enables setting protocol range";
       description
         "Inclusive range representing IP protocol(s) to be used. When
          only start-protocol is present, it represents a single
          protocol.";
       leaf start-protocol {
         type uint8;
         mandatory true;
         description
           "The first 8-bit protocol value to be matched.";
        }
        leaf end-protocol {
          type uint8;
          must ". >= ../start-protocol" {
            error-message
              "The end-protocol must be greater than or equal to
             start-protocol";
          }
        description
          "The last 8-bit protocol value to be matched.";
        }
      }
      description "ipv4 binary traffic selector";
    }
     grouping ipv6-binary-traffic-selector {
      container source-address-range-v6 {
        presence "Enables setting source IPv6 address range";
         description
          "Inclusive range representing IPv6 addresses to be used.
          When only start-address is present, it represents a
          single address.";
         leaf start-address {
           type inet:ipv6-address;
           mandatory true;
           description
           "The first source address, from the
           range of 128-bit IPv6 addresses to be matched";
         }
         leaf end-address {
           type inet:ipv6-address;
           description
               "The last source address, from the
               range of 128-bit IPv6 addresses to be matched";
         }
      }
      container destination-address-range-v6 {
        presence "Enables setting destination IPv6 address range";
        description
          "Inclusive range representing IPv6 addresses to be used.
           When only start-address is present, it represents a
           single address.";
        leaf start-address {
          type inet:ipv6-address;
          mandatory true;
          description
              "The first destination address, from the
              range of 128-bit IPv6 addresses to be matched";
        }
        leaf end-address {
          type inet:ipv6-address;
          description
              "The last destination address, from the
              range of 128-bit IPv6 addresses to be matched";
       }
     }
     container flow-label-range {
       presence "Enables setting Flow Label range";
       description
         "Inclusive range representing IPv4 addresses to be used. When
          only start-flow-label is present, it represents a single
          flow label.";
       leaf start-flow-label {
         type inet:ipv6-flow-label;
         description
           "The first flow label value to be matched";
       }
       leaf end-flow-label {
         type inet:ipv6-flow-label;
         must ". >= ../start-flow-label" {
           error-message
             "The end-flow-lable must be greater than or equal to
              start-flow-label";
         }
         description
            "The first flow label value to be matched";
       }
      }
     container traffic-class-range {
       presence "Enables setting the traffic class range";
       description
        "Inclusive range representing IPv4 addresses to be used. When
         only start-traffic-class is present, it represents a single
         traffic class.";
       leaf start-traffic-class {
         type inet:dscp;
         description
          "The first traffic class value to be matched";
         reference
          "RFC 3260: New Terminology and Clarifications for Diffserv
           RFC 3168: The Addition of Explicit Congestion Notification
```

```
            (ECN) to IP";
        }
        leaf end-traffic-class {
          type inet:dscp;
          must ". >= ../start-traffic-class" {
            error-message
              "The end-traffic-class must be greater than or equal to
               start-traffic-class";
          }
          description
            "The last traffic class value to be matched";
        }
      }
      container next-header-range {
        presence "Enables setting Next Header range";
        description
         "Inclusive range representing Next Headers to be used. When
          only start-next-header is present, it represents a
          single Next Header.";
        leaf start-next-header {
          type uint8;
          description
           "The first 8-bit next header value to be matched.";
        }
        leaf end-next-header {
          type uint8;
          must ". >= ../start-next-header" {
            error-message
              "The end-next-header must be greater than or equal to
              start-next-header";
          }
          description
            "The last 8-bit next header value to be matched.";
        }
      }
      description "ipv6 binary traffic selector";
    }

      grouping traffic-selector {
        leaf ts-format {
           type identityref {
             base traffic-selector-format;
           }
           description "Traffic Selector Format";
        }
        uses traffic-selectors:traffic-selector-base;
        uses traffic-selectors:ipv4-binary-traffic-selector;
        uses traffic-selectors:ipv6-binary-traffic-selector;
```

```
       description
        "The traffic selector includes the parameters used to match
          packets for a specific flow binding.";
       reference
        "RFC 6089: Flow Bindings in Mobile IPv6 and Network
          Mobility (NEMO) Basic Support";
     }
   }
   <CODE ENDS>
```

A.5. RFC 5777 Classifier YANG Model

This module defines the RFC 5777 Classifer.

This module references [RFC5777].

```
   <CODE BEGINS> file "ietf-diam-trafficclassifier@2018-05-17.yang"
   module ietf-diam-trafficclassifier {
    yang-version 1.1;

    namespace
    "urn:ietf:params:xml:ns:yang:ietf-diam-trafficclassifier";

    prefix "diamclassifier";

    import ietf-inet-types {
      prefix inet;
      revision-date 2013-07-15;
    }
    import ietf-yang-types { prefix yang-types; }

    organization "IETF Distributed Mobility Management (DMM)
    Working Group";

    contact
    "WG Web: <http://tools.ietf.org/wg/netmod/>
    WG List: <mailto:netmod@ietf.org>

    WG Chair: Dapeng Liu
    <mailto:maxpassion@gmail.com>

    WG Chair: Sri Gundavelli
    <mailto:sgundave@cisco.com>

    Editor: Satoru Matsushima
    <mailto:satoru.matsushima@g.softbank.co.jp>

    Editor: Lyle Bertz
    <mailto:lylebe551144@gmail.com>";

    description
    "This module contains a collection of YANG definitions for
    traffic classification and QoS Attributes for Diameter.

    Copyright (c) 2018 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.";

     revision 2018-05-17 {
       description
           "Initial";
       reference
       "RFC 5777: Traffic Classification and Quality of Service (QoS)
           Attributes for Diameter";
     }

       typedef eui64-address-type {
           type string {
               length "6";
           }
           description
               "specifies a single layer 2 address in EUI-64 format.
                The value is an 8-octet encoding of the address as
                it would appear in the frame header.";
       }
       typedef direction-type {
           type enumeration {
               enum IN {
                   value 0;
                   description
                       "Applies to flows from the managed terminal.";
               }
               enum OUT {
                   value 1;
                   description
                    "Applies to flows to the managed terminal.";
               }
```

```
                enum BOTH {
                    value 2;
                    description
                        "Applies to flows both to and from the managed
                            terminal.";
                }
            }
            description
                "Specifies in which direction to apply the classifier.";
        }
        typedef negated-flag-type {
            type enumeration {
                enum False { value 0;
                    description "false"; }
                enum True { value 1;
                    description "True"; }
            }
            description
                "When set to True, the meaning of the match is
                 inverted and the classifier will match addresses
                 other than those specified by the From-Spec or
                 To-Spec AVP.

                 Note that the negation does not impact the port
                    comparisons.";
        }
        grouping index {
            leaf index {
                type uint16;
                mandatory true;
                description "Identifier used for referencing";
            }
            description "Index Value";
        }
        grouping to-from-spec-value {
            leaf-list ip-address {
                type inet:ip-address;
                description "IP address";
            }
            list ip-address-range {
                key index;
                uses diamclassifier:index;
                leaf ip-address-start {
                    type inet:ip-address;
                    description "IP Address Start";
                }
                leaf ip-address-end {
                    type inet:ip-address;
```

```
                   description "IP Address End";
               }
               description "IP Address Range";
           }
           leaf-list ip-address-mask {
               type inet:ip-prefix;
               description "IP Address Mask";
           }
           leaf-list mac-address {
               type yang-types:mac-address;
               description "MAC address";
           }
           list mac-address-mask {
               key mac-address;
               leaf mac-address {
                   type yang-types:mac-address;
                   mandatory true;
                   description "MAC address";
               }
               leaf macaddress-mask-pattern {
                   type yang-types:mac-address;
                   mandatory true;
                   description
                    "The value specifies the bit positions of a
                     MAC address that are taken for matching.";
               }
               description "MAC Address Mask";
           }
           leaf-list eui64-address {
               type diamclassifier:eui64-address-type;
               description "EUI64 Address";
           }
           list eui64-address-mask {
               key eui64-address;
               leaf eui64-address {
                   type diamclassifier:eui64-address-type;
                   mandatory true;
                   description "eui64 address";
               }
               leaf eui64-address-mask-pattern {
                   type diamclassifier:eui64-address-type;
                   mandatory true;
                   description
                    "The value is 8 octets specifying the bit
                     positions of a EUI64 address that are taken
                    for matching.";
               }
               description "EUI64 Address Mask";
```

```
            }
            leaf-list port {
                type inet:port-number;
                description "Port Number";
            }
            list port-range {
                key index;
                uses diamclassifier:index;
                leaf ip-address-start {
                    type inet:port-number;
                    description "Port Start";
                }
                leaf ip-address-end {
                    type inet:port-number;
                    description "Port End";
                }
                description "Port Range";
            }
            leaf negated {
                type diamclassifier:negated-flag-type;
                description "Negated";
            }
            leaf use-assigned-address {
                type boolean;
                description "Use Assigned Address";
            }
            description
                "Basic traffic description value";
        }

        grouping option-type-group {
            leaf option-type {
                type uint8;
                mandatory true;
                description "Option Type";
            }
            leaf-list ip-option-value {
                type string;
                description "Option Value";
            }
            leaf negated {
                type diamclassifier:negated-flag-type;
                description "Negated";
            }
            description "Common X Option Pattern";
        }
        typedef vlan-id {
            type uint32 {
                range "0..4095";
            }
            description "VLAN ID";
        }

      grouping classifier {
        leaf protocol {
            type uint8;
            description "Protocol";
        }
        leaf direction {
            type diamclassifier:direction-type;
            description "Direction";
        }
        list from-spec {
            key index;
            uses diamclassifier:index;
            uses diamclassifier:to-from-spec-value;
            description "from specification";
        }
        list to-spec {
            key index;
            uses diamclassifier:index;
            uses diamclassifier:to-from-spec-value;
            description "to specification";
        }
        leaf-list disffserv-code-point {
            type inet:dscp;
            description "DSCP";
        }
        leaf fragmentation-flag {
        type enumeration {
                enum DF {
                    value 0;
                    description "Don’t Fragment";
                }
                enum MF {
                    value 1;
                    description "More Fragments";
                }
            }
            description "Fragmenttation Flag";
        }
        list ip-option {
            key option-type;
            uses diamclassifier:option-type-group;
            description "IP Option Value";
        }
        list tcp-option {
            key option-type;
            uses diamclassifier:option-type-group;
            description "TCP Option Value";
        }
        list tcp-flag {
            key tcp-flag-type;
            leaf tcp-flag-type {
                type uint32;
                mandatory true;
                description "TCP Flag Type";
            }
            leaf negated {
                type diamclassifier:negated-flag-type;
                description "Negated";
            }
            description "TCP Flags";
        }
        list icmp-option {
            key option-type;
            uses diamclassifier:option-type-group;
            description "ICMP Option Value";
        }
        list eth-option {
            key index;
            uses diamclassifier:index;
            container eth-proto-type {
                leaf-list eth-ether-type {
                    type string {
                        length "2";
                    }
                    description "value of ethertype field";
                }
                leaf-list eth-sap {
                    type string {
                        length "2";
                    }
                    description "802.2 SAP";
                }
                description "Ether Proto Type";
            }
            list vlan-id-range {
                key index;
                uses diamclassifier:index;
                leaf-list s-vlan-id-start {
                    type diamclassifier:vlan-id;
                    description "S-VID  VLAN ID Start";
                }
                leaf-list s-vlan-id-end {
                    type diamclassifier:vlan-id;
                    description "S-VID VLAN ID End";
                }
                leaf-list c-vlan-id-start {
                    type diamclassifier:vlan-id;
                    description "C-VID  VLAN ID Start";
                }
                leaf-list c-vlan-id-end {
                    type diamclassifier:vlan-id;
                    description "C-VID  VLAN ID End";
                }
                description "VLAN ID Range";
            }
            list user-priority-range {
                key index;
                uses diamclassifier:index;
                leaf-list low-user-priority {
                    type uint32 {
                        range "0..7";
                    }
                    description "Low User Priority";
                }
                leaf-list high-user-priority {
                    type uint32 {
                        range "0..7";
                    }
                    description "High User Priority";
                }
                description "User priority range";
            }
            description "Ether Option";
        }
        description "RFC 5777 Classifier";
      }
    }
    <CODE ENDS>
```

Appendix B. FPC YANG Tree Structure

This section only shows the structure for FPC YANG model. NOTE, it does NOT show the settings, Action values or Descriptor Value.

```
   descriptor_value:
   +--rw (descriptor-value)
     +--:(all-traffic)
     |  +--rw all-traffic?                    empty
     +--:(no-traffic)
     |  +--rw no-traffic?                     empty
     +--:(prefix-descriptor)
     |  +--rw destination-ip?                 inet:ip-prefix
     |  +--rw source-ip?                      inet:ip-prefix
     +--:(pmip-selector)
     |  +--rw ts-format?                      identityref
     |  +--rw ipsec-spi-range!
     |  |  +--rw start-spi    ipsec-spi
     |  |  +--rw end-spi?     ipsec-spi
     |  +--rw source-port-range!
     |  |  +--rw start-port    inet:port-number
     |  |  +--rw end-port?     inet:port-number
     |  +--rw destination-port-range!
     |  |  +--rw start-port    inet:port-number
     |  |  +--rw end-port?     inet:port-number
     |  +--rw source-address-range-v4!
     |  |  +--rw start-address    inet:ipv4-address
     |  |  +--rw end-address?     inet:ipv4-address
     |  +--rw destination-address-range-v4!
     |  |  +--rw start-address    inet:ipv4-address
     |  |  +--rw end-address?     inet:ipv4-address
     |  +--rw ds-range!
     |  |  +--rw start-ds    inet:dscp
     |  |  +--rw end-ds?     inet:dscp
     |  +--rw protocol-range!
     |  |  +--rw start-protocol    uint8
     |  |  +--rw end-protocol?     uint8
     |  +--rw source-address-range-v6!
     |  |  +--rw start-address    inet:ipv6-address
     |  |  +--rw end-address?     inet:ipv6-address
     |  +--rw destination-address-range-v6!
     |  |  +--rw start-address    inet:ipv6-address
     |  |  +--rw end-address?     inet:ipv6-address
     |  +--rw flow-label-range!
     |  |  +--rw start-flow-label?   inet:ipv6-flow-label
     |  |  +--rw end-flow-label?     inet:ipv6-flow-label
     |  +--rw traffic-class-range!
     |  |  +--rw start-traffic-class?   inet:dscp
     |  |  +--rw end-traffic-class?     inet:dscp
     |  +--rw next-header-range!
     |     +--rw start-next-header?   uint8
     |     +--rw end-next-header?     uint8
     +--:(rfc5777-classifier-template)
     |  +--rw rfc5777-classifier-template
     |     +--rw protocol?              uint8
     |     +--rw direction?             diamclassifier:direction-type
     |     +--rw from-spec* [index]
     |     |  +--rw index                  uint16
     |     |  +--rw ip-address*             inet:ip-address
     |     |  +--rw ip-address-range* [index]
     |     |  |  +--rw index               uint16
     |     |  |  +--rw ip-address-start?   inet:ip-address
     |     |  |  +--rw ip-address-end?     inet:ip-address
     |     |  +--rw ip-address-mask*        inet:ip-prefix
     |     |  +--rw mac-address*            yang-types:mac-address
     |     |  +--rw mac-address-mask* [mac-address]
     |     |  |  +--rw mac-address              yang-types:mac-address
     |     |  |  +--rw macaddress-mask-pattern  yang-types:mac-address
     |     |  +--rw eui64-address*
                         diamclassifier:eui64-address-type
     |     |  +--rw eui64-address-mask* [eui64-address]
     |     |  |  +--rw eui64-address
                         diamclassifier:eui64-address-type
     |     |  |  +--rw eui64-address-mask-pattern
                         diamclassifier:eui64-address-type
     |     |  +--rw port*                   inet:port-number
     |     |  +--rw port-range* [index]
     |     |  |  +--rw index               uint16
     |     |  |  +--rw ip-address-start?   inet:port-number
     |     |  |  +--rw ip-address-end?     inet:port-number
     |     |  +--rw negated?
                         diamclassifier:negated-flag-type
     |     |  +--rw use-assigned-address?   boolean
     |     +--rw to-spec* [index]
     |     |  +--rw index                   uint16
     |     |  +--rw ip-address*             inet:ip-address
     |     |  +--rw ip-address-range* [index]
     |     |  |  +--rw index               uint16
     |     |  |  +--rw ip-address-start?   inet:ip-address
     |     |  |  +--rw ip-address-end?     inet:ip-address
     |     |  +--rw ip-address-mask*        inet:ip-prefix
     |     |  +--rw mac-address*            yang-types:mac-address
     |     |  +--rw mac-address-mask* [mac-address]
     |     |  |  +--rw mac-address                yang-types:mac-address
     |     |  |  +--rw macaddress-mask-pattern    yang-types:mac-address
     |     |  +--rw eui64-address*
                         diamclassifier:eui64-address-type
     |     |  +--rw eui64-address-mask* [eui64-address]
     |     |  |  +--rw eui64-address
                         diamclassifier:eui64-address-type
     |     |  |  +--rw eui64-address-mask-pattern
                         diamclassifier:eui64-address-type
     |     |  +--rw port*                   inet:port-number
     |     |  +--rw port-range* [index]
     |     |  |  +--rw index               uint16
     |     |  |  +--rw ip-address-start?   inet:port-number
     |     |  |  +--rw ip-address-end?     inet:port-number
     |     |  +--rw negated?
                         diamclassifier:negated-flag-type
     |     |  +--rw use-assigned-address?   boolean
     |     +--rw disffserv-code-point*   inet:dscp
     |     +--rw fragmentation-flag?     enumeration
     |     +--rw ip-option* [option-type]
     |     |  +--rw option-type        uint8
     |     |  +--rw ip-option-value*   string
     |     |  +--rw negated?           diamclassifier:negated-flag-type
     |     +--rw tcp-option* [option-type]
     |     |  +--rw option-type        uint8
     |     |  +--rw ip-option-value*   string
     |     |  +--rw negated?           diamclassifier:negated-flag-type
     |     +--rw tcp-flag* [tcp-flag-type]
     |     |  +--rw tcp-flag-type    uint32
     |     |  +--rw negated?         diamclassifier:negated-flag-type
     |     +--rw icmp-option* [option-type]
     |     |  +--rw option-type        uint8
     |     |  +--rw ip-option-value*   string
     |     |  +--rw negated?           diamclassifier:negated-flag-type
     |     +--rw eth-option* [index]
     |        +--rw index                 uint16
     |        +--rw eth-proto-type
     |        |  +--rw eth-ether-type*   string
     |        |  +--rw eth-sap*          string
     |        +--rw vlan-id-range* [index]
     |        |  +--rw index              uint16
     |        |  +--rw s-vlan-id-start*   diamclassifier:vlan-id
     |        |  +--rw s-vlan-id-end*     diamclassifier:vlan-id
     |        |  +--rw c-vlan-id-start*   diamclassifier:vlan-id
     |        |  +--rw c-vlan-id-end*     diamclassifier:vlan-id
     |        +--rw user-priority-range* [index]
     |           +--rw index                 uint16
     |           +--rw low-user-priority*    uint32
     |           +--rw high-user-priority*   uint32
     +--:(packet-filter)
     |  +--rw packet-filter
     |     +--rw direction?             fpcbase:packet-filter-direction
     |     +--rw identifier?              uint8
     |     +--rw evaluation-precedence?   uint8
     |     +--rw contents* [component-type-identifier]
     |        +--rw component-type-identifier fpcbase:component-type-id
     |        +--rw (value)?
     |           +--:(ipv4-local)
     |           |  +--rw ipv4-local?                  inet:ipv4-address
     |           +--:(ipv6-prefix-local)
     |           |  +--rw ipv6-prefix-local?           inet:ipv6-prefix
     |           +--:(ipv4-ipv6-remote)
     |           |  +--rw ipv4-ipv6-remote?            inet:ip-address
     |           +--:(ipv6-prefix-remote)
     |           |  +--rw ipv6-prefix-remote?          inet:ipv6-prefix
     |           +--:(next-header)
     |           |  +--rw next-header?                 uint8
     |           +--:(local-port)
     |           |  +--rw local-port?                  inet:port-number
     |           +--:(local-port-range)
     |           |  +--rw local-port-lo?               inet:port-number
     |           |  +--rw local-port-hi?               inet:port-number
     |           +--:(remote-port)
     |           |  +--rw remote-port?                 inet:port-number
     |           +--:(remote-port-range)
     |           |  +--rw remote-port-lo?              inet:port-number
     |           |  +--rw remote-port-hi?              inet:port-number
     |           +--:(ipsec-index)
     |           |  +--rw ipsec-index?     traffic-selectors:ipsec-spi
     |           +--:(traffic-class)
     |           |  +--rw traffic-class?               inet:dscp
     |           +--:(traffic-class-range)
     |           |  +--rw traffic-class-lo?            inet:dscp
     |           |  +--rw traffic-class-hi?            inet:dscp
     |           +--:(flow-label)
     |              +--rw flow-label*    inet:ipv6-flow-label
     +--:(tunnel-info)
        +--rw tunnel-info
           +--rw tunnel-local-address?    inet:ip-address
           +--rw tunnel-remote-address?   inet:ip-address
           +--rw mtu-size?                uint32
           +--rw tunnel?                  identityref
           +--rw payload-type?            enumeration
           +--rw gre-key?                 uint32
           +--rw gtp-tunnel-info
           |  +--rw local-tunnel-identifier?    uint32
           |  +--rw remote-tunnel-identifier?   uint32
           |  +--rw sequence-numbers-enabled?   boolean
           +--rw ebi?                     fpcbase:ebi-type
           +--rw lbi?                     fpcbase:ebi-type

   action_value:
   +--:(action-value)
   |  +--rw (action-value)
   |     +--:(drop)
   |     |  +--rw drop?                           empty
   |     +--:(rewrite)
   |     |  +--rw rewrite
   |     |     +--rw (rewrite-value)?
   |     |        +--:(prefix-descriptor)
   |     |        |  +--rw destination-ip?               inet:ip-prefix
   |     |        |  +--rw source-ip?                    inet:ip-prefix
   |     |        +--:(pmip-selector)
   |     |        |  +--rw ts-format?                      identityref
   |     |        |  +--rw ipsec-spi-range!
   |     |        |  |  +--rw start-spi    ipsec-spi
   |     |        |  |  +--rw end-spi?     ipsec-spi
   |     |        |  +--rw source-port-range!
   |     |        |  |  +--rw start-port    inet:port-number
   |     |        |  |  +--rw end-port?     inet:port-number
   |     |        |  +--rw destination-port-range!
   |     |        |  |  +--rw start-port    inet:port-number
   |     |        |  |  +--rw end-port?     inet:port-number
   |     |        |  +--rw source-address-range-v4!
   |     |        |  |  +--rw start-address    inet:ipv4-address
   |     |        |  |  +--rw end-address?     inet:ipv4-address
   |     |        |  +--rw destination-address-range-v4!
   |     |        |  |  +--rw start-address    inet:ipv4-address
   |     |        |  |  +--rw end-address?     inet:ipv4-address
   |     |        |  +--rw ds-range!
   |     |        |  |  +--rw start-ds    inet:dscp
   |     |        |  |  +--rw end-ds?     inet:dscp
   |     |        |  +--rw protocol-range!
   |     |        |  |  +--rw start-protocol    uint8
   |     |        |  |  +--rw end-protocol?     uint8
   |     |        |  +--rw source-address-range-v6!
   |     |        |  |  +--rw start-address    inet:ipv6-address
   |     |        |  |  +--rw end-address?     inet:ipv6-address
   |     |        |  +--rw destination-address-range-v6!
   |     |        |  |  +--rw start-address    inet:ipv6-address
   |     |        |  |  +--rw end-address?     inet:ipv6-address
   |     |        |  +--rw flow-label-range!
   |     |        |  |  +--rw start-flow-label?   inet:ipv6-flow-label
   |     |        |  |  +--rw end-flow-label?     inet:ipv6-flow-label
   |     |        |  +--rw traffic-class-range!
   |     |        |  |  +--rw start-traffic-class?   inet:dscp
   |     |        |  |  +--rw end-traffic-class?     inet:dscp
   |     |        |  +--rw next-header-range!
   |     |        |     +--rw start-next-header?   uint8
   |     |        |     +--rw end-next-header?     uint8
   |     |        +--:(rfc5777-classifier-template)
   |     |           +--rw rfc5777-classifier-template
   |     |              +--rw protocol?               uint8
   |     |              +--rw direction?
                         diamclassifier:direction-type
   |     |              +--rw from-spec* [index]
   |     |              |  +--rw index                   uint16
   |     |              |  +--rw ip-address*             inet:ip-address
   |     |              |  +--rw ip-address-range* [index]
   |     |              |  |  +--rw index              uint16
   |     |              |  |  +--rw ip-address-start?   inet:ip-address
   |     |              |  |  +--rw ip-address-end?     inet:ip-address
   |     |              |  +--rw ip-address-mask*        inet:ip-prefix
   |     |              |  +--rw mac-address*    yang-types:mac-address
   |     |              |  +--rw mac-address-mask* [mac-address]
   |     |              |  |  +--rw mac-address
   |     |                           yang-types:mac-address
   |     |              |  |  +--rw macaddress-mask-pattern
   |     |                           yang-types:mac-address
   |     |              |  +--rw eui64-address*
   |     |                           diamclassifier:eui64-address-type
   |     |              |  +--rw eui64-address-mask* [eui64-address]
   |     |              |  |  +--rw eui64-address
   |     |                           diamclassifier:eui64-address-type
   |     |              |  |  +--rw eui64-address-mask-pattern
   |     |                           diamclassifier:eui64-address-type
   |     |              |  +--rw port*                 inet:port-number
   |     |              |  +--rw port-range* [index]
   |     |              |  |  +--rw index              uint16
   |     |              |  |  +--rw ip-address-start?   inet:port-number
   |     |              |  |  +--rw ip-address-end?     inet:port-number
   |     |              |  +--rw negated?
   |     |                           diamclassifier:negated-flag-type
   |     |              |  +--rw use-assigned-address?   boolean
   |     |              +--rw to-spec* [index]
   |     |              |  +--rw index                   uint16
   |     |              |  +--rw ip-address*             inet:ip-address
   |     |              |  +--rw ip-address-range* [index]
   |     |              |  |  +--rw index              uint16
   |     |              |  |  +--rw ip-address-start?   inet:ip-address
   |     |              |  |  +--rw ip-address-end?     inet:ip-address
   |     |              |  +--rw ip-address-mask*        inet:ip-prefix
   |     |              |  +--rw mac-address*
   |     |                           yang-types:mac-address
   |     |              |  +--rw mac-address-mask* [mac-address]
   |     |              |  |  +--rw mac-address
   |     |                           yang-types:mac-address
   |     |              |  |  +--rw macaddress-mask-pattern
   |     |                           yang-types:mac-address
   |     |              |  +--rw eui64-address*
   |     |                           diamclassifier:eui64-address-type
   |     |              |  +--rw eui64-address-mask* [eui64-address]
   |     |              |  |  +--rw eui64-address
   |     |                           diamclassifier:eui64-address-type
   |     |              |  |  +--rw eui64-address-mask-pattern
```

```
                                    diamclassifier:eui64-address-type
   |     |              |  +--rw port*                 inet:port-number
   |     |              |  +--rw port-range* [index]
   |     |              |  |  +--rw index               uint16
   |     |              |  |  +--rw ip-address-start?   inet:port-number
   |     |              |  |  +--rw ip-address-end?     inet:port-number
   |     |              |  +--rw negated?
                                    diamclassifier:negated-flag-type
   |     |              |  +--rw use-assigned-address?   boolean
   |     |              +--rw disffserv-code-point*   inet:dscp
   |     |              +--rw fragmentation-flag?     enumeration
   |     |              +--rw ip-option* [option-type]
   |     |              |  +--rw option-type        uint8
   |     |              |  +--rw ip-option-value*   string
   |     |              |  +--rw negated?
                                    diamclassifier:negated-flag-type
   |     |              +--rw tcp-option* [option-type]
   |     |              |  +--rw option-type        uint8
   |     |              |  +--rw ip-option-value*   string
   |     |              |  +--rw negated?
                                    diamclassifier:negated-flag-type
   |     |              +--rw tcp-flag* [tcp-flag-type]
   |     |              |  +--rw tcp-flag-type    uint32
   |     |              |  +--rw negated?
                                    diamclassifier:negated-flag-type
   |     |              +--rw icmp-option* [option-type]
   |     |              |  +--rw option-type        uint8
   |     |              |  +--rw ip-option-value*   string
   |     |              |  +--rw negated?
                                    diamclassifier:negated-flag-type
   |     |              +--rw eth-option* [index]
   |     |                 +--rw index                  uint16
   |     |                 +--rw eth-proto-type
   |     |                 |  +--rw eth-ether-type*   string
   |     |                 |  +--rw eth-sap*          string
   |     |                 +--rw vlan-id-range* [index]
   |     |                 |  +--rw index              uint16
   |     |                 |  +--rw s-vlan-id-start*
                                    diamclassifier:vlan-id
   |     |                 |  +--rw s-vlan-id-end*
                                    diamclassifier:vlan-id
   |     |                 |  +--rw c-vlan-id-start*
                                    diamclassifier:vlan-id
   |     |                 |  +--rw c-vlan-id-end*
                                    diamclassifier:vlan-id
   |     |                 +--rw user-priority-range* [index]
   |     |                    +--rw index                 uint16
   |     |                    +--rw low-user-priority*    uint32
   |     |                    +--rw high-user-priority*   uint32
   |     +--:(copy-forward-nexthop)
   |     |  +--rw copy-forward-nexthop
   |     |     +--rw (next-hop-value)?
   |     |        +--:(ip-address)
   |     |        |  +--rw ip-address?           inet:ip-address
   |     |        +--:(mac-address)
   |     |        |  +--rw mac-address?          ytypes:mac-address
   |     |        +--:(service-path)
   |     |        |  +--rw service-path?   fpcbase:fpc-service-path-id
   |     |        +--:(mpls-path)
   |     |        |  +--rw mpls-path?            fpcbase:fpc-mpls-label
   |     |        +--:(nsh)
   |     |        |  +--rw nsh?                  string
   |     |        +--:(interface)
   |     |        |  +--rw interface?            uint16
   |     |        +--:(segment-identifier)
   |     |        |  +--rw segment-identifier?   fpcbase:segment-id
   |     |        +--:(mpls-label-stack)
   |     |        |  +--rw mpls-label-stack*     fpcbase:fpc-mpls-label
   |     |        +--:(mpls-sr-stack)
   |     |        |  +--rw mpls-sr-stack*        fpcbase:fpc-mpls-label
   |     |        +--:(srv6-stack)
   |     |        |  +--rw srv6-stack*           fpcbase:segment-id
   |     |        +--:(tunnel-info)
   |     |           +--rw tunnel-info
   |     |              +--rw tunnel-local-address?    inet:ip-address
   |     |              +--rw tunnel-remote-address?   inet:ip-address
   |     |              +--rw mtu-size?                uint32
   |     |              +--rw tunnel?                  identityref
   |     |              +--rw payload-type?            enumeration
   |     |              +--rw gre-key?                 uint32
   |     |              +--rw gtp-tunnel-info
   |     |              |  +--rw local-tunnel-identifier?    uint32
   |     |              |  +--rw remote-tunnel-identifier?   uint32
   |     |              |  +--rw sequence-numbers-enabled?   boolean
   |     |              +--rw ebi?                     fpcbase:ebi-type
   |     |              +--rw lbi?                     fpcbase:ebi-type
   |     +--:(nexthop)
   |     |  +--rw nexthop
   |     |     +--rw (next-hop-value)?
   |     |        +--:(ip-address)
   |     |        |  +--rw ip-address?           inet:ip-address
   |     |        +--:(mac-address)
   |     |        |  +--rw mac-address?          ytypes:mac-address
   |     |        +--:(service-path)
   |     |        |  +--rw service-path?   fpcbase:fpc-service-path-id
   |     |        +--:(mpls-path)
   |     |        |  +--rw mpls-path?            fpcbase:fpc-mpls-label
   |     |        +--:(nsh)
   |     |        |  +--rw nsh?                  string
   |     |        +--:(interface)
   |     |        |  +--rw interface?            uint16
   |     |        +--:(segment-identifier)
   |     |        |  +--rw segment-identifier?   fpcbase:segment-id
   |     |        +--:(mpls-label-stack)
   |     |        |  +--rw mpls-label-stack*     fpcbase:fpc-mpls-label
   |     |        +--:(mpls-sr-stack)
   |     |        |  +--rw mpls-sr-stack*        fpcbase:fpc-mpls-label
   |     |        +--:(srv6-stack)
   |     |        |  +--rw srv6-stack*           fpcbase:segment-id
   |     |        +--:(tunnel-info)
   |     |           +--rw tunnel-info
   |     |              +--rw tunnel-local-address?    inet:ip-address
   |     |              +--rw tunnel-remote-address?   inet:ip-address
   |     |              +--rw mtu-size?                uint32
   |     |              +--rw tunnel?                  identityref
   |     |              +--rw payload-type?            enumeration
   |     |              +--rw gre-key?                 uint32
   |     |              +--rw gtp-tunnel-info
   |     |              |  +--rw local-tunnel-identifier?    uint32
   |     |              |  +--rw remote-tunnel-identifier?   uint32
   |     |              |  +--rw sequence-numbers-enabled?   boolean
   |     |              +--rw ebi?                     fpcbase:ebi-type
   |     |              +--rw lbi?                     fpcbase:ebi-type
   |     +--:(qos)
   |        +--rw trafficclass?                   inet:dscp
   |        +--rw per-mn-agg-max-dl?
                    qos-pmip:Per-MN-Agg-Max-DL-Bit-Rate-Value
   |        +--rw per-mn-agg-max-ul?
                    qos-pmip:Per-MN-Agg-Max-UL-Bit-Rate-Value
   |        +--rw per-session-agg-max-dl
   |        |  +--rw max-rate        uint32
   |        |  +--rw service-flag    boolean
   |        |  +--rw exclude-flag    boolean
   |        +--rw per-session-agg-max-ul
   |        |  +--rw max-rate        uint32
   |        |  +--rw service-flag    boolean
   |        |  +--rw exclude-flag    boolean
   |        +--rw priority-level                  uint8
   |        +--rw preemption-capability            enumeration
   |        +--rw preemption-vulnerability         enumeration
   |        +--rw agg-max-dl?
                    qos-pmip:Aggregate-Max-DL-Bit-Rate-Value
   |        +--rw agg-max-ul?
                    qos-pmip:Aggregate-Max-UL-Bit-Rate-Value
   |        +--rw gbr-dl?
                    qos-pmip:Guaranteed-DL-Bit-Rate-Value
   |        +--rw gbr-ul?
                    qos-pmip:Guaranteed-UL-Bit-Rate-Value
   |        +--rw qci?
                    fpcbase:fpc-qos-class-identifier
   |        +--rw ue-agg-max-bitrate?             uint32
   |        +--rw apn-ambr?                       uint32

   policy-configuration-value:
   |  |  |     +--rw (policy-configuration-value)?
   |  |  |        +--:(descriptor-value)
   |  |  |        |  ...
   |  |  |        +--:(action-value)
   |  |  |        |  ...
   |  |  |        +--:(setting-value)
   |  |  |           +--rw setting?                        <anydata>

   policy-configuration:
   |  |  |  +--rw policy-configuration* [index]
   |  |  |     +--rw index                           uint16
   |  |  |     +--rw extensible?             boolean
   |  |  |     +--rw static-attributes*      string
   |  |  |     +--rw mandatory-attributes*   string
   |  |  |     +--rw entity-state?           enumeration
   |  |  |     +--rw version?                uint32
   |  |  |     +--rw (policy-configuration-value)?
   |  |  |        ...

   module: ietf-dmm-fpc
   +--rw tenant* [tenant-key]
     +--rw tenant-key                    fpc:fpc-identity
     +--rw topology-information-model
     |  +--rw service-group* [service-group-key role-key]
     |  |  +--rw service-group-key     fpc:fpc-identity
     |  |  +--rw service-group-name?   string
     |  |  +--rw role-key              identityref
     |  |  +--rw role-name?            string
     |  |  +--rw protocol*             identityref
     |  |  +--rw feature*              identityref
     |  |  +--rw service-group-configuration* [index]
     |  |  |  +--rw index                           uint16
     |  |  |  +--rw (policy-configuration-value)?
     |  |  |        |  ...
     |  |  +--rw dpn* [dpn-key]
     |  |     +--rw dpn-key                 fpc:fpc-identity
     |  |     +--rw referenced-interface* [interface-key]
     |  |        +--rw interface-key             fpc:fpc-identity
     |  |        +--rw peer-service-group-key*   fpc:fpc-identity
     |  +--rw dpn* [dpn-key]
     |  |  +--rw dpn-key                           fpc:fpc-identity
     |  |  +--rw dpn-name?                         string
     |  |  +--rw dpn-resource-mapping-reference?   string
     |  |  +--rw domain-key                        fpc:fpc-identity
     |  |  +--rw service-group-key*                fpc:fpc-identity
     |  |  +--rw interface* [interface-key]
     |  |  |  +--rw interface-key        fpc:fpc-identity
     |  |  |  +--rw interface-name?      string
     |  |  |  +--rw role?                identityref
     |  |  |  +--rw protocol*            identityref
     |  |  |  +--rw interface-configuration* [index]
     |  |  |     +--rw (policy-configuration-value)?
     |  |  |           |  ...
     |  |  +--rw dpn-policy-configuration* [policy-template-key]
     |  |     +--rw policy-template-key     fpc:fpc-identity
     |  |     +--rw policy-configuration* [index]
     |  |        +--rw index    uint16
     |  |          +--rw (policy-configuration-value)?
     |  |               |  ...
     |  +--rw domain* [domain-key]
     |  |  +--rw domain-key        fpc:fpc-identity
     |  |  +--rw domain-name?      string
     |  |  +--rw domain-policy-configuration* [policy-template-key]
     |  |     +--rw policy-template-key     fpc:fpc-identity
     |  |     +--rw policy-configuration* [index]
     |  |              |         ...
     |  +--rw dpn-checkpoint
     |  |  +--rw basename?          fpc:fpc-identity
     |  |  +--rw base-checkpoint?   string
     |  +--rw service-group-checkpoint
     |  |  +--rw basename?          fpc:fpc-identity
     |  |  +--rw base-checkpoint?   string
     |  +--rw dpn-checkpoint
     |  |  +--rw basename?                   fpc:fpc-identity
     |  |  +--rw base-checkpoint?            string
     +--rw policy-information-model
     |  +--rw action-template* [action-template-key]
     |  |  +--rw action-template-key        fpc:fpc-identity
     |  |  +--rw (action-value)
     |  |  |                   ...
     |  |  +--rw extensible?                boolean
     |  |  +--rw static-attributes*         string
     |  |  +--rw mandatory-attributes*      string
     |  |  +--rw entity-state?              enumeration
     |  |  +--rw version?                   uint32
     |  +--rw descriptor-template* [descriptor-template-key]
     |  |  +--rw descriptor-template-key         fpc:fpc-identity
     |  |  +--rw (descriptor-value)
     |  |  |                   ...
     |  |  +--rw extensible?                     boolean
     |  |  +--rw static-attributes*              string
     |  |  +--rw mandatory-attributes*           string
     |  |  +--rw entity-state?                   enumeration
     |  |  +--rw version?                        uint32
     |  +--rw rule-template* [rule-template-key]
     |  |  +--rw rule-template-key           fpc:fpc-identity
     |  |  +--rw descriptor-match-type       enumeration
     |  |  +--rw descriptor-configuration* [descriptor-template-key]
     |  |  |  +--rw descriptor-template-key    fpc:fpc-identity
     |  |  |  +--rw direction?                 rfc5777:direction-type
     |  |  |  +--rw setting?                        <anydata>
     |  |  |  +--rw attribute-expression* [index]
     |  |  |     +--rw index                           uint16
     |  |  |     +--rw (descriptor-value)
     |  |  |     |  ...
     |  |  +--rw action-configuration* [action-order]
     |  |  |  +--rw action-order            uint32
     |  |  |  +--rw action-template-key     fpc:fpc-identity
     |  |  |  +--rw setting?                        <anydata>
     |  |  |  +--rw attribute-expression* [index]
     |  |  |     +--rw index                      uint16
     |  |  |     +--rw (action-value)
     |  |  |     |  ...
     |  |  +--rw extensible?                 boolean
     |  |  +--rw static-attributes*          string
     |  |  +--rw mandatory-attributes*       string
     |  |  +--rw entity-state?               enumeration
     |  |  +--rw version?                    uint32
     |  |  +--rw rule-configuration* [index]
     |  |     +--rw index    uint16
     |  |         +--rw (policy-configuration-value)?
     |  |           |  ...
     |  +--rw policy-template* [policy-template-key]
     |  |  +--rw policy-template-key     fpc:fpc-identity
     |  |  +--rw rule-template* [precedence]
     |  |  |  +--rw precedence           uint32
     |  |  |  +--rw rule-template-key    fpc:fpc-identity
     |  |  +--rw extensible?             boolean
     |  |  +--rw static-attributes*      string
     |  |  +--rw mandatory-attributes*   string
     |  |  +--rw entity-state?           enumeration
     |  |  +--rw version?                uint32
     |  |  +--rw policy-configuration* [index]
     |  |      ...
     |  +--rw basename?              fpc:fpc-identity
     |  +--rw base-checkpoint?       string
     +--rw mobility-context* [mobility-context-key]
     |  +--rw mobility-context-key    fpc:fpc-identity
     |  +--rw delegating-ip-prefix*   inet:ip-prefix
     |  +--rw parent-context?         fpc:fpc-identity
     |  +--rw child-context*          fpc:fpc-identity
     |  +--rw mobile-node
     |  |  +--rw ip-address*    inet:ip-address
     |  |  +--rw imsi?          fpcbase:imsi-type
     |  |  +--rw mn-policy-configuration* [policy-template-key]
     |  |     +--rw policy-template-key     fpc:fpc-identity
     |  |     +--rw policy-configuration* [index]
     |  |        ...
     |  +--rw domain
     |  |  +--rw domain-key?        fpc:fpc-identity
     |  |  +--rw domain-policy-configuration* [policy-template-key]
     |  |     +--rw policy-template-key     fpc:fpc-identity
     |  |     +--rw policy-configuration* [index]
     |  |        ...
     |   +--rw dpn* [dpn-key]
     |      +--rw dpn-key              fpc:fpc-identity
     |      +--rw dpn-policy-configuration* [policy-template-key]
     |      |  +--rw policy-template-key     fpc:fpc-identity
     |      |  +--rw policy-configuration* [index]
     |      |     ...
     |      +--rw role?                identityref
     |      +--rw service-data-flow* [identifier]
     |         +--rw identifier           uint32
     |         +--rw service-group-key?   fpc:fpc-identity
     |         +--rw interface* [interface-key]
     |         |  +--rw interface-key    fpc:fpc-identity
     |         +--rw service-data-flow-policy-
                       configuration* [policy-template-key]
     |            +--rw policy-template-key     fpc:fpc-identity
     |            +--rw policy-configuration* [index]
     |               ...
     +--rw monitor* [monitor-key]
       +--rw extensible?             boolean
       +--rw static-attributes*      string
       +--rw mandatory-attributes*   string
       +--rw entity-state?           enumeration
       +--rw version?                uint32
       +--rw monitor-key             fpc:fpc-identity
       +--rw target?                 string
       +--rw deferrable?             boolean
       +--rw (configuration)
          +--:(period)
          |  +--rw period?           uint32
          +--:(threshold-config)
          |  +--rw low?              uint32
          |  +--rw hi?               uint32
          +--:(schedule)
          |  +--rw schedule?         uint32
          +--:(event-identities)
          |  +--rw event-identities*       identityref
          +--:(event-ids)
             +--rw event-ids*        uint32

     rpcs:
       +---x configure
       |  +---w input
       |  |  +---w client-id          fpc:client-identifier
       |  |  +---w execution-delay?   uint32
       |  |  +---w yang-patch
       |  |     +---w patch-id    string
       |  |     +---w comment?    string
       |  |     +---w edit* [edit-id]
       |  |        +---w edit-id         string
       |  |        +---w operation       enumeration
       |  |        +---w target          target-resource-offset
       |  |        +---w point?          target-resource-offset
       |  |        +---w where?          enumeration
       |  |        +---w value?          <anydata>
       |  |        +---w reference-scope?   fpc:ref-scope
       |  |        +---w command-set
       |  |           +---w (instr-type)?
       |  |              +--:(instr-3gpp-mob)
       |  |              |  +---w instr-3gpp-mob? fpcbase:threegpp-instr
       |  |              +--:(instr-pmip)
       |  |                 +---w instr-pmip?       pmip-commandset
       |  +--ro output
       |     +--ro yang-patch-status
       |        +--ro patch-id       string
       |        +--ro (global-status)?
       |        |  +--:(global-errors)
       |        |  |  +--ro errors
       |        |  |     +--ro error*
       |        |  |        +--ro error-type       enumeration
       |        |  |        +--ro error-tag        string
       |        |  |        +--ro error-app-tag?   string
       |        |  |        +--ro error-path?      instance-identifier
       |        |  |        +--ro error-message?   string
       |        |  |        +--ro error-info?      <anydata>
       |        |  +--:(ok)
       |        |     +--ro ok?            empty
       |        +--ro edit-status
       |           +--ro edit* [edit-id]
       |              +--ro edit-id            string
       |              +--ro (edit-status-choice)?
       |                 +--:(ok)
       |                 |  +--ro ok?                empty
       |                 |  +--ro notify-follows?    boolean
       |                 |  +--ro subsequent-edit* [edit-id]
       |                 |     +--ro edit-id      string
       |                 |     +--ro operation    enumeration
       |                 |     +--ro target
                                   ypatch:target-resource-offset
       |                 |     +--ro point?
                                       ypatch:target-resource-offset
       |                 |     +--ro where?       enumeration
       |                 |     +--ro value?       <anydata>
       |                 +--:(errors)
       |                    +--ro errors
       |                       +--ro error*
       |                          +--ro error-type       enumeration
       |                          +--ro error-tag        string
       |                          +--ro error-app-tag?   string
       |                          +--ro error-path?
                                           instance-identifier
       |                          +--ro error-message?   string
       |                          +--ro error-info?      <anydata>
       +---x register_monitor
       |  +---w input
       |  |  +---w client-id          fpc:client-identifier
       |  |  +---w execution-delay?   uint32
       |  |  +---w operation-id              uint64
       |  |  +---w monitor* [monitor-key]
       |  |     +---w extensible?             boolean
       |  |     +---w static-attributes*      string
       |  |     +---w mandatory-attributes*   string
       |  |     +---w entity-state?           enumeration
       |  |     +---w version?                uint32
       |  |     +---w monitor-key             fpc:fpc-identity
       |  |     +---w target?                 string
       |  |     +---w deferrable?             boolean
       |  |     +---w (configuration)
       |  |        +--:(period)
       |  |        |  +---w period?                 uint32
       |  |        +--:(threshold-config)
       |  |        |  +---w low?                    uint32
       |  |        |  +---w hi?                     uint32
       |  |        +--:(schedule)
       |  |        |  +---w schedule?               uint32
       |  |        +--:(event-identities)
       |  |        |  +---w event-identities*       identityref
       |  |        +--:(event-ids)
       |  |           +---w event-ids*              uint32
       |  +--ro output
       |     +--ro operation-id     uint64
       |     +--ro (edit-status-choice)?
       |        +--:(ok)
       |        |  +--ro ok?       empty
       |        +--:(errors)
       |           +--ro errors
       |              +--ro error*
       |                 +--ro error-type       enumeration
       |                 +--ro error-tag        string
       |                 +--ro error-app-tag?   string
       |                 +--ro error-path?      instance-identifier
       |                 +--ro error-message?   string
       |                 +--ro error-info?      <anydata>
       +---x deregister_monitor
       |  +---w input
       |  |  +---w client-id          fpc:client-identifier
       |  |  +---w execution-delay?   uint32
       |  |  +---w operation-id              uint64
       |  |  +---w monitor* [monitor-key]
       |  |     +---w monitor-key    fpc:fpc-identity
       |  |     +---w send_data?     boolean
       |  +--ro output
       |     +--ro operation-id     uint64
       |     +--ro (edit-status-choice)?
       |        +--:(ok)
       |        |  +--ro ok?       empty
       |        +--:(errors)
       |           +--ro errors
       |              +--ro error*
       |                 +--ro error-type       enumeration
       |                 +--ro error-tag        string
       |                 +--ro error-app-tag?   string
       |                 +--ro error-path?      instance-identifier
       |                 +--ro error-message?   string
       |                 +--ro error-info?      <anydata>
       +---x probe
          +---w input
          |  +---w client-id          fpc:client-identifier
          |  +---w execution-delay?   uint32
          |  +---w operation-id              uint64
          |  +---w monitor* [monitor-key]
          |     +---w monitor-key    fpc:fpc-identity
          +--ro output
```

```
             +--ro operation-id     uint64
             +--ro (edit-status-choice)?
                +--:(ok)
                |  +--ro ok?       empty
                +--:(errors)
                   +--ro errors
                      +--ro error*
                         +--ro error-type       enumeration
                         +--ro error-tag        string
                         +--ro error-app-tag?   string
                         +--ro error-path?      instance-identifier
                         +--ro error-message?   string
                         +--ro error-info?      <anydata>

     notifications:
       +---n config-result-notification
       |  +--ro yang-patch-status
       |  |  +--ro patch-id       string
       |  |  +--ro (global-status)?
       |  |  |  +--:(global-errors)
       |  |  |  |  +--ro errors
       |  |  |  |     +--ro error*
       |  |  |  |        +--ro error-type       enumeration
       |  |  |  |        +--ro error-tag        string
       |  |  |  |        +--ro error-app-tag?   string
       |  |  |  |        +--ro error-path?      instance-identifier
       |  |  |  |        +--ro error-message?   string
       |  |  |  |        +--ro error-info?      <anydata>
       |  |  |  +--:(ok)
       |  |  |     +--ro ok?            empty
       |  |  +--ro edit-status
       |  |     +--ro edit* [edit-id]
       |  |        +--ro edit-id    string
       |  |        +--ro (edit-status-choice)?
       |  |           +--:(ok)
       |  |           |  +--ro ok?        empty
       |  |           +--:(errors)
       |  |              +--ro errors
       |  |                 +--ro error*
       |  |                    +--ro error-type       enumeration
       |  |                    +--ro error-tag        string
       |  |                    +--ro error-app-tag?   string
       |  |                    +--ro error-path?
       |  |                            instance-identifier
       |  |                    +--ro error-message?   string
       |  |                    +--ro error-info?      <anydata>
       |  +--ro subsequent-edit* [edit-id]
       |     +--ro edit-id      string
       |     +--ro operation    enumeration
       |     +--ro target       ypatch:target-resource-offset
       |     +--ro point?       ypatch:target-resource-offset
       |     +--ro where?       enumeration
       |     +--ro value?       <anydata>
       +---n notify
          +--ro notification-id?   uint32
          +--ro timestamp?         uint32
          +--ro report* [monitor-key]
             +--ro monitor-key               fpc:fpc-identity
             +--ro trigger?                  identityref
             +--ro (value)?
                +--:(dpn-candidate-available)
                |  +--ro node-id?                    inet:uri
                |  +--ro supported-interface-list* [role-key]
                |     +--ro role-key    identityref
                +--:(dpn-unavailable)
                |  +--ro dpn-id?                     fpc:fpc-identity
                +--:(report-value)
                   +--ro report-value?               <anydata>
```

Figure 38: YANG FPC Agent Tree

Appendix C. Change Log

C.1. Changes since Version 09

The following changes have been made since version 09

Migration to a Template based framework. This affects all elements. The framework has a template definition language.

Basename is split into two aspects. The first is version which applies to Templates. The second is checkpointing which applies to specific sections only.

Rule was inside Policy and now is Rule-Template and stands as a peer structure to Policy.

Types, e.g. Descriptor Types, Action Types, etc., are now templates that have no values filled in.

The embedded rule has been replaced by a template that has no predefined variables. All rules, pre-configured or embedded, are realized as Policy instantiations.

The Unassigned DPN is used to track requests vs. those that are installed, i.e. Agent assignment of Policy is supported.

The Topology system supports selection information by ServiceGroup or ServiceEndpoint.

DPN Peer Groups and DPN Groups are now PeerServiceGroup and ServiceGroup.

Bulk Configuration and Configuration now follow a style similar to YANG Patch. Agents MAY response back with edits it made to complete the Client edit request.

RFC 5777 Classifiers have been added.

All operations have a common error format.

C.2. Changes since Version 10

The following changes have been made since version 10

Sevice-Endpoints eliminated. Service-Group and DPN interfaces changed to hold information previously held by Service-Endpoint as noted in ML during IETF 101.

Service-Group resides under the Topology-Information-Mode

The Domain now has a checkpoint and the Topology Information Model checkpoint was removed to avoid any overlaps in checkpoints.

Scrubbed YANG for NMDA compliance and Guidelines (RFC 6087bis).

Monitor lifecycle, policy and policy installation examples added.

Authors’ Addresses

Satoru Matsushima
SoftBank
1-9-1,Higashi-Shimbashi,Minato-Ku
Tokyo 105-7322
Japan

Email: satoru.matsushima@g.softbank.co.jp

Lyle Bertz
6220 Sprint Parkway
Overland Park KS, 66251
USA

Email: lylebe551144@gmail.com

Marco Liebsch
NEC Laboratories Europe
NEC Europe Ltd.
Kurfuersten-Anlage 36
D-69115 Heidelberg
Germany

Phone: +49 6221 4342146
Email: liebsch@neclab.eu

Sri Gundavelli
Cisco
170 West Tasman Drive
San Jose, CA 95134
USA

Email: sgundave@cisco.com

Danny Moses

Email: danny.moses@intel.com

Charles E. Perkins
Futurewei Inc.
2330 Central Expressway
Santa Clara, CA 95050
USA

Phone: +1-408-330-4586
Email: charliep@computer.org

Distributed Mobility Management (DMM)
Internet-Draft
Intended status: Standards Track
Expires: October 7, 2016

J. Lee
Sangmyung University
Z. Yan
CNNIC

April 5, 2016

Deprecated Network Prefix Provision
draft-jhlee-dmm-dnpp-01

Abstract

This document introduces new extensions to router advertisement and router solicitation messages. The extensions are used to provide a mobile node's deprecated network prefix information to an access router. This document updates [RFC4861].

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

Router advertisement and router solicitation messages defined in [RFC4861] are used during stateless IPv6 autoconfiguration. A mobile node listens for router advertisement messages that are periodically sent by access routers on the local link or are explicitly requested by the mobile node using a router solicitation message. The router advertisement message contains information to allow the mobile node configures a global unicast IPv6 address. The provided information by the router advertisement message is, for instance, network prefix(es), default router address(es), hop limit, etc.

The router advertisement message is used by an access router to provide the network prefix information required for stateless IPv6 autoconfiguration, whereas the router solicitation message is used by a mobile node to quickly receive the router advertisement message from the access router. In other words, the current specification of Neighbor Discovery for IP version 6 does not specifies how the access router obtains the deprecated network prefix information (e.g., previous network prefix information) from the mobile node keeping deprecated IPv6 address(es) that are for instance global unicast IPv6 address(es) generated and used at previous access networks.

This document introduces new extensions to router advertisement and router solicitation messages to allow a mobile node provides its deprecated network prefix information to an access router.

2. Motivation

A mobile node changes its point of attachment from a previous network to a new network while keeping its global unicast IPv6 address for communications with a correspondent node. At the new network the mobile node’s global unicast IPv6 address previously configured at the previous network becomes a deprecated address. The mobile node configures a new global unicast IPv6 address at the new network and uses the new address for new communications. The mobile node also may use the deprecated address (i.e., the previously configured address at the previous network) for the communications with the correspondent node.

Nowadays ingress filtering is widely used to prevent source address spoofing. In this case, when the mobile node sends packets with the deprecated address as a source address at the new network, the packets will be filtered unless a rule of ingress filtering is updated in advance.

An access router at the new network may need to obtain the mobile node’s deprecated network prefix information. For instance, the new access router needs to establish a bidirectional tunnel with the previous access router of the mobile node for the communications associated with the deprecated address of the mobile node [Paper-Distributed.Mobility].

3. Option Formats

A router solicitation message is extended to include the deprecated network prefix information. A router advertisement message is extended to include a flag that requests the mobile node to send a router solicitation message including the deprecated network prefix information.

3.1. Deprecated Network Prefix Provision in Router Solicitations

A new flag is defined to indicate that a router solicitation message includes the deprecated network prefix information of a mobile node in the prefix information option.

```
 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+
|L|A|D|Reserved1|
+-+-+-+-+-+-+-+-+
```

D: 1-bit "Deprecated network prefix" flag. It indicates that the prefix included in this prefix information option is the deprecated network prefix of the mobile node.

When the mobile node wants to send the deprecated prefix to the new access router, the prefix information option is used to carry the deprecated network prefix and the D flag is set to 1.

3.2. Deprecated Network Prefix Request in Router Advertisements

A new flag is defined to request the deprecated network prefix information in a router solicitation message.

```
 8 9 0 1 2 3 4 5
+-+-+-+-+-+-+-+-+
|M|O|D|Reserved |
+-+-+-+-+-+-+-+-+
```

D: 1-bit "Deprecated network prefix" flag. It indicates that the deprecated network prefix of the mobile node is needed by an access router.

When a mobile node receives the router advertisement message containing the D flag set to 1, the mobile node should respond with a router solicitation message carrying the prefix information option and with D flag set to 1 in the option. In the prefix information option, the deprecated network prefix of the mobile node is contained.

After the new access router learns the previous prefix of the specific mobile node, it updates the rule of ingress filter. Afterwards, the D flag in the following router advertisement message sent to this mobile node will be set to 0 and the mobile node will recognize that its previous prefix has been recorded by the access router. Then the mobile node will not feedback with a router solicitation message.

This extension is not supported by the access router if the D flag is not included in the router advertisement message and then the mobile node should not send a router solicitation message accordingly.

4. Security Considerations

TBD

5. IANA Considerations

TBD

6. References

6.1. Normative References

[RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, DOI 10.17487/RFC4861, September 2007, <http://www.rfc-editor.org/info/rfc4861>.

6.2. Informative References

[Paper-Distributed.Mobility] Lee, J., Bonnin, J., Seite, P., and H. Chan, "Distributed IP Mobility Management from the Perspective of the IETF: Motivations, Requirements, Approaches, Comparison, and Challenges", IEEE Wireless Communications, October 2013.

[RFC7333] Chan, H., Ed., Liu, D., Seite, P., Yokota, H., and J. Korhonen, "Requirements for Distributed Mobility Management", RFC 7333, DOI 10.17487/RFC7333, August 2014, <http://www.rfc-editor.org/info/rfc7333>.

Authors' Addresses

Jong-Hyouk Lee
Sangmyung University
31, Sangmyeongdae-gil, Dongnam-gu
Cheonan 31066
Republic of Korea

Email: jonghyouk@smu.ac.kr

Zhiwei Yan
CNNIC
No.4 South 4th Street, Zhongguancun
Beijing 100190
China

Email: yanzhiwei@cnnic.cn

Network Working Group V.Liu
Internet Draft ChinaMobile
Intended status: Standards Track D.Liu
Expires: Mar 18, 2016 Alibaba
H. Chan
Huawei Technologies
H. Deng
China Mobile
X.Wei
Huawei Technologies
October 19, 2015

Distributed mobility management deployment scenario and architecture
draft-liu-dmm-deployment-scenario-05

Status of this Memo

Copyright Notice

Abstract

This document discusses the deployment scenario of distributed mobility management. The purpose of this document is to trigger the discussion in the group to understnad the DMM deployment scenario and consideration from the operator's perspective.

Table of Contents

1. Introduction

Distributed mobility management aims at solving the centralized mobility anchor problems of the traditional mobility management protocol. The benefit of DMM solution is that the data plane traffic does not need to traverse the centralized anchoring point. This document discusses the potential deployment scenario of DMM. The purpose of this document is to help the group to reach consensus regarding the deployment model of DMM and then develop the DMM solution based on the deployment model.

2. Conventions used in this document

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].

2.1. Terminology

All the general mobility-related terms and their acronyms used in this document are to be interpreted as defined in the Mobile IPv6 base specification [RFC6275], in the Proxy mobile IPv6 specification [RFC5213], and in Mobility Related Terminology [RFC3753]. These terms include the following: mobile node (MN), correspondent node (CN), and home agent (HA) as per [RFC6275]; local mobility anchor (LMA) and mobile access gateway (MAG) as per [RFC5213], and context as per [RFC3753].

In addition, this draft introduces the following terms.

Location information (LI) function

is the logical function that manages and keeps track of the internet work location information of a mobile node which may change its IP address as it moves. The information may associate with each session identifier, the IP routing address of the MN, or of a node that can forward packets destined to the MN.

Forwarding management (FM)

is the logical function that intercepts packets to/from the IP address/prefix delegated to a mobile node and forwards them, based on internetwork location information, either directly towards their destination or to some other network element that knows how to forward the packets to their ultimate destination. With data plane and control plane separation, the forwarding management may be separated into a data-plane forwarding management (FM-DP) function and a control-plane forwarding management (FM-CP) function.

3. Deployment Scenario and Model of DMM

As discussed in the DMM requirement document, the centralized mobility management has several drawbacks. The main problem of the centralized mobility management protocols is that all the traffic need to anchor to a centralized anchor point. This approach does not cause any problem in current mobile network deployment but in the scenario that will be discussed later in this document, centralized mobility management protocols will have many drawbacks and it is believed that DMM is more suitable in that scenario.

The main deployment scenario discussed in this document is divided into three scenarios. The first one is the network function virtualization scenario. In this scenario, the mobile core network’s control plane function is centralized in the mobile cloud. Apparently, deploying the data plane function also in the same centralized mobile cloud is not optimized from the traffic forwarding’s perspective. For the control plane The MME and PGW-F are implemented by NFV. For the dataplane the PGW-F/SGW-F can weither be implemented by NFV or lagacy devices. The second deployment scenario is the SIPTO/LIPA scenario which is discussed in 3GPP. In this scenario, DMM can provide optimized traffic offloading solution. The Third deploy scenario is the WLAN scenario. In this scenario, the AC is implemented in the cloud and the authentication status can maintained as the terminal move from one AP to another.

4. Network Function Virtualization Scenario

This section discusses network function virtualization scenario, the associated control - data plane separation and the possible mobility management functions to support this scenario.

4.1. Network function virtualization deployment architecture

The network function virtualization scenario is shown in Figure 1.

```
                    Mobile Cloud
             ...........
                  (’             ’)
                (                )))
               (((  +-----------+ )))
              ((    |Mobile Core|    )))
              ((    |MME+P/SGW-C|    )))
              (((   +-----------+   )))
                 (’..............’)
                                  |
                          | IP Transit Network
                      (.........)
                    (           )) MN-Internet communication
                   (           ^ ))
            ^ > > >(           ^  ))> > > > > > > > >
            ^       ((         ^  )                  v
            ^        (.........^.)                   v
            ^ +-------|     |  ^|                    v
            ^ |             |  ^+--------------+     v
            ^ |             |  < <             |     v  MN-MN communication
            ^ |             |    ^             |     v
      +--------------+    +--------------+    +--------------+
      |Access Network|    |Access Network|    |Access Network|
      |  PGW-F/SGW-F  |   |  PGW-F/SGW-F  |   |  PGW-F/SGW-F  |
      +--------------+    +--------------+    +--------------+
            ^                    ^                   v
            ^                    ^                   v
        +---------+         +---------+         +---------+
        |   MN     |        |    MN   |          |   MN     |
        +---------+         +---------+         +---------+
```

Figure 1: Network function virtualization deployment architecture

In this architecture, the mobile core include MME and PGW-F is located in the cloud data center, which can be the operator’s private cloud using NFV. The access network cantains PGW-F/SGW-F is connected through an IP transit network. The PGW-F/SGW-F may also implement by NFV of small data center in convergence layer. The architecture of NFV based Mobile Core is shown in Figure 2.

```
   ---------------------------------     -------------------
   | ----------------------------- |     |   MANO LAYER     |
   | |        OSS/BSS LAYER      | |     |                  |
   | ----------------------------- |     |   -------------  |
   |               |               |     |   |Orchestrator| |
   | ----------------------------- |     |   -------------  |
   | |      VNF LAYER            | |     |         |        |
   | | ---------       ------    | |     |   -------------  |
   | | |P/SGW-C|      |  MME |   | |-----|  |    VNFM    |  |
   | | ---------         ------    |     |   -------------  |
   | ----------------------------- |     |        |         |
   |               |               |     |        |         |
   | ----------------------------- |     |   -------------  |
   | |      NFVI LAYER           | |-----|  |     VIM     | |
   | ----------------------------- |     |   -------------  |
   ---------------------------------     -------------------
```

Figure 2: NFV based Mobile Core Architecture

In Figure 2, the MANO layer contains Orchestrator, VNFM and VIM. The Orchestrator is in charge of top-down service and source monitor and fulfillment. VNFM is incharge of manage the VNFs. And VIM normally is the Openstack which provide management of the whole virtualization layer.

4.2. Control and data plane separation

The cloud based mobile core network architecture implies separation of the control and data planes. The control plane is located in the cloud and the data plane should be distributed. Otherwise, all the data traffic will go through the cloud which is obviously not optimized for the mobile node to mobile node communication. For the mobile node to Internet communication, the Internet access point is normally located in the metro IP transit network. In this case, the mobile node to Internet traffic should also go through the Internet access point instead of the mobile core in the cloud.

However, in some deployment scenario, the operator may choose to put the mobile core cloud in the convergence layer of IP metro network. In this case, the Internet access point may co-located with the mobile core cloud. In this case, the mobile node to Internet traffic may go through the mobile core cloud.

4.3. Mobility management architecture

Since the control plane and data plane are separated and the data plane is distributed, traditional mobility management cannot meet this requirement. Distributed mobility management or SDN based mobility management may be used in this architecture to meet the traffic forwarding requirement (e.g. MN to MN and MN to Internet traffic should not go through from the mobile core cloud.). The traditional mobility management functions is not separating the data plane from the control plane. Basic mobility management functions include location information (LI) function and Forwarding management (FM). The former is a control plane function. The latter can be separated into data plane forwarding management (FM-DP) and control plane forwarding management (FM-CP).

The data plane function is FM-DP, while the control plane functions include FM-CP and LI. Then the control plane functions in the cloud-based mobile core includes LI and FM-CP. They are of cause other functions in the control plane such as policy function. The

distributed data plane may have multiple instances of FM-DP in the network.

```
                core network controller

                     +---------+
                     |LI, FM-CP|
                     +---------+

      +-------+        +-------+        +-------+
      | FM-DP |        | FM-DP |        | FM-DP |
      +-------+        +-------+        +-------+
```

Figure 2: Mobility management functions with data plane - control plane separation under one controller When the control of the access network is separate from that of the core, there will be separate controllers as shown in Figure 3.

```
 Access network controller            Core network controller

          +---------+                         +---------+
          |LI, FM-CP|                         |LI, FM-CP|
          +---------+                         +---------+

    +-------+       +-------+        +-------+       +-------+
    | FM-DP |       | FM-DP |        | FM-DP |       | FM-DP |
    +-------+       +-------+        +-------+       +-------+
```

Figure 2: Mobility management functions with data plane - control plane separation with separate control in core and in access networks.

4.4. NFV based deployment architecture

Here is the deployment architecture in NFV.

```
             -------------             ------------
            | Yang Model  |           | Tosca Model|
             -------------             ------------
-------------------------------------------------------------------------
| [VNFO]              -------------------------------------------------- |
|  ------------    |  --------------------   ------------------------ ||
| |    NBI     |   | |  ----- ----- ----- | | ---------------------- |||
|  ------------    | |  |NSD| |FGD| |VLD| | ||VNFD(LI,FM-CP)(FD-DP)  |||
|                  | |  ----- ----- ----- | | ---------------------- |||
|  ------------    |  --------------------   ------------------------ ||
| | API Router|    |                  DataBase                        ||
|  ------------     ---------------------------------------------------|
|                                                                      |
|  ------------                                                        |
| |Core Engine|                                                        |
|  ------------                                                        |
-------------------------------------------------------------------------
      ------------     ------------    ------------
     |VNFM Driver|     |VIM Driver|     |PNF Driver|
      ------------     ------------    ------------
 ----------------------------------------------     -----------------
|   [VNFM]                                     |    |[PNF]           |
|   VNF Life cycle management;                 |    |   ------------ |
|   VNF configuration;                         |    |   |   OVS    | |
|   VNF update;                                |    |   ------------ |
|   VNF status monitor;                        |     -----------------
|   VNF Auto healing/Scale in/Scale out        |
 ----------------------------------------------
-----------------------------------------------------------------
|                           Vim                                  |
-----------------------------------------------------------------
   --------------------------------------------------
   |[VNF]  LI Slicing ; FM-CP Slicing; FD-DP Slicing |
   --------------------------------------------------
```

Figure 3 Deployment architecture

5. SIPTO deployment scenario

The Second deployment scenario is the SIPTO scenario which is discussed in 3GPP. DMM is believed to be able to provide dynamic anchoring. It allows the mobile node to have several anchoring points and to change the anchoring point according to the application requirement. In SIPTO scenario, the gateway function is located very near to the access network and to the user. If using current centralized mobility management, the traffic will need to tunnel back to the previous anchor point even when the mobile node has changed the point of attachment to a new one. Figure 3 shows the architecture of SIPTO.

```
                         +---------+
                         |   GGSN  |
                         +---------+
                             |
                             |
           ------------------------------------------
           |                 |             |         |
       +---------+    +---------+    +---------+   +---------+
       |   MSC   |    |   SGSN  |    |   MSC   |   |   SGSN   |
       +---------+    +---------+    +---------+   +---------+
           |             |    |         |    |          |
             ------------      ----------      ---------
                  |            |                  |
         <<<<<<<<<|     <<<<<<<|                  |>>>>>>>>
                +---------+    +---------+     +---------+
                |   GGSN  |    |   GGSN   |   |   GGSN   |
                |    RNC  |    |    RNC   |   |    RNC   |
                +---------+    +---------+     +---------+
                   ^|                 |^            ^|
                   ^|                 |^            ^|
             ---------------          |^            ^|
            ^|             |^         |^            ^|
       +---------+     +---------+    +---------+   +---------+
       |  NodeB  |     |  NodeB  |    |  NodeB  |   |  NodeB  |
       +---------+     +---------+    +---------+   +---------+
           ^|             ^|                            ^|
       +---------+        +---------+              +---------+
       | Terminal|   ->   | Terminal|    -> ->     | Terminal|
       +---------+        +---------+              +---------+
```

Figure 4 SIPTO Scenario

6. WLAN deployment scenario

The Third deployment scenario is the WLAN scenario. DMM can enable the AC in the cloud. The cloud AC and maintain the authentication and connection status. As the terminal move from one AP to another, it still can have the connection.

```
                    ...........
                  ('             ')
               (((  +-----------+ )))
              ((    |  Mobile AC |   )))
              (((   +-----------+   )))
                 ('..............')
                          |
                          | IP Transit Network
                      (.........)
                    (          ))
                   (            ))
                   (             ))
                   ((           )
                    (..........)
            +-----------------------------------+
            |                |                      |
            |                |                      |
            |                |                      |
      +----------+        +-----------+        +----------+
      |     AP     |        |     AP     |        |     AP     |
      +----------+        +-----------+        +----------+
         |                             |             |
      +---------+        +---------+            +---------+
      | Terminal|   ->   | Terminal|    ->     | Terminal|
      +---------+         +---------+           +---------+
```

Figure 5 WLAN deployment scenario

7. Conclusion

This document discusses the deployment scenario of DMM. Three types of deployment scenario is discussed in this document. Further types of deployment scenario can be added to this document according to the progress of the group’s discussion.

8. Security Considerations

N/A

9. IANA Considerations

N/A

10. Normative References

[1] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[2] Crocker, D. and Overell, P.(Editors), "Augmented BNF for Syntax Specifications: ABNF", RFC 2234, Internet Mail Consortium and Demon Internet Ltd., November 1997.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2234] Crocker, D. and Overell, P.(Editors), "Augmented BNF for Syntax Specifications: ABNF", RFC 2234, Internet Mail Consortium and Demon Internet Ltd., November 1997.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC3753] Manner, J. and M. Kojo, "Mobility Related Terminology", RFC 3753, June 2004.

[RFC5213] Gundavelli, S., Leung, K., Devarapalli, V., Chowdhury, K., and B. Patil, "Proxy Mobile IPv6", RFC 5213, August 2008.

[RFC6275] Perkins, C., Johnson, D., and J. Arkko, "Mobility Support in IPv6", RFC 6275, July 2011.

11. Informative References

[3] Faber, T., Touch, J. and W. Yue, "The TIME-WAIT state in TCP and Its Effect on Busy Servers", Proc. Infocom 1999 pp. 1573-1583.

[Fab1999] Faber, T., Touch, J. and W. Yue, "The TIME-WAIT state in TCP and Its Effect on Busy Servers", Proc. Infocom 1999 pp. 1573-1583.

12. Acknowledgments

This document was prepared using 2-Word-v2.0.template.dot.

Authors' Addresses

Vic Liu
China Mobile
32 Xuanwumen West AVE, Xicheng, Beijing
Email: liuzhiheng@chinamobile.com

Dapeng Liu
Alibaba

Email: max@dotalks.com

H Anthony Chan
Huawei Technologies
5340 Legacy Dr. Building 3
Plano, TX 75024
USA
Email: h.a.chan@ieee.org

Hui Deng
China Mobile
32 Xuanwumen West AVE, Xicheng, Beijing
Email: denglingli@chinamobile.com

Xinpeng Wei
Huawei Technologies

Email: Xinpengwei@huawei.com

IETF P. McCann, Ed.
Internet-Draft J. Kaippallimalil, Ed.
Intended status: Informational Huawei
Expires: October 13, 2016 April 11, 2016

Communicating Prefix Cost to Mobile Nodes
draft-mccann-dmm-prefixcost-03

Abstract

In a network implementing Distributed Mobility Management, it has been agreed that Mobile Nodes (MNs) should exhibit agility in their use of IP addresses. For example, an MN might use an old address for ongoing socket connections but use a new, locally assigned address for new socket connections. Determining when to assign a new address, and when to release old addresses, is currently an open problem. Making an optimal decision about address assignment and release must involve a tradeoff in the amount of signaling used to allocate the new addresses, the amount of utility that applications are deriving from the use of a previously assigned address, and the cost of maintaining an address that was assigned at a previous point of attachment. As the MN moves farther and farther from the initial point where an address was assigned, more and more resources are used to redirect packets destined for that IP address to its current location. The MN currently does not know the amount of resources used as this depends on mobility path and internal routing topology of the network(s) which are known only to the network operator. This document provides a mechanism to communicate to the MN the cost of maintaining a given prefix at the MN's current point of attachment so that the MN can make better decisions about when to release old addresses and assign new ones.

Status of This Memo

This Internet-Draft will expire on October 13, 2016.

Copyright Notice

Table of Contents

1. Introduction

Previous discussions on address agility in distributed mobility management have focused on "coloring" prefixes with one of a small number of categories, such as Fixed, Sustained, or Nomadic. The assumption here is that the MN should use a permanent home address for sessions that need a persistent IP address, and a local, ephemeral address for short-lived sessions such as browsing. However, a small set of address categories lacks expressive power and leads to false promises being made to mobile nodes. For example, the concept that a home address can be maintained permanently and offered as an on-link prefix by any access router to which the MN may be attached in future is simply not attainable in the real world. There will always exist some access routers that do not have arrangements in place with the home network to re-route (via tunneling or other mechanisms) the home prefix to the current point of attachment.

Conversely, the assumption that a Nomadic prefix will never be available to an MN after it changes its current point of attachment is too limiting. There is no reason why an MN should not be able to keep a prefix that was assigned by a first network after it moves to a second network, provided that measures are put in place to re-route such prefixes to the new attachment point.

Rather, this document argues that there is in reality a continuum of cost associated with an address as the MN moves from one attachment point to another or from one network to another. The sources of the cost are the increased latency, network bandwidth, and network state being maintained by a network-based mobility management scheme to route packets destined to the prefix to the MN’s current point of attachment. By communicating this cost to the MN every time its attachment point changes, the MN can make intelligent decisions about when to release old addresses and when to acquire new ones.

The cost should be communicated to the MN because of several constraints inherent in the problem:

(1) The MN is the entity that must make decisions about allocating new addresses and releasing old ones. This is because only the MN has the information about which addresses are still in use by applications or have been registered with other entities such as DNS servers.

(2) Only the network has information about the cost of maintaining the prefix in a network-based mobility management scheme, because the MN cannot know the network topology that gives rise to the inefficiencies.

If the cost of maintaining a prefix is not made available to the mobile node, it may attempt to infer the cost through heuristic mechanisms. For example, it can measure increased end-to-end latency after a mobility event, and attribute the increased latency to a longer end-to-end path. However, this method does not inform the MN about the network bandwidth being expended or network state being maintained on its behalf. Alternatively, a MN may attempt to count mobility events or run a timer in an attempt to guess at which older prefixes are more costly and in need of being released. However, these methods fail because the number of mobility events is not an indication of how far the MN has moved in a topological sense from its original attachment point which is what gives rise to the costs outlined above. Re-allocating an address upon expiration of a timer may introduce uneccessary and burdensome signaling load on the network and air interface.

1.1. 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 [1].

1.2. Abbreviations

ANDSF	Access Network Discovery and Selection Function
MN	Mobile Node
MPTCP	Multi-Path Transmission Control Protocol
ND	Neighbor Discovery
NGMN	Next Generation Mobile Networks
NUD	Neighbor Unreachability Detection
OMA-DM	Open Mobile Alliance - Device Management
PIO	Prefix Information Discovery
PGW	Packet data network Gateway
SeND	Secure Neighbor Discovery
SGW	Serving Gateway

2. Motivation

The Introduction speaks in general terms about the cost of a prefix. More specifically, we are talking about the aggregate amount of state being maintained in the network on behalf of the mobile node in addition to the transport resources being used (or wasted) to get packets to the MN’s current point of attachment.

In a non-mobile network, the addresses can be assigned statically in a manner that is aligned with the topology of the network. This means that prefix aggregation can be used for maximum efficiency in the state being maintained in such a network. Nodes deep in the network need only concern themselves with a small number of short prefixes, and only nodes near the end host need to know longer more specific prefixes. In the best case, only the last-hop router(s) need to know the actual address assigned to the end host. Also, routing protocols ensure that packets follow the least-cost path to the end host in terms of number of routing hops or according to other policies defined by the service provider, and these routing paths can change dynamically as links fail or come back into service.

However, mobile nodes in a wide-area wireless network are often handled very differently. A mobile node is usually assigned a fixed gateway somewhere in the network, either in a fixed central location or (better) in a location near where the MN first attaches to the network. For example, in a 3GPP network this gateway is a PGW that can be allocated in the home or visited networks. Initially, the cost of such a prefix is the state entry in the fixed gateway plus any state entries in intermediate tunneling nodes (like SGWs) plus whatever transport resources are being used to get the packet to the MN’s initial point of attachment.

When an MN changes its point of attachment, but keeps a fixed address, the cost of the prefix changes (usually it increases). Even if the fixed gateway was initially allocated very close to the initial point of attachment, as the MN moves away from this point, additional state must be inserted into the network and additional transport resources must be provided to get the packets to the current point of attachment. For example, a new SGW might be allocated in a new network, and now the packets must traverse the network to which the MN first attached before being forwarded to their destination, even though there may be a better and more direct route to communication peers from the new network. Whatever aggregation was possible at the initial point of attachment is now lost and tunnels must be contructed or holes must be punched in routing tables to ensure continued connectivity of the fixed IP address at the new point of attachment. Over time, as the MN moves farther and farther from its initial point of attachment, these costs can become large. When summed over millions of mobile nodes, the costs can be quite large.

Obviously, the assignment of a new address at a current point of attachment and release of the older, more costly prefix will help to reduce costs and may be the only way to meet emerging more stringent latency requirements [8]. However, the MN does not in general know the current cost of a prefix because it depends on the network topology and the number of handovers that have taken place and whether these handovers have caused the MN to transition between different topological parts of the network. It is the purpose of the protocol extension defined in this document to communicate the current cost of a prefix to the MN so that it can make intelligent decisions about when to get a new address and when to release older addresses. Only the MN can make a decision about when to release an address, because it is the only entity that knows whether applications are still listening waiting to receive packets at the old address.

Section 4 describes MN behavior when Router Advertisements with Prefix Cost is received.

3. Prefix Cost Sub-option

This document defines a prefix cost option to be carried in router advertisements. It is a sub-option that carries meta-data as defined by Korhonen et al. [7]

```
      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
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |     TBD1      |       1       |C|         Reserved1           |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |           Prefix Cost         |         Reserved2             |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 1: Prefix Cost suboption

The prefix cost is carried as a 16-bit, unsigned number in network byte order. An higher number indicates an increased cost.

This sub-option is appended in Router Advertimsement messages that are sent on a periodic basis. No additional signaling cost is incurred to support this mechanism.

It should be noted that link layer events do not cause a change in the prefix cost.

The prefix cost is for a connection segment. No end-to-end congestion or flow control mechanisms are implied with this cost.

4. Host Considerations

Prefix Cost in a Router Advertisement PIO serves as a hint for the MN to use along with application knowledge, MN policy configuration on network cost and available alternative routes to determine the IP addresses and routes used. For example, if the application is downloading a large file, it may want to maintain an IP address and route until the download is complete. On the other hand, some applications may use multiple connections (e.g., with MPTCP) and may not want to maintain an IP address above a configured cost. It could also be the case that the MN maintains the IP address even at high cost if there is no alternative route/address. These decisions are made based on configured policy, and interaction with applications, all of which are decided by the MN.

When the MN is ready to release an IP address, it may send a DHCPv6 [5] Release message. The network may also monitor the status of a high cost connection with Neighbor Unreachability Detection (NUD) [2], [6], and determine that an address is not used after the NUD times out. The network should not continue to advertise this high cost route following the explicit release of the address or NUD timeout. It can initiate the release of network resources dedicated to providing the IP address to the MN.

The operator of the network or host’s service provider can configure policy that determines how the host should handle the prefix cost values. In a 3GPP network, the subscription provider may configure policies in the host via OMA-DM or S14 (ANDSF). For example, the service provider may configure rules to state that prefix cost values below 500 indicate low cost and ideal access network conditions, values from 501 - 5000 indicate that the host should try to relocate connections, and values above 5000 indicate a risk and impending loss of connectivity. The policies themselves can be (re-)configured as needed by the operator. Prefix cost information with each Router Advertisement allows the host to interpet a simple number and associated policies to (re-)select optimal routes. For networks service providers, when this cost is associated with charging, it can be a valuable tool in dynamically managing the utilization of network resources.

This draft does not aim to provide definitive guidance on how an OS or application process receives indications as a result of prefix cost option being conveyed in Router Advertisements. Only high level design options are listed here. New socket options or other APIs can be used to communicate the cost of an address in use on a given connnection. For example, a new "prefix-cost" socket option, if set, can indicate that the application is interested in being notified when there is a change in the prefix cost. The actual mechanisms used to either notify or other means of busy polling on this change of prefix cost information need to be specified in other drafts. An alternative to the application discovering the changed prefix cost is to use a model where a connection manager handles the interface between the network and the application (e.g., Android Telephony Manager [9]). In this case, the connection manager is responsible to select and manage addresses based on policies (configured via OMA-DM or S14) and prefix cost obtained from the Router Advertisements.

5. Security Considerations

Security of the prefix cost option in the PIO needs to be considered. Neighbor Discovery (ND) and Prefix Information Option (PIO) security are described in [2] and [3]. A malicious node on a shared link can advertise a low cost route in the prefix cost option and cause the MN to switch. Alternatively, an incorrect higher cost route in the prefix cost option can result in the suboptimal use of network resources. In order to avoid such on-link attacks, SeND [4] can be used to reject Router Advertisements from nodes whose identities are not validated.

6. IANA Considerations

This memo defines a new Prefix Information Option (PIO) sub-option in Section 3.

7. References

7.1. Normative References

[1] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[2] Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, DOI 10.17487/RFC4861, September 2007, <http://www.rfc-editor.org/info/rfc4861>.

[3] Draves, R. and D. Thaler, "Default Router Preferences and More-Specific Routes", RFC 4191, DOI 10.17487/RFC4191, November 2005, <http://www.rfc-editor.org/info/rfc4191>.

[4] Arkko, J., Ed., Kempf, J., Zill, B., and P. Nikander, "SEcure Neighbor Discovery (SEND)", RFC 3971, DOI 10.17487/RFC3971, March 2005, <http://www.rfc-editor.org/info/rfc3971>.

[5] Droms, R., Ed., Bound, J., Volz, B., Lemon, T., Perkins, C., and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3315, DOI 10.17487/RFC3315, July 2003, <http://www.rfc-editor.org/info/rfc3315>.

[6] Nordmark, E. and I. Gashinsky, "Neighbor Unreachability Detection Is Too Impatient", RFC 7048, DOI 10.17487/RFC7048, January 2014, <http://www.rfc-editor.org/info/rfc7048>.

7.2. Informative References

[7] Korhonen, J., Gundavelli, S., Seite, P., and D. Liu, "IPv6 Prefix Properties", draft-korhonen-dmm-prefix-properties-05 (work in progress), February 2016.

[8] NGMN Alliance, "NGMN 5G Whitepaper", February 2015.

[9] Android Telephony Developer’s Forum, http://developer.android.com/reference/android/telephony/TelephonyManager.html, "Android Telephony Manager".

Authors’ Addresses

Peter J. McCann (editor)
Huawei
400 Crossing Blvd, 2nd Floor
Bridgewater, NJ 08807
USA

Phone: +1 908 541 3563
Email: peter.mccann@huawei.com

John Kaippallimalil (editor)
Huawei
5340 Legacy Dr., Suite 175
Plano, TX 75024
USA

Email: john.kaippallimalil@huawei.com

```
DMM WG                                                          P. Seite
Internet-Draft                                                    Orange
Intended status: Standards Track                                A. Yegin
Expires: April 21, 2016                                          Samsung
                                                           S. Gundavelli
                                                                   Cisco
                                                        October 19, 2015
```

MAG Multipath Binding Option
draft-seite-dmm-rg-multihoming-02.txt

Abstract

The document [RFC4908] proposes to rely on multiple Care-of Addresses (CoAs) capabilities of Mobile IP [RFC6275] an Network Mobility (NEMO; [RFC3963]) to enable Multihoming technology for Small-Scale Fixed Networks. In the continuation of [RFC4908], this document specifies a multiple proxy Care-of Addresses (pCoAs) extension for Proxy Mobile IPv6 [RFC5213]. This extension allows a multihomed Mobile Access Gateway (MAG) to register more than one proxy care-of-address to the Local Mobility Anchor (LMA).

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

Using several links, the multihoming technology can improve connectivity availability and quality of communications; the goals and benefits of multihoming are as follows:

- o Redundancy/Fault-Recovery

- o Load balancing

- o Load sharing

- o Preferences settings

According to [RFC4908], users of Small-Scale Networks can take benefit of multihoming using mobile IP [RFC6275] and Network Mobility (NEMO) [RFC3963] architecture in a mobile and fixed networking environment. This document was introducing the concept of multiple Care-of Addresses (CoAs) that have been specified since then [RFC5648].

In the continuation of [RFC4908], a Proxy Mobile IPv6 [RFC5213] based multihomed achitecture could be defined. The motivation to update [RFC4908] with proxy Mobile IPv6 is to leverage on latest mobility working group achievments, namely:

- using GRE as mobile tuneling, possibly with its key extension [RFC5845] (a possible reason to use GRE is given on Section 3.2).

- using UDP encapsulation [RFC5844] in order to support NAT traversal in IPv4 networking environment.

- Prefix Delegation mechanism [RFC7148].

Proxy Mobile IPv6 (PMIPv6) relies on two mobility entities: the mobile access gateway (MAG), which acts as the default gateway for the end-node and the local mobility anchor (LMA), which acts as the topological anchor point. Point-to-point links are established, using IP-in-IP tunnels, between MAG and LMA. Then, the MAG and LMA are distributing traffic over these tunnels. All PMIPv6 operations are performed on behalf of the end-node and its corespondent node, it thus makes PMIPv6 well adapted to multihomed architecture, as considered in [RFC4908]. Taking the LTE and DSL networking environments as an example, the PMIPv6 based multihomed architecture is depicted on Figure 1. Flow-1,2 and 3 are distributed either on Tunnel-1 (over LTE) or Tunnel-2 (ober DSL), while Flow-4 is spread on both Tunnel-1 and 2.

```
     Flow-1
      |
      |Flow-2             _----_
      | |         CoA-1  _(      )_   Tunnel-1
      | |    .---=======(   LTE    )========\ Flow-1
      | |    |          (_     ,_)          \Flow-4
      | |    |           '----'             \
      | | +=====+                              \  +=====+    _----_
      | '-|     |                               \ |     |  _(      )_
      '---| MAG |                                 | LMA |-( Internet )--
      .---|     |                                 |     |  (_     ,_)
      | .-|     |                               / |     |    '----'
      | | +=====+                              /  +=====+
      | |    |          _----_                /
      | |    |    CoA-2  _(      )_ Tunnel-2 /
      | |    .---=======(   DSL    )========/ Flow-2
      | |               (_      _)         Flow-3
      | |                '----'          Flow-4
      |Flow-3
      |
     Flow0=-4
```

Figure 1: Multihomed MAG using Proxy Mobile IPv6

Current version of Proxy Mobile IPv6 does not allow a MAG to register more than one proxy Care-of-Adresse to the LMA. In other words, only one MAG/LMA link, i.e. IP-in-IP tunnel, tunnel can be used at the same time. This document overcome this limitation by defining the multiple proxy Care-of Addresses (pCoAs) extension for Proxy Mobile IPv6.

2. Conventions and Terminology

2.1. 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].

2.2. Terminology

All mobility related terms used in this document are to be interpreted as defined in [RFC5213], [RFC5844] and [RFC7148]. Additionally, this document uses the following terms:

IP-in-IP

IP-within-IP encapsulation [RFC2473], [RFC4213]

3. Overview

3.1. Example Call Flow

Figure 2 is the callflow detailing hybrid access support with PMIPv6. The MAG in this example scenario is equipped with both WLAN and LTE interfaces and is also configured with the MAG functionality. A logical-NAI with ALWAYS-ON configuration is enabled on the MAG. The mobility session that is created on the LMA is for the logical-NAI. The IP hosts MN_1 and MN_2 are assigned IP addresses from the delegated mobile network prefix.

```
   +=====+ +=====+        +=====+      +=====+      +=====+      +=====+
   | MN_1| | MN_2|        | MAG |      | WLAN|      | LTE |      | LMA |
   +=====+ +=====+        +=====+      +=====+      +=====+      +=====+
      |       |              |            |            |            |
      |       |              |            |            |            |
      |       |              | (1) ATTACH |            |            |
      |       |              | <--------> |            |            |
      |       |              | (2) ATTACH              |            |
      |       |              | <---------------------->|            |
      |       |              | (3) PBU (NAI, MAG-NAI, DMNP, MMB)    |
      |       |              | ------------------------*----------> |
      |       |              | (4) PBA (NAI, DMNP)                  |
      |       |              | <----------------------*----------- |
      |       |              | (5) TUNNEL INTERFACE CREATION        |
      |       |              |-============== TUNNEL ==*===========-|
      |       |              |                                      |
      |       |              | (6) PBU (NAI, MAG-NAI, DMNP, MMB)    |
      |       |              | -----------*-----------------------> |
      |       |              | (7) PBA (NAI, DMNP)                  |
      |       |              | <----------*------------------------ |
      |       |              | (8) TUNNEL INTERFACE CREATION        |
      |       |              |-===========*== TUNNEL ==============-|
      |   (9)                |                                      |
      | <------------------> |                                      |
      |       |  (10)        |                                      |
      |       |<-----------> |                                      |
```

Figure 2: Functional Separation of the Control and User Plane

3.2. Traffic distribution schemes

IP mobility protocols allow to establish the forwarding plane over the WAN interfaces of a multihomed RG. Then, traffic distribution schemes define the way to distribute data packets over these paths (i.e. IP tunnels). Traffic distribution can be managed either on a per-flow or on a per-packet basis:

- per-flow traffic management: each IP flow (both upstream and downstream) is mapped to a given mobile IP tunnel, corresponding to a given WAN interface. This scenario is based on IP flow mobility mechanism using the Flow binding extension [RFC6089]. The mobility anchor provides IP session continuity when an IP flow is moved from one WAN interfaces to another. The flow binding extension allows the IP mobility anchor and the RG to exchange, and synchronize, IP flow management policies (i.e. policy routing rules associating traffic selectors [RFC6088] to mobility bindings).

- Per-packet management: distribute the IP packets of a same IP flow, or of a group of IP flows, over more than one WAN interface. In this scenario, traffic management slightly differs from the default mobile IP behaviour; the mobility entities (mobility anchor and client) distribute packets, belonging to a same IP flow, over more than one bindings simultaneously. The definition of control algorithm of a Per-packet distribution scheme (how to distribute packets) is out the scope of this document. When operating at the packet level, traffic distribution scheme may introduce packet latency and out-of-order delivery. It may require the aggeregation entities (RG and mobility anchor) to be able to reorder (ans thus, to buffer) received packets before delivering. A possible implementation is to use GRE as mobile tunnelling mechanism, together with the GRE KEY option [RFC5845] to add sequence number to GRE packets, and so, to allow the receiver to perform reordering. However, more detailed buffering and reordering considerations are out of the scope of this document.

The traffic distribution scheme may require the RG and the to exchange interface metrics to make traffic steering decision.For example, the RG may sent its DSL synchronization rate to the mobility anchor, so that the latter can make traffic forwarding decision accordingly. In this case, the vendor specific mobility option [RFC5094] can be used for that purpose.

Per-flow and per-packet distribution schemes are not exclusive mechanisms; they can cohabit in the same hybrid access system. For example, High throughput services (e.g. video streaming) may benefit from per-packet distribution scheme, while some other may not. Typically VoIP application are sensitive to latency and thus should not be split over different WAN paths. In this situation, the aggregation entities (RG and mobility anchor) must exchange traffic management policies to associate distribution scheme, traffic and WAN interface (physical or virtual). [RFC6088] and [RFC6089] define traffic management on a flow basis but there is no such policy on a per packet basis.

4. Protocol Extensions

4.1. MAG Multipath-Binding Option

The MAG Multipath-Binding option is a new mobility header option defined for use with Proxy Binding Update and Proxy Binding Acknowledgement messages exchanged between the local mobility anchor and the mobile access gateway.

This mobility header option is used for requesting multipath support. It indicates that the mobile access gateway is requesting the local mobility anchor to register the current care-of address associated with the request as one of the many care-addresses through which the mobile access gateway can be reached. It is also for carrying the information related to the access network associated with the care-of address.

The MAG Multipath-Binding option has an alignment requirement of 8n+2. Its format is as shown in Figure 3:

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Type     |   Length      |    If-ATT     |    If-Label   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Binding-Id   |B|O|             RESERVED                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 3: MAG Multipath Binding Option

Type

<IANA-1> To be assigned by IANA.

Length

8-bit unsigned integer indicating the length of the option in octets, excluding the type and length fields.

This 8-bit field identifies the Access-Technology type of the interface through which the mobile node is connected. The permitted values for this are from the Access Technology Type registry defined in [RFC5213].

This 8-bit field represents the interface label represented as an unsigned integer. The mobile node identifies the label for each of the interfaces through which it registers a CoA with the home agent. When using static traffic flow policies on the mobile node and the home agent, the label can be used for generating forwarding policies. For example, the operator may have policy which binds traffic for Application "X" needs to interface with Label "Y". When a registration through an interface matching Label "Y" gets activated, the home agent and the mobile node can dynamically generate a forwarding policy for forwarding traffic for Application "X" through mobile IP tunnel matching Label "Y". Both the home agent and the mobile node can route the Application-X traffic through that interface. The permitted values for If-Label are 1 through 255.

This 8-bit field is used for carrying the binding identifier. It uniquely identifies a specific binding of the mobile node, to which this request can be associated. Each binding identifier is represented as an unsigned integer. The permitted values are 1 through 254. The BID value of 0 and 255 are reserved. The mobile access gateway assigns a unique value for each of its interfaces and includes them in the message.

This flag, if set to a value of (1), is to notify the local mobility anchor to consider this request as a request to update the binding lifetime of all the mobile node’s bindings, upon accepting this specific request. This flag MUST NOT be set to a value of (1), if the value of the Registration Overwrite Flag (O) flag is set to a value of (1).

This flag, if set to a value of (1), notifies the local mobility anchor that upon accepting this request, it should replace all of the mobile node’s existing bindings with this binding. This flag MUST NOT be set to a value of (1), if the value of the Bulk Re-registration Flag (B) is set to a value of (1). This flag MUST be set to a value of (0), in de-registration requests.

Reserved

This field is unused in this specification. The value MUST be set to zero (0) by the sender and MUST be ignored by the receiver.

4.2. MAG Identifier Option

The MAG Identifier option is a new mobility header option defined for use with Proxy Binding Update and Proxy Binding Acknowledgement messages exchanged between the local mobility anchor and the mobile access gateway. This mobility header option is used for conveying the MAG's identity.

This option does not have any alignment requirements.

```
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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Type     |   Length      |  Subtype      |  Reserved     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            Identifier ...                     ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 4: MAG Identifier Option

Type

<IANA-2> To be assigned by IANA.

Length

8-bit unsigned integer indicating the length of the option in octets, excluding the type and length fields.

Subtype

One byte unsigned integer used for identifying the type of the Identifier field. Accepted values for this field are the registered type values from the Mobile Node Identifier Option Subtypes registry.

Reserved

This field is unused in this specification. The value MUST be set to zero (0) by the sender and MUST be ignored by the receiver.

Identifier

A variable length identifier of type indicated in the Subtype field.

4.3. New Status Code for Proxy Binding Acknowledgement

This document defines the following new Status Code value for use in Proxy Binding Acknowledgement message.

CANNOT_SUPPORT_MULTIPATH_BINDING (Cannot Support Multipath Binding): <IANA-4>

5. IANA Considerations

This document requires the following IANA actions.

- o Action-1: This specification defines a new mobility option, the MAG Multipath-Binding option. The format of this option is described in Section 4.1. The type value <IANA-1> for this mobility option needs to be allocated from the Mobility Options registry at <http://www.iana.org/assignments/mobility-parameters>. RFC Editor: Please replace <IANA-1> in Section 4.1 with the assigned value and update this section accordingly.

- o Action-2: This specification defines a new mobility option, the MAG Identifier option. The format of this option is described in Section 4.2. The type value <IANA-2> for this mobility option needs to be allocated from the Mobility Options registry at <http://www.iana.org/assignments/mobility-parameters>. RFC Editor: Please replace <IANA-2> in Section 4.2 with the assigned value and update this section accordingly.

- o Action-4: This document defines a new status value, CANNOT_SUPPORT_MULTIPATH_BINDING (<IANA-4>) for use in Proxy Binding Acknowledgement message, as described in Section 4.3. This value is to be assigned from the "Status Codes" registry at <http://www.iana.org/assignments/mobility-parameters>. The allocated value has to be greater than 127. RFC Editor: Please replace <IANA-4> in Section 4.3 with the assigned value and update this section accordingly.

6. Security Considerations

This specification allows a mobile access gateway to establish multiple Proxy Mobile IPv6 tunnels with a local mobility anchor, by registering a care-of address for each of its connected access networks. This essentially allows the mobile node’s IP traffic to be routed through any of the tunnel paths and either based on a static or a dynamically negotiated flow policy. This new capability has no impact on the protocol security. Furthermore, this specification defines two new mobility header options, MAG Multipath-Binding option and the MAG Identifier option. These options are carried like any other mobility header option as specified in [RFC5213]. Therefore, it inherits security guidelines from [RFC5213]. Thus, this specification does not weaken the security of Proxy Mobile IPv6 Protocol, and does not introduce any new security vulnerabilities.

7. Acknowledgements

The authors of this draft would like to acknowledge the discussions and feedback on this topic from the members of the DMM working group.

8. References

8.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC3963] Devarapalli, V., Wakikawa, R., Petrescu, A., and P. Thubert, "Network Mobility (NEMO) Basic Support Protocol", RFC 3963, DOI 10.17487/RFC3963, January 2005, <http://www.rfc-editor.org/info/rfc3963>.

[RFC5094] Devarapalli, V., Patel, A., and K. Leung, "Mobile IPv6 Vendor Specific Option", RFC 5094, DOI 10.17487/RFC5094, December 2007, <http://www.rfc-editor.org/info/rfc5094>.

[RFC5213] Gundavelli, S., Ed., Leung, K., Devarapalli, V., Chowdhury, K., and B. Patil, "Proxy Mobile IPv6", RFC 5213, DOI 10.17487/RFC5213, August 2008, <http://www.rfc-editor.org/info/rfc5213>.

[RFC5648] Wakikawa, R., Ed., Devarapalli, V., Tsirtsis, G., Ernst, T., and K. Nagami, "Multiple Care-of Addresses Registration", RFC 5648, DOI 10.17487/RFC5648, October 2009, <http://www.rfc-editor.org/info/rfc5648>.

[RFC5844] Wakikawa, R. and S. Gundavelli, "IPv4 Support for Proxy Mobile IPv6", RFC 5844, DOI 10.17487/RFC5844, May 2010, <http://www.rfc-editor.org/info/rfc5844>.

[RFC5845] Muhanna, A., Khalil, M., Gundavelli, S., and K. Leung, "Generic Routing Encapsulation (GRE) Key Option for Proxy Mobile IPv6", RFC 5845, DOI 10.17487/RFC5845, June 2010, <http://www.rfc-editor.org/info/rfc5845>.

[RFC6088] Tsirtsis, G., Giarreta, G., Soliman, H., and N. Montavont, "Traffic Selectors for Flow Bindings", RFC 6088, DOI 10.17487/RFC6088, January 2011, <http://www.rfc-editor.org/info/rfc6088>.

[RFC6089] Tsirtsis, G., Soliman, H., Montavont, N., Giaretta, G., and K. Kuladinithi, "Flow Bindings in Mobile IPv6 and Network Mobility (NEMO) Basic Support", RFC 6089, DOI 10.17487/RFC6089, January 2011, <http://www.rfc-editor.org/info/rfc6089>.

[RFC6275] Perkins, C., Ed., Johnson, D., and J. Arkko, "Mobility Support in IPv6", RFC 6275, DOI 10.17487/RFC6275, July 2011, <http://www.rfc-editor.org/info/rfc6275>.

[RFC7148] Zhou, X., Korhonen, J., Williams, C., Gundavelli, S., and CJ. Bernardos, "Prefix Delegation Support for Proxy Mobile IPv6", RFC 7148, DOI 10.17487/RFC7148, March 2014, <http://www.rfc-editor.org/info/rfc7148>.

8.2. Informative References

[RFC2473] Conta, A. and S. Deering, "Generic Packet Tunneling in IPv6 Specification", RFC 2473, DOI 10.17487/RFC2473, December 1998, <http://www.rfc-editor.org/info/rfc2473>.

[RFC4213] Nordmark, E. and R. Gilligan, "Basic Transition Mechanisms for IPv6 Hosts and Routers", RFC 4213, DOI 10.17487/RFC4213, October 2005, <http://www.rfc-editor.org/info/rfc4213>.

[RFC4908] Nagami, K., Uda, S., Ogashiwa, N., Esaki, H., Wakikawa, R., and H. Ohnishi, "Multi-homing for small scale fixed network Using Mobile IP and NEMO", RFC 4908, DOI 10.17487/RFC4908, June 2007, <http://www.rfc-editor.org/info/rfc4908>.

Authors’ Addresses

Pierrick Seite
Orange
4, rue du Clos Courtel, BP 91226
Cesson-Sevigne 35512
France

Email: pierrick.seite@orange.com

Alper Yegin
Samsung
Istanbul
Turkey

Email: alper.yegin@partner.samsung.com

Sri Gundavelli
Cisco
170 West Tasman Drive
San Jose, CA 95134
USA

Email: sgundave@cisco.com

DMM Working Group S. Jeon
Internet-Draft Sungkyunkwan University
Intended status: Standards Track S. Figueiredo
Expires: March 15, 2018 Altran Research
Y. Kim
Soongsil University
J. Kaippallimalil
Huawei
September 11, 2017

Use Cases and API Extension for Source IP Address Selection
draft-sijeon-dmm-use-cases-api-source-07.txt

Abstract

This draft specifies and analyzes the expected cases regarding the selection of a proper source IP address and address type by an application in a distributed mobility management (DMM) network. It also proposes a new Socket API to address further selection issues with three source IP address types defined in the on-demand mobility API draft.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

Applications to select source IP address type in a mobile node (MN) need to consider IP session continuity and/or IP address reachability. [I-D.ietf-dmm-ondemand-mobility], defines three types of source IP addresses based on mobility management capabilities: fixed IP address, session-lasting IP address, and non-persistent IP address. Based on the address type requested by the application, the MN configures a proper source IP address. However, the source IP address type itself in a socket request may not be enough to convey all the requirements of an application. For example, more than one IP address of the same type requested may be available. It may be that as a result of mobility the MN can potentially obtain new IP prefixes from different serving networks belonging to different subnets. This draft categorizes and analyzes use cases that an MN is likely to encounter. In addition, this draft proposes an extension that allows the application to express its preferences when more than one source address of a type is present.

2. Use Cases and Analysis

This section outlines use cases where an application on the MN tries to obtain a source IP address.

2.1. Application has no specific IP address type requirement or address preference

Applications such as text-based web browsing or information service, e.g. weather and stock information, as well as legacy applications belong to this category. Many applications use short-lived Internet connections with no requirements for session continuity or IP address reachability. Assigning a non-persistent IP address can be thus considered as default for MNs. However, it is subject to address assignment policy of a network operator. The suggested flag, IPV6_REQUIRE_NON-PERSISTENT_IP, defined in [I-D.ietf-dmm-ondemand-mobility] can used for expressing its preference to the IP stack.

2.2. Application has specific IP address type requirement and address preference

This category is for an application requiring IP session continuity with different granularity of IP address reachability. This case may be further divided in three sub-cases with regard to IP address type availability and/or address selection.

2.2.1. Case 1: there is no configured IP address based on a requested type in the IP stack, but there is a further selection preference by the application

Once an IP address is requested by an application regardless of any source IP address type defined in [I-D.ietf-dmm-ondemand-mobility], the network stack will configure an IP address after obtaining an IP prefix based on the requested source IP address type from the current serving gateway.

2.2.2. Case 2: there are one or more IP addresses configured with a requested type in the IP stack, and no selection preference by the application

This is the same as Case 1 described above, except the existence of more than one configured IP addresses belonging to the requested IP address type in the IP stack, e.g. due to different address assignment policy by an operator.

When a non-persistent IP address is requested, if an application requests a non-persistent IP address to the IP stack, the IP address is obtained from the serving IP gateway as the previous one is not maintained across gateway changes.

When a session-lasting IP address is requested, an expected sequence can be described as follows;

1. The MN has one or more session-lasting IP addresses configured in the IP stack.

2. If an application requests a session-lasting IP address to the IP stack, it will try to use an existing session-lasting IP address as it is already configured in the IP stack. If there are multiple available session-lasting IP addresses, the default address selection rules will be applied [RFC6724], e.g. with scope preference, longest prefix matching, and/or so on. The best-matched IP address among them will be selected and assigned to the application.

3. Subsequently, the MN moves to another serving network, and the previous (mobile) sessions are still in use. A new application requests a session-lasting IP address with flag, IPV6_REQUIRE_SESSION_LASTING_IP to the IP stack. The selection of the session-lasting IP address follows the same procedure as described in Step 2.

When a fixed IP address is requested, it will follow the same procedure with session-lasting IP address request as described.

2.2.3. Case 3: there are one or more IP addresses with a requested type configured in the IP stack, but there is a further selection preference by the application

Assume that there are one or multiple applications with session-lasting IP address running. A newly initiated application might get one of the session-lasting IP addresses being used, not initiating a protocol procedure, i.e. DHCP or SLAAC for a new session-lasting IP address to the network. On the contrary, the IP stack might try to get a new session-lasting IP address from the current serving gateway by default. Acquiring a new session-lasting IP address may take some time (due to the exchange with the network) while using an existing one is instantaneous. On the other hand, using the existing one might yield less optimal routing. For example, the use of the IP address with an existing one configured might provide a suboptimal routing path as a result of a handover. This situation might not be preferred by newly initiated applications because the application incurs the costs of IP mobility even though the MN has not moved from the current serving network. Eventually, the new session is served by a remote IP mobility anchor with mobility management functions, though the MN has not moved yet.

If the application is allowed to further define its preference for an optimally routed, this situation can be avoided. See Section 3 for the proposed flag.

2.3. Gaps in the consistency with the default address selection

The need of an indication mechanism can be sought in the consistency with the former IETF standards. For example, in [RFC6724] where default behavior for IPv6 is specified, without a proper indication mechanism, following conflicts are expected to happen. In Rule 6 in [RFC6724], it is said that the matching label between source address of an IPv6 host and destination address is preferred among the combinations between other source addresses and destination address, where the label is a numeric value representing policies that prefer a particular source address prefix for use with a destination address prefix in [RFC6724]. In Rule 8 in [RFC6724], it is said that the longest matching prefix between source address of an IPv6 host and destination address is preferred among the combinations between other source addresses and destination address. Following Rules 6 and 8 may result in the selection of a source IP address with which packets that are sub-optimally routed.

3. Indications for expressing address preference requirement

When an application prefers a new IP address of the requested IP address type, additional indication flags should be delivered through the socket API interface.

To obtain an address that supports dynamic mobility using session-lasting IP address, a new address preference flag needs to be defined. The flag should be simple and useful while aligned with the three types of IP addresses. The objective of the hereby presented address preference flag is letting the IP stack check whether it has an available IP address assigned from the current serving network when the flag is received by an initiated application. If not, it will trigger the IP stack to get a new IP address from the current serving network. We call it "ON_NET" property.

If the application requests an IP address with ON_NET flag set in the socket request, the IP address returned by the stack should conform to the address preference requirement. This should be the case even though other session-lasting IP addresses, not belonging to the current serving network are available. If there are multiple session-lasting IP addresses matched with ON_NET property, the default source address selection rules will be applied.

```
IPV6_XX_SRC_ON_NET

/* Require (or Prefer) an IP address based on a requested IP address
type as source, assigned from the current serving network, whatever
it has been assigned or should be assigned */
```

This flag aims to express the preference to check an IP address, being used by an application, previously assigned from the current serving network and to use it or to get an IP address from the current serving network, as well as enabling differentiated per-flow anchoring where an obtained session-lasting IP address might be used for all initiated session-lasting IP applications. The use of the flag can be combined together with the three types of IP address defined in [I-D.ietf-dmm-ondemand-mobility].

In [I-D.mccann-dmm-prefixcost], it proposes that the Router Advertisement signaling messages communicate the cost of maintaining a given prefix at the MN’s current point of attachment. The objective is to make a dynamic and optimal decision of address assignment and release, i.e. when to release old addresses and assign new ones. The proposed ON_NET property may present a way to deliver a prefix decision for an application, specifically from a routing distance point of view, to the IP stack.

4. IANA Considerations

This document makes no request of IANA.

5. Security Considerations

T.B.D.

6. Acknowledgements

We would like to thank Danny Moses, Marco Liebsch, Brian Haberman, Sri Gundavelli, Alexandru Petrescu for their valueable comments and suggestions on this work.

7. References

7.1. Normative References

[RFC6724] Thaler, D., Ed., Draves, R., Matsumoto, A., and T. Chown, "Default Address Selection for Internet Protocol Version 6 (IPv6)", RFC 6724, DOI 10.17487/RFC6724, September 2012, <https://www.rfc-editor.org/info/rfc6724>.

7.2. Informative References

[I-D.ietf-dmm-ondemand-mobility]
Yegin, A., Moses, D., Kweon, K., Lee, J., Park, J., and S. Jeon, "On Demand Mobility Management", draft-ietf-dmm-ondemand-mobility-12 (work in progress), July 2017.

[I-D.mccann-dmm-prefixcost]
McCann, P. and J. Kaippallimalil, "Communicating Prefix Cost to Mobile Nodes", draft-mccann-dmm-prefixcost-03 (work in progress), April 2016.

Authors’ Addresses

Seil Jeon
Sungkyunkwan University
2066 Seobu-ro, Jangan-gu
Suwon, Gyeonggi-do
Korea

Email: seiljeon@skku.edu

Sergio Figueiredo
Altran Research
2, Rue Paul Dautier
Velizy-Villacoublay 78140
France

Email: sergio.figueiredo@altran.com

Younghan Kim
Soongsil University
369, Sangdo-ro, Dongjak-gu
Seoul 156-743
Korea

Email: younghak@ssu.ac.kr

John Kaippallimalil
Huawei
5340 Legacy Dr., Suite 175
Plano, TX 75024
USA

Email: john.kaippallimalil@huawei.com

DMM Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 13, 2016

Z. Yan
CNNIC
J. Lee
Sangmyung University
X. Lee
CNNIC

October 11, 2015

Home Network Prefix Renumbering in PMIPv6
draft-yan-dmm-hnprenum-03.txt

Abstract

In the basic Proxy Mobile IPv6 (PMIPv6) specification, a Mobile Node (MN) is assigned with a 64-bit Home Network Prefix (HNP) during its initial attachment for the Home Address (HoA) configuration. During the movement of the MN, this prefix remains unchanged and in this way it is unnecessary for the MN to reconfigure its HoA and reconnect the ongoing communications. However, the current protocol (RFC5213) does not specify related operations to support the MN to timely receive and use a new HNP when the allocated HNP changes. In this draft, a solution to support the HNP renumbering is proposed, as an update of RFC5213.

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.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

Network managers currently prefer Provider Independent (PI) addressing for IPv6 to attempt to minimize the need for future possible renumbering. However, widespread use of PI addresses may cause Border Gateway Protocol (BGP) scaling problems. It is thus desirable to develop tools and practices that may make IPv6 renumbering a simpler process to reduce demand for IPv6 PI space [RFC6879]. In this draft, we aim to solve the HNP renumbering problem when the HNP in PMIPv6 [RFC5213] is not the type of PI.

2. Usage scenarios

There are a number of reasons why the HNP renumbering support in PMIPv6 is useful and a few are identified below:

o Scenario 1: the PMIPv6 service provider is assigned with the HNP set from the (uplink) Internet Service Provider (ISP), and then the HNP renumbering may happen if the PMIPv6 service provider switches to a different ISP.

- Scenario 2: multiple Local Mobility Anchors (LMAs) may be deployed by the same PMIPv6 service provider, and then each LMA may serve for a specific HNP set. In this case, the HNP of an MN may change if the current serving LMA switches to another LMA but without inheriting the assigned HNP set [RFC6463].

- Scenario 3: the PMIPv6 HNP renumbering may be caused by the re-building of the network architecture as the companies split, merge, grow, relocate or reorganize. For example, the PMIPv6 service provider may reorganize its network topology.

In the scenario 1, we assume that only the HNP is renumbered while the serving LMA remains unchanged and this is the basic scenario of this document. In the scenario 2 and scenario 3, more complex results may be caused, for example, the HNP renumbering may happen due to the switchover of serving LMA.

In the Mobile IPv6 (MIPv6) protocol, when the home network prefix changes (may be also caused by the above reasons), the Home Agent (HA) will actively notify the new prefix to the MN and then the renumbering of the HoA can be well supported [RFC6275]. While in the basic PMIPv6, the PMIPv6 binding is triggered by the Mobile Access Gateway (MAG), which detected the attachment of the MN. When the HNP renumbering happens, a scheme is also needed for the LMA to immediately initiate the PMIPv6 binding state refreshment. Although this issue is also discussed in the [RFC5213] (Section 6.12), the related solution has not been specified.

3. PMIPv6 extensions

When the HNP renumbering happens in PMIPv6, the LMA has to notify the new HNP to the MAG and then the MAG has to announce the new HNP to the MN accordingly. Also, the LMA and the MAG must update the routing states for the prefixes. To support this procedure, RFC7077 can be adopted which specifies asynchronously update from the LMA to the MAG about the updated session parameters. This document considers the following two cases:

(1)HNP is renumbered in the same LMA

In this case, the LMA remains unchanged as in the scenario 1 and scenario 3. The operation steps are shown in Figure 1.

```
   +-----+                +-----+                +-----+
   | MN  |                | MAG |                | LMA |
   +-----+                +-----+                +-----+
     |                      |                      |
     |                      |           Allocate new HNP
     |                      |                      |
     |                      |<------------- UPN ---|
     |                      |                      |
     |               Update routing state          |
     |                      |                      |
     |<-----RA/DHCP --------|                      |
     |                      |                      |
   HoA Configuration        |                      |
     |                      |                      |
     |                      |--- UNA ------------->|
     |                      |                      |
     |                      |      Update routing state
     |                      |                      |
```

Figure 1: Signaling call flow of HNP renumbering

- When the PMIPv6 service provider renumbers the HNP set in the same LMA, the serving LMA will initiate the HNP renumbering operation. The LMA allocates a new HNP for the related MN.

- The LMA sends the Update Notification (UPN) message to the MAG to update the HNP information. If the Dynamic Host Configuration Protocol (DHCP) is used in PMIPv6 to allocate the HoA, the new HNP should be also notified to the DHCP infrastructure.

- After the MAG receives this UPN message, it recognizes that the related MN has a new HNP. Then the MAG should notify the MN about the new HNP with a Router Advertisement (RA) message or allocate a new address within the new HNP with a DHCP message.

- When the MN obtains the new HNP information, it deletes the old HoA and configures a new HoA with the newly allocated HNP.

- The MAG sends back the Update Notification Acknowledgement (UNA) to the LMA for the notification of successful update of the HNP, related binding state, and routing state. Then the LMA updates the routing information corresponding to the MN to replace the old HNP with the new one.

(2) HNP renumbering caused by LMA switchover

Because the HNP is assigned by the LMA, the HNP renumbering may be caused by the LMA switchover, as in the scenario 2 and scenario 3.

The information of LMA is the basic configuration information of MAG. When the LMA changes, the related profile should be updated by the service provider. In this way, the MAG will initiate the re-registration to the new LMA as specified in RFC5213. When the HNP renumbering is caused in this case, the new HNP information will be sent by the LMA during the new binding procedure. Accordingly, the MAG will withdraw the old HNP information of the MN and advise the new HNP to the MN as Step (3) in Section 3.1.

4. Session connectivity

HNP renumbering may cause the disconnection of the ongoing communications of the MN. Basically, there are two modes to manage the session connectivity during the HNP renumbering.

(1)Soft-mode

The LMA will temporarily maintain the state of the old HNP during the HNP renumbering (after the UNA reception) in order to redirect the packets to the MN before the MN reconnects the ongoing session and notifies its new HoA to the Correspondent Node (CN). This mode is aiming to reduce the packet loss during the HNP renumbering but the binding state and routing entry corresponding to the old HNP should be marked for example as transient binding [RFC6058]. This temporary binding should only be used for the downwards packet transmission and the LMA should not broadcast the routing information about the old HNP if it is no longer anchored at this LMA.

(2)Hard-mode

If the HNP renumbering happens with the switchover of the LMA, the hard-mode is recommended to keep the protocol simple and efficient. In this mode, the LMA will delete the state of the old HNP after it receives the UNA message from MAG and the LMA will silently discard the packets destinated to the old HNP.

5. Message format

(1)UPN message

In the UPN message sent from the LMA to the MAG, the notification reason is set to 2 (UPDATE-SESSION-PARAMETERS). Besides, the HNP option containing the new HNP and the Mobile Node Identifier option carrying Identifier of MN are contained as Mobility Options of UPN.

(2)RA Message

When the RA message is used by the MAG to advise the new HNP, two Prefix Information options are contained in the RA message [RFC2461]. In the first Prefix Information option, the old HNP is carried but both the related Valid Lifetime and Preferred Lifetime are set to 0. In the second Prefix Information option, the new HNP is carried with the Valid Lifetime and Preferred Lifetime set to larger than 0.

(3)DHCP Message

When the DHCP is used in PMIPv6 to configure the HoA for the MN, a new IPv6 HoA is generated based on the new HNP. Trigged by the UPN message, the MAG will request the new HoA from the DHCP server first and then the MAG updates the allocated HoA to the MN through the DHCP server-initiated configuration exchange [RFC3315].

6. Other issues

In order to maintain the reachability of the MN, the Domain Name System (DNS) resource record corresponding to this MN may need to be updated when the HNP of MN changes [RFC3007]. However, this is out the scope of this draft.

7. Security considerations

This extension causes no further security problem. The security considerations in [RFC5213] and [RFC7077] are enough for the basic operation of this draft.

Other security issues will be analyzed further.

8. Normative References

[RFC2461] Narten, T., Nordmark, E., and W. Simpson, "Neighbor Discovery for IP Version 6 (IPv6)", RFC 2461, DOI 10.17487/RFC2461, December 1998, <http://www.rfc-editor.org/info/rfc2461>.

[RFC3007] Wellington, B., "Secure Domain Name System (DNS) Dynamic Update", RFC 3007, DOI 10.17487/RFC3007, November 2000, <http://www.rfc-editor.org/info/rfc3007>.

[RFC3315] Droms, R., Ed., Bound, J., Volz, B., Lemon, T., Perkins, C., and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3315, DOI 10.17487/RFC3315, July 2003, <http://www.rfc-editor.org/info/rfc3315>.

[RFC5213] Gundavelli, S., Ed., Leung, K., Devarapalli, V., Chowdhury, K., and B. Patil, "Proxy Mobile IPv6", RFC 5213, DOI 10.17487/RFC5213, August 2008, <http://www.rfc-editor.org/info/rfc5213>.

[RFC6058] Liebsch, M., Ed., Muhanna, A., and O. Blume, "Transient Binding for Proxy Mobile IPv6", RFC 6058, DOI 10.17487/RFC6058, March 2011, <http://www.rfc-editor.org/info/rfc6058>.

[RFC6275] Perkins, C., Ed., Johnson, D., and J. Arkko, "Mobility Support in IPv6", RFC 6275, DOI 10.17487/RFC6275, July 2011, <http://www.rfc-editor.org/info/rfc6275>.

[RFC6463] Korhonen, J., Ed., Gundavelli, S., Yokota, H., and X. Cui, "Runtime Local Mobility Anchor (LMA) Assignment Support for Proxy Mobile IPv6", RFC 6463, DOI 10.17487/RFC6463, February 2012, <http://www.rfc-editor.org/info/rfc6463>.

[RFC6879] Jiang, S., Liu, B., and B. Carpenter, "IPv6 Enterprise Network Renumbering Scenarios, Considerations, and Methods", RFC 6879, DOI 10.17487/RFC6879, February 2013, <http://www.rfc-editor.org/info/rfc6879>.

[RFC7077] Krishnan, S., Gundavelli, S., Liebsch, M., Yokota, H., and J. Korhonen, "Update Notifications for Proxy Mobile IPv6", RFC 7077, DOI 10.17487/RFC7077, November 2013, <http://www.rfc-editor.org/info/rfc7077>.

Authors’ Addresses

Zhiwei Yan
CNNIC
No.4 South 4th Street, Zhongguancun
Beijing 100190
China

EMail: yan@cnnic.cn

Jong-Hyouk Lee
Sangmyung University
31, Sangmyeongdae-gil, Dongnam-gu
Cheonan
Republic of Korea

EMail: jonghyouk@smu.ac.kr

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

EMail: xl@cnnic.cn