

DMM Working Group
Internet-Draft
Intended status: Informational
Expires: May 7, 2020

U. Chunduri, Ed.
R. Li
Futurewei
S. Bhaskaran
Altiostar
J. Kaippallimalil
Futurewei
J. Tantsura
Apstra, Inc.
L. Contreras
Telefonica
P. Muley
Nokia
November 4, 2019

Transport Network aware Mobility for 5G
draft-clt-dmm-tn-aware-mobility-05

Abstract

This document specifies a framework and a mapping function for 5G mobile user plane with transport network slicing, integrated with Mobile Radio Access and a Virtualized Core Network. The integrated approach is specified in a way to fit into the 5G core network architecture defined in [TS23.501].

It focuses on an optimized mobile user plane functionality with various transport services needed for some of the 5G traffic needing low and deterministic latency, real-time, mission-critical services. This document describes, how this objective is achieved agnostic to the transport underlay used (IPv6, MPLS, IPv4) in various deployments and with a new transport network underlay routing, called Preferred Path Routing (PPR).

Requirements Language

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

Status of This Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute

working documents as Internet-Drafts. The list of current Internet-Drafts is at <https://datatracker.ietf.org/drafts/current/>.

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

This Internet-Draft will expire on May 7, 2020.

Copyright Notice

Copyright (c) 2019 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 (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
1.1. Problem Statement	4
1.2. Solution Approach	4
1.3. Acronyms	5
2. Transport Network and Slice aware Mobility on N3/N9	6
2.1. XHaul Transport Network	7
2.2. Mobile Transport Network Context (MTNC) and Scalability	8
2.3. Transport Network Function (TNF)	9
2.4. TNF Interfaces	9
2.5. Transport Provisioning	10
2.6. MTNC-ID in the Data Packet	11
2.7. Functionality for E2E Management	12
3. Using PPR as TN Underlay	13
3.1. PPR with Transport Awareness for 5GS on N3/N9 Interfaces	14
3.2. Path Steering Support to native IP user planes	16
3.3. Service Level Guarantee in Underlay	16
4. Other TE Technologies Applicability	16
5. Acknowledgements	17
6. IANA Considerations	17
7. Security Considerations	17
8. Contributing Authors	17

9. References	17
9.1. Normative References	17
9.2. Informative References	17
Appendix A. New Control Plane and User Planes	20
A.1. Slice aware Mobility: Discrete Approach	20
Appendix B. PPR with various 5G Mobility procedures	21
B.1. SSC Model1	22
B.2. SSC Mode2	23
B.3. SSC Mode3	23
Authors' Addresses	25

1. Introduction

The 3GPP architecture for 5GS is defined in [TS.23.501-3GPP], [TS.23.502-3GPP] and [TS.23.503-3GPP]. The architecture defines a comprehensive set of functions for access mobility, session handling and related functions for subscription management, authentication and policy among others. These network functions (NF) are defined using a service-based architecture (SBA) that allows NFs to expose their functions via an API and common service framework. The architecture also defines slicing aspects where the Network Slice Selection Function (NSSF) assists the Access Mobility Manager (AMF) and Session Management Function (SMF) to assist and select the right entities and resources corresponding to the slice requested by the user equipment (UE). The User Equipment (UE) indicates slice information in the Network Slice Selection Assistance Information (NSSAI) field during session establishment (Attach). The AMF selects the right SMF and the SMF in turn selects the User Plane Functions (UPF) so that the QoS and capabilities requested can be fulfilled. UPFs are the data forwarding entities in the 5GC architecture. The architecture allows the placement of Branching Point (BP) and Uplink Classifier (ULCL) UPFs closer to the access network (5G-AN). The 5G-AN can be a radio access network or any non-3GPP access network, for example, WLAN. The IP address is anchored by a PDU session anchor UPF (PSA UPF).

5GS allows more than one UPF on the path for a PDU (Protocol Data Unit) session that provides various functionality including session anchoring, uplink classification and branching point for a multihomed IPv6 PDU session. The interface between the BP/ULCL UPF and the PSA UPF is called N9 [TS.23.501-3GPP]. 3GPP has adopted GTP-U for the N9 and N3 interface between the various UPF instances and the (R)AN. 3GPP has specified control and user plane aspects in [TS.23.501-3GPP] to provide slice and QoS support. 3GPP has defined three broad slice types to cover enhanced mobile broadband (eMBB) communications, ultra-reliable low latency communications (URLLC) and massive internet of things (mIoT). ATIS [ATIS075] has defined an additional slice type for V2X services. There may be multiple instances of a slice type to satisfy some characteristics like isolation. The slice

details in 3GPP, ATIS or NGMN do not specify how slice characteristics for QoS, hard /soft isolation, protection and other aspects should be satisfied in IP transport networks. This is explored further in this document.

1.1. Problem Statement

[TS.23.501-3GPP] and [TS.23.502-3GPP] define network slicing as one of the core capability of 5GC with slice awareness from Radio and 5G Core (5GC) network. The 5G System (5GS) as defined, does not consider the resources and functionalities needed from transport network for the selection of UPF. This is seen as independent functionality and currently not part of 5GS.

However, the lack of underlying Transport Network (TN) awareness may lead to selection of sub-optimal UPF(s) and/or 5G-AN during 5GS various procedures (e.g., session establishment, mobility). Meeting the specific slice characteristics on the N3 and N9 interfaces depends on the IP transport underlay providing these resources and capabilities. This could also lead to the inability in meeting SLAs for real-time, mission-critical or latency sensitive services. 5GS procedures including but not limited to Service Request, PDU Session Establishment, or UE mobility need same service level characteristics from the TN for the Protocols Data Unit (PDU) session, similar to as provided in Radio and 5GC for the various Slice Service Types (SST) and 5QI's defined in [TS.23.501-3GPP].

The 5GS provides slices to its clients (UEs). The UE's PDU session spans the access network (radio) and N3 and N9 transport segments which have an IP transport underlay. The 5G operator needs to obtain slice capability from the IP transport provider. Several UE sessions that match a slice may be mapped to an IP transport segment. Thus there needs to be a mapping between the slice capability offered to the UE (NSSAI) and what is provided by the IP transport.

1.2. Solution Approach

This document specifies an approach to fulfil the needs of 5GS to transport user plane traffic from 5G-AN to UPF for all service continuity modes [TS.23.501-3GPP] in an optimized fashion. This is done by, keeping mobility procedures aware of underlying transport network along with slicing requirements.

Section 2 describes in detail on how TN aware mobility can be built irrespective of underlying TN technology used. Using Preferred Path Routing (PPR), applicable to any transport network underlay (IPv6, MPLS and IPv4) is detailed in Section 3. How other IETF TE technologies applicable for this draft is specified in Section 4. At

the end, Appendix B further describes the applicability and procedures of PPR with 5G SSC modes on N3 and N9 interfaces.

1.3. Acronyms

5QI	-	5G QoS Indicator
5G-AN	-	5G Access Network
AMF	-	Access and Mobility Management Function (5G)
BP	-	Branch Point (5G)
CSR	-	Cell Site Router
DN	-	Data Network (5G)
eMBB	-	enhanced Mobile Broadband (5G)
FRR	-	Fast ReRoute
gNB	-	5G NodeB
GBR	-	Guaranteed Bit Rate (5G)
GTP-U	-	GPRS Tunneling Protocol - Userplane (3GPP)
IGP	-	Interior Gateway Protocols (e.g. IS-IS, OSPFv2, OSPFv3)
LFA	-	Loop Free Alternatives (IP FRR)
mIOT	-	Massive IOT (5G)
MPLS	-	Multi Protocol Label Switching
QFI	-	QoS Flow ID (5G)
PPR	-	Preferred Path Routing
PDU	-	Protocol Data Unit (5G)
PW	-	Pseudo Wire
RQI	-	Reflective QoS Indicator (5G)
SBI	-	Service Based Interface (5G)
SID	-	Segment Identifier

- SMF - Session Management Function (5G)
- SSC - Session and Service Continuity (5G)
- SST - Slice and Service Types (5G)
- SR - Segment Routing
- TE - Traffic Engineering
- ULCL - Uplink Classifier (5G)
- UPF - User Plane Function (5G)
- URLLC - Ultra reliable and low latency communications (5G)

2. Transport Network and Slice aware Mobility on N3/N9

Currently specified Control Plane (CP) functions - the AMF, the SMF and the User plane (UP) components 5G-AN (e.g. gNB), UPF with N2, N3, N4, N6 and N9 interfaces are relevant to this document. Other Virtualized 5G control plane components NRF, AUSF, PCF, AUSF, UDM, NEF, and AF are not directly relevant for the discussion in this document and one can see the functionalities of these in [TS.23.501-3GPP].

From encapsulation perspective, N3 interface is similar to S1U in 4G/LTE [TS.23.401-3GPP] network and uses GTP-U [TS.29.281-3GPP] to transport any UE PDUs (IPv4, IPv6, IPv4v6, Ethernet or Unstructured). Unlike S1U, N3 has some additional aspects as there is no bearer concept and no per bearer GTP-U tunnels. Instead, QoS information is carried in the PDU Session Container GTP-U extension header.

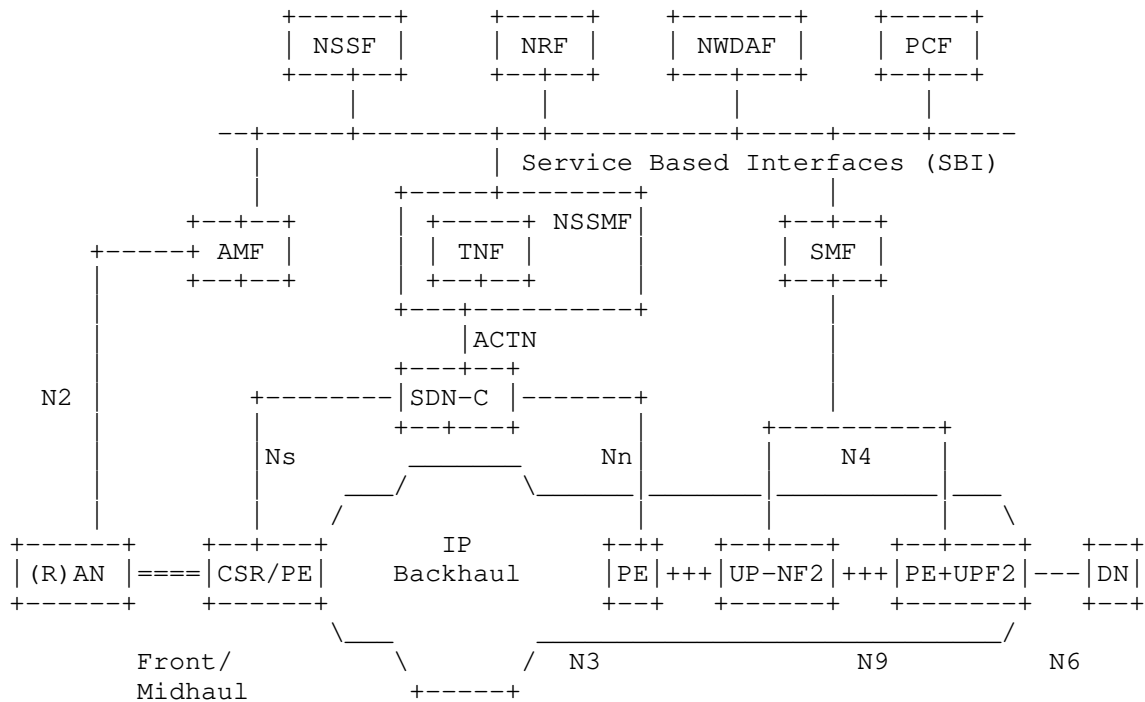


Figure 1: 5G Service Based Architecture with Transport Network

TN Aware Mobility with optimized transport network functionality is explained below. How PPR fits in this framework in detail along with other various TE technologies briefly are in Section 3 and Section 4 respectively.

2.1. XHaul Transport Network

Figure 1 depicts IP Xhaul network with SDN-C and CSR/PE (Provider Edge) routers provide IP transport service to 5GS user plane entities 5G-AN (e.g. gNB) and UPF. 5GS architecture with key control and user plane entities and its interaction with the IP transport plane is shown here. When a UE initiates session establishment, it indicates the desired slice type in the S-NSSAI (Specific Network Slice Selection Assistance Information) field. The AMF uses the S-NSSAI, other subscription information and configuration in the NSSF to select the appropriate SMF and the SMF in turn selects UPFs (User Plane Functions) that are able to provide the specified slice resources and capabilities. Some of the slice capabilities along the user plane path between the (R)AM and UPFs (N3, N9 segments) such as

a low latency path, jitter, protection and priority needs these to be provided by the IP transport network.

Interface between (R)AN/5G-AN and CSR includes fronthaul and midhaul part of the transport network. In some cases midhaul can also a layer-2 network. In deployments where virtualized 5G-AN is used, F1U interface with GTP encapsulation is considered between the Distributed Unit (DU) and Centralized Unit (CU).

Slice capability required in IP transport networks is estimated and provisioned by the functionality as specified in Section 2.3 (TNF) with support from various other functions such as the Network Data Analytics Function (NWDAF), Network Resource Function (NRF) and Policy Control Function (PCF). Figure 1 depicts the PE router, where transport paths are initiated/terminated can be deployed separately with UPF or both functionalities can be in the same node. The TNF provisions this in the SDN-C of the IP XHaul network using ACTN [RFC8453]. When a GTP encapsulated user packet from the (R)AN or UPF with the slice information traverses the F1U/N3/N9 segment, the PE router of the IP transport underlay can inspect the slice information and provide the provisioned capabilities. This is elaborated further in Section 2.5.

2.2. Mobile Transport Network Context (MTNC) and Scalability

The MTNC represents a slice, QoS configuration for a transport path between two 3GPP user plane functions. The Mobile-Transport Network Context Identifier (MTNC-ID) is generated by the TNF to be unique for each path and per traffic class (including QoS and slice aspects). Thus, there may be more than one MTNC-ID for the same QoS and path if there is a need to provide isolation (slice) of the traffic. It should be noted that MTNC are per class/path and not per user session (nor is it per data path entity). The MTNC-IDs are configured by the TNF to be unique within a provisioning domain.

Since the MTNC-IDs are not generated per user flow or session, there is no need for unique MTNC-IDs per flow/session. In addition, since the traffic estimation not performed at the time of session establishment, there is no provisioning delay experienced during session setup. The MTNC-ID space scales as a square of the number sites between which 3GPP user plane functions require paths. If there are T traffic classes across N sites, the number of MTNC-IDs in a fully meshed network is $(N*(N-1)/2) * T$. For example, if there are 3 traffic classes between 25 sites, there would be at most 900 MTNC-IDs required. Multiple slices for the same QoS class that need to be fully isolated, will add to the MTNC provisioning. An MTNC-ID space of 16 bits (65K+ identifiers) can be expected to be sufficient.

2.3. Transport Network Function (TNF)

Figure 1 shows a view of the functions and interfaces for provisioning the MTNC-IDs. The focus is on provisioning between the 3GPP management plane (NSSMF), transport network (SDN-C) and carrying the MTNC-IDs in PDU packets for the transport network to grant the provisioned resources.

In Figure 1, the TNF (logical functionality within the NSSMF) requests the SDN-C in the transport domain to program the TE path using ACTN [RFC 8453]. The SDN-C programs the Provider Edge (PE) routers and internal routers according to the underlay transport technology (e.g., PPR, MPLS, SRv6). The PE router inspects incoming PDU data packets for the MTNC-ID, classifies and provides the VN service provisioned across the transport network.

The detailed mechanisms by which the NSSMF provides the MTNC-IDs to the control plane and user plane functions are for 3GPP to specify. Two possible options are outlined below for completeness. The NSSMF may provide the MTNC-IDs to the 3GPP control plane by either providing it to the Session Management Function (SMF), and the SMF in turn provisions the user plane functions (UP-NF1, UP-NF2) during PDU session setup. Alternatively, the user plane functions may request the MTNC-IDs directly from the TNF/NSSMF. Figure 1 shows the case where user plane entities request the TNF/NSSMF to translate the Request and get the MTNC-ID (over interface Ns/Nn).

The TNF should be seen as a logical entity that can be part of NSSMF in the 3GPP management plane [TS.28.533-3GPP]. The NSSMF may use network configuration, policies, history, heuristics or some combination of these to derive traffic estimates that the TNF would use. How these estimates are derived are not in the scope of this document. The focus here is only in terms of how the TNF and SDN-C are programmed given that slice and QoS characteristics across a transport path can be represented by an MTNC-ID. The TNF requests the SDN-C in the transport network to provision paths in the transport domain based on the MTNC-ID. The TNF is capable of providing the MTNC-ID provisioned to control and user plane functions in the 3GPP domain. Detailed mechanisms for programming the MTNC-ID should be part of the 3GPP specifications.

2.4. TNF Interfaces

A south bound interface Ns is shown which interacts with the 5G Access Network (e.g. CSR). 'Ns' can use one or more mechanism available today (PCEP [RFC5440], NETCONF [RFC6241], RESTCONF [RFC8040] or gNMI) to provision the L2/L3 VPNs along with TE underlay paths from CSR to PE@UPF. Ns and Nn interfaces can be part of the

integrated 3GPP architecture, but the specification/ownership of these interfaces SHOULD be left out of scope of 3GPP.

A north bound interface 'Nn' is shown from one or more of the transport network nodes (or PE @ ULCL/BP UPF, Anchor Point UPF) to TNF as shown in Figure 1. It would enable learning the TE characteristics of all links and nodes of the network continuously (through BGP-LS [RFC7752] or through a passive IGP adjacency and PCEP [RFC5440]).

These VPNs and/or underlay TE paths MUST be similar on all 5G-AN/CSRs and UPFs concerned to allow mobility of UEs while associated with one of the Slice/Service Types (SSTs) as defined in [TS.23.501-3GPP].

2.5. Transport Provisioning

Functionality of transport provisioning for an engineered IP transport that supports 3GPP slicing and QoS requirements in [TS.23.501-3GPP] is described in this section.

During a PDU session setup, the AMF using input from the NSSF selects a network slice and SMF. The SMF with user policy from Policy Control Function (PCF) sets 5QI (QoS parameters) and the UPF on the path of the PDU session. While QoS and slice selection for the PDU session can be applied across the 3GPP control and user plane functions as outlined in Section 2, the IP transport underlay across N3 and N9 segments do not have enough information to apply the resource constraints represented by the slicing and QoS classification. Current guidelines for interconnection with transport networks [IR.34-GSMA] provide an application mapping into DSCP. However, these recommendations do not take into consideration other aspects in slicing like isolation, protection and replication.

IP transport networks have their own slice and QoS configuration based on domain policies and the underlying network capability. Transport networks can enter into an agreement for virtual network services (VNS) with client domains using the ACTN [RFC8453] framework. An IP transport network may provide such slice instances to mobile network operators, CDN providers or enterprises for example. The 3GPP mobile network, on the other hand, defines a slice instance for UEs as are the the mobile operator's 'clients'. The Network Slice Selection Management Function (NSSMF) [TS 28.533] that interacts with a TN controller like an SDN-C (that is out of scope of 3GPP).

The ACTN VN service can be used across the IP transport networks to provision and map the slice instance and QoS of the 3GPP domain to the IP transport domain. An abstraction that represents QoS and

slice instance in the mobile domain and mapped to ACTN VN service in the transport domain is represented here as MTNC-IDs. Details of how the MTNC-IDs are derived are upto functions that can estimate the level of traffic demand.

The 3GPP network/5GS provides slices instances to its clients (UE) that include resources for radio and mobile core segments. The UE's PDU session spans the access network (radio) and N3 and N9 transport segments which have an IP transport underlay. The 5G operator needs to obtain slice capability from the IP transport provider since these resources are not seen by the 5GS. Several UE sessions that match a slice may be mapped to an IP transport segment. Thus, there needs to be a mapping between the slice capability offered to the UE (NSSAI) and what is provided by the IP transport.

When the 3GPP user plane function (5G-AN, UPF) does not terminate the transport underlay protocol (e.g., MPLS), it needs to be carried in the IP protocol header from end-to-end of the mobile transport connection (N3, N9). [I-D.ietf-dmm-5g-uplane-analysis] discusses these scenarios in detail.

2.6. MTNC-ID in the Data Packet

When the 3GPP user plane function (5G-AN, UPF) and transport provider edge is on different nodes, the PE router needs to have the means by which to classify the PDU packet. The mapping information is provisioned between the 5G provider and IP transport network and corresponding information should be carried in each IP packet on the N3, N9 interface. To allow the IP transport edge nodes to inspect the transport context information efficiently, it should be carried in an IP header field that is easy to inspect. It may be noted that the N3 and N9 interfaces in 5GS are IP interfaces. Thus, Layer 2 alternatives such as VLAN will fail if there are multiple L2 networks on the N3 or N9 path. Another alternative is to use L3 VPNs. However, in the 5G architecture, there may be a large number of smaller UPFs that are dynamically inserted on path. Adding this VPN information for dynamic UPF insertion is configuration intensive and error prone. GTP (N3, N9 encapsulation header) field extensions offer a possibility, however these extensions are hard for a transport edge router to parse efficiently on a per packet basis. Other IP header fields like DSCP are not suitable as it only conveys the QoS aspects (but not other aspects like isolation, protection, etc.)

IPv6 extension headers like Fast, or SRv6 may be options to carry the MTNC-ID when such mechanism is a viable (if complete transport network is IPv6 based). To minimise the protocol changes are required and make this underlay transport independent (IPv4/IPv6/MPLS/

L2), an option is to provision a mapping of MTNC-ID to a UDP port range of the GTP encapsulated user packet. A simple mapping table between the MTNC-ID and the source UDP port number can be configured to ensure that ECMP /load balancing is not affected adversely by encoding the UDP source port with an MTNC-ID mapping. This mapping is configured in 3GPP user plane functions (5G-AN, UPF) and Provider Edge (PE) Routers that process MTNC-IDs. PE routers can thus provision a policy based on the source UDP port number (which reflects the mapped MTNC-ID) to underlying transport path and then deliver the QoS/slice resource provisioned in the transport network. The source UDP port that is encoded is the outer IP (corresponding to GTP header) while the inner IP packet (UE payload) is unaltered.

2.7. Functionality for E2E Management

With the TNF functionality in 5GS Service Based Interface, the following additional functionalities are required for end-2-end slice management including the transport network:

- o The Specific Network Slice Selection Assistance Information (S-NSSAI) of PDU session SHOULD be mapped to the assigned transport VPN and the TE path information for that slice.
- o For transport slice assignment for various SSTs (eMBB, URLLC, MIIOT) corresponding underlay paths need to be created and monitored from each transport end point (CSR and PE@UPF).
- o During PDU session creation, apart from radio and 5GC resources, transport network resources needed to be verified matching the characteristics of the PDU session traffic type.
- o The TNF MUST provide an API that takes as input the source and destination 3GPP user plane element address, required bandwidth, latency and jitter characteristics between those user plane elements and returns as output a particular TE path's identifier, that satisfies the requested requirements.
- o Mapping of PDU session parameters to underlay SST paths need to be done. One way to do this is to let the SMF install a Forwarding Action Rule (FAR) in the UPF via N4 with the FAR pointing to a "Network Instance" in the UPF. A "Network Instance" is a logical identifier for an underlying network. The "Network Instance" pointed by the FAR can be mapped to a transport path (through L2/L3 VPN). FARs are associated with Packet Detection Rule (PDR). PDRs are used to classify packets in the uplink (UL) and the downlink (DL) direction. For UL procedures specified in Section 2.5, Section 2.6 can be used for classifying a packet belonging to a particular slice characteristics. For DL, at a PSA

UPF, the UE IP address is used to identify the PDU session, and hence the slice a packet belongs to and the IP 5 tuple can be used for identifying the flow and QoS characteristics to be applied on the packet at UPF. If a PE is not co-located at the UPF then mapping to the underlying TE paths at PE happens based on the encapsulated GTP-US packet as specified in Section 2.6.

- o If any other form of encapsulation (other than GTP-U) either on N3 or N9 corresponding QFI information MUST be there in the encapsulation header.
- o In some SSC modes Appendix B, if segmented path (CSR to PE@staging/ULCL/BP-UPF to PE@anchor-point-UPF) is needed, then corresponding path characteristics MUST be used. This includes a path from CSR to PE@UL-CL/BP UPF [TS.23.501-3GPP] and UL-CL/BP UPF to eventual UPF access to DN.
- o Continuous monitoring of the underlying transport path characteristics should be enabled at the endpoints (technologies for monitoring depends traffic engineering technique used as described in Section 3 and Section 4). If path characteristics are degraded, reassignment of the paths at the endpoints should be performed. For all the affected PDU sessions, degraded transport paths need to be updated dynamically with similar alternate paths.
- o During UE mobility event similar to 4G/LTE i.e., gNB mobility (Xn based or N2 based), for target gNB selection, apart from radio resources, transport resources MUST be factored. This enables handling of all PDU sessions from the UE to target gNB and this require co-ordination of gNB, AMF, SMF with the TNF module.

Integrating the TNF as part of the 5GS Service Based Interfaces, provides the flexibility to control the allocation of required characteristics from the TN during a 5GS signaling procedure (e.g. PDU Session Establishment). If TNF is seen as part of management plane, this real time flexibility is lost. Changes to detailed signaling to integrate the above for various 5GS procedures as defined in [TS.23.502-3GPP] is beyond the scope of this document.

3. Using PPR as TN Underlay

In a network implementing source routing, packets may be transported through the use of Segment Identifiers (SIDs), where a SID uniquely identifies a segment as defined in [I-D.ietf-spring-segment-routing]. Section 5.3 [I-D.bogineni-dmm-optimized-mobile-user-plane] lays out all SRv6 features along with a few concerns in Section 5.3.7 of the same document. Those concerns as well as need for underlay agnostic (L2/Ipv4/IPv6/MPLS) TE requirements are addressed by a new XHaul

routing mechanism called Preferred Path Routing (PPR), of which this section provides an overview.

With PPR, the label/PPR-ID refer not to individual segments of which the path is composed, but to the identifier of a path that is deployed on network nodes. The fact that paths and path identifiers can be computed and controlled by a controller, not a routing protocol, allows the deployment of any path that network operators prefer, not just shortest paths. As packets refer to a path towards a given destination and nodes make their forwarding decision based on the identifier of a path, not the identifier of a next segment node, it is no longer necessary to carry a sequence of labels. This results in multiple benefits including significant reduction in network layer overhead, increased performance and hardware compatibility for carrying both path and services along the path.

Details of the IGP extensions for PPR are provided here:

- o IS-IS - [I-D.chunduri-lsr-isis-preferred-path-routing]
- o OSPF - [I-D.chunduri-lsr-ospf-preferred-path-routing]

3.1. PPR with Transport Awareness for 5GS on N3/N9 Interfaces

PPR does not remove GTP-U, unlike some other proposals laid out in [I-D.bogineni-dmm-optimized-mobile-user-plane]. Instead, PPR works with the existing cellular user plane (GTP-U) for both N3 and any approach selected for N9 (encapsulation or no-encapsulation). In this scenario, PPR will only help providing TE benefits needed for 5G slices from transport domain perspective. It does so for any underlying data plane used in the transport network (L2/IPv4/IPv6/MPLS). This is achieved by:

- o For 3 different SSTs, 3 PPR-IDs can be signaled from any node in the transport network. For Uplink traffic, the 5G-AN will choose the right PPR-ID of the UPF based on the S-NSSAI the PDU Session belongs to and/or the UDP Source port (corresponds to the MTNC-ID Section 2.5) of the GTP-U encapsulation header. Similarly in the Downlink direction matching PPR-ID of the 5G-AN is chosen based on the S-NSSAI the PDU Session belongs to. The table below shows a typical mapping:

GTP/UDP SRC PORT	SST in S-NSSAI	Transport Path Info	Transport Path Characteristics
Range Xx - Xy X1, X2 (discrete values)	MIOT (massive IOT)	PW ID/VPN info, PPR-ID-A	GBR (Guaranteed Bit Rate) Bandwidth: Bx Delay: Dx Jitter: Jx
Range Yx - Yy Y1, Y2 (discrete values)	URLLC (ultra-low latency)	PW ID/VPN info, PPR-ID-B	GBR with Delay Req. Bandwidth: By Delay: Dy Jitter: Jy
Range Zx - Zy Z1, Z2 (discrete values)	EMBB (broadband)	PW ID/VPN info, PPR-ID-C	Non-GBR Bandwidth: Bx

Figure 2: Mapping of PPR-IDs on N3/N9

- o It is possible to have a single PPR-ID for multiple input points through a PPR tree structure separate in UL and DL direction.
- o Same set of PPRs are created uniformly across all needed 5G-ANs and UPFs to allow various mobility scenarios.
- o Any modification of TE parameters of the path, replacement path and deleted path needed to be updated from TNF to the relevant ingress points. Same information can be pushed to the NSSF, and/or SMF as needed.
- o PPR can be supported with any native IPv4 and IPv6 data/user planes (Section 3.2) with optional TE features (Section 3.3) . As this is an underlay mechanism it can work with any overlay encapsulation approach including GTP-U as defined currently for N3 interface.

3.2. Path Steering Support to native IP user planes

PPR works in fully compatible way with SR defined user planes (SR-MPLS and SRv6) by reducing the path overhead and other challenges as listed in Section 5.3.7 of [I-D.bogineni-dmm-optimized-mobile-user-plane]. PPR also expands the source routing to user planes beyond SR-MPLS and SRv6 i.e., L2, native IPv6 and IPv4 user planes.

This helps legacy transport networks to get the immediate path steering benefits and helps in overall migration strategy of the network to the desired user plane. Some of these benefits with PPR can be realized with no hardware upgrade except control plane software for native IPv6 and IPv4 user planes.

3.3. Service Level Guarantee in Underlay

PPR also optionally allows to allocate resources that are to be reserved along the preferred path. These resources are required in some cases (for some 5G SSTs with stringent GBR and latency requirements) not only for providing committed bandwidth or deterministic latency, but also for assuring overall service level guarantee in the network. This approach does not require per-hop provisioning and reduces the OPEX by minimizing the number of protocols needed and allows dynamism with Fast-ReRoute (FRR) capabilities.

4. Other TE Technologies Applicability

RSVP-TE [RFC3209] provides a lean transport overhead for the TE path for MPLS user plane. However, it is perceived as less dynamic in some cases and has some provisioning overhead across all the nodes in N3 and N9 interface nodes. Also it has another drawback with excessive state refresh overhead across adjacent nodes and this can be mitigated with [RFC8370].

SR-TE [I-D.ietf-spring-segment-routing] does not explicitly signal bandwidth reservation or mechanism to guarantee latency on the nodes/links on SR path. But, SR allows path steering for any flow at the ingress and particular path for a flow can be chosen. Some of the issues around path overhead/tax, MTU issues are documented at Section 5.3 of [I-D.bogineni-dmm-optimized-mobile-user-plane]. SR-MPLS allows reduction of the control protocols to one IGP (with out needing for LDP and RSVP-TE).

However, as specified above with PPR (Section 3), in the integrated transport network function (TNF) a particular RSVP-TE path for MPLS

or SR path for MPLS and IPv6 with SRH user plane, can be supplied to SMF for mapping a particular PDU session to the transport path.

5. Acknowledgements

Thanks to Young Lee for discussions on this document including ACTN applicability for the proposed TNF. Thanks to Sri Gundavelli and 3GPP delegates who provided detailed feedback on this document.

6. IANA Considerations

This document has no requests for any IANA code point allocations.

7. Security Considerations

This document does not introduce any new security issues.

8. Contributing Authors

The following people contributed substantially to the content of this document and should be considered co-authors.

Xavier De Foy
InterDigital Communications, LLC
1000 Sherbrooke West
Montreal
Canada

Email: Xavier.Defoy@InterDigital.com

9. References

9.1. 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>>.

9.2. Informative References

[ATIS075] Alliance for Telecommunications Industry Solutions (ATIS), "IOT Categorization: Exploring the Need for Standardizing Additional Network Slices ATIS-I-0000075", September 2019.

- [I-D.bashandy-rtgwg-segment-routing-ti-lfa]
Bashandy, A., Filsfils, C., Decraene, B., Litkowski, S., Francois, P., daniel.voyer@bell.ca, d., Clad, F., and P. Camarillo, "Topology Independent Fast Reroute using Segment Routing", draft-bashandy-rtgwg-segment-routing-ti-lfa-05 (work in progress), October 2018.
- [I-D.bogineni-dmm-optimized-mobile-user-plane]
Bogineni, K., Akhavain, A., Herbert, T., Farinacci, D., Rodriguez-Natal, A., Carofiglio, G., Auge, J., Muscariello, L., Camarillo, P., and S. Homma, "Optimized Mobile User Plane Solutions for 5G", draft-bogineni-dmm-optimized-mobile-user-plane-01 (work in progress), June 2018.
- [I-D.chunduri-lsr-isis-preferred-path-routing]
Chunduri, U., Li, R., White, R., Tantsura, J., Contreras, L., and Y. Qu, "Preferred Path Routing (PPR) in IS-IS", draft-chunduri-lsr-isis-preferred-path-routing-04 (work in progress), July 2019.
- [I-D.chunduri-lsr-ospf-preferred-path-routing]
Chunduri, U., Qu, Y., White, R., Tantsura, J., and L. Contreras, "Preferred Path Routing (PPR) in OSPF", draft-chunduri-lsr-ospf-preferred-path-routing-03 (work in progress), May 2019.
- [I-D.farinacci-lisp-mobile-network]
Farinacci, D., Pillay-Esnault, P., and U. Chunduri, "LISP for the Mobile Network", draft-farinacci-lisp-mobile-network-06 (work in progress), September 2019.
- [I-D.ietf-dmm-5g-uplane-analysis]
Homma, S., Miyasaka, T., Matsushima, S., and D. Voyer, "User Plane Protocol and Architectural Analysis on 3GPP 5G System", draft-ietf-dmm-5g-uplane-analysis-02 (work in progress), July 2019.
- [I-D.ietf-dmm-srv6-mobile-uplane]
Matsushima, S., Filsfils, C., Kohno, M., Camarillo, P., Voyer, D., and C. Perkins, "Segment Routing IPv6 for Mobile User Plane", draft-ietf-dmm-srv6-mobile-uplane-06 (work in progress), September 2019.
- [I-D.ietf-intarea-gue-extensions]
Herbert, T., Yong, L., and F. Templin, "Extensions for Generic UDP Encapsulation", draft-ietf-intarea-gue-extensions-06 (work in progress), March 2019.

- [I-D.ietf-spring-segment-routing]
Filsfils, C., Previdi, S., Ginsberg, L., Decraene, B.,
Litkowski, S., and R. Shakir, "Segment Routing
Architecture", draft-ietf-spring-segment-routing-15 (work
in progress), January 2018.
- [IR.34-GSMA]
GSM Association (GSMA), "Guidelines for IPX Provider
Networks (Previously Inter-Service Provider IP Backbone
Guidelines, Version 14.0", August 2018.
- [RFC3209] Awduche, D., Berger, L., Gan, D., Li, T., Srinivasan, V.,
and G. Swallow, "RSVP-TE: Extensions to RSVP for LSP
Tunnels", RFC 3209, DOI 10.17487/RFC3209, December 2001,
<<https://www.rfc-editor.org/info/rfc3209>>.
- [RFC5440] Vasseur, JP., Ed. and JL. Le Roux, Ed., "Path Computation
Element (PCE) Communication Protocol (PCEP)", RFC 5440,
DOI 10.17487/RFC5440, March 2009,
<<https://www.rfc-editor.org/info/rfc5440>>.
- [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>>.
- [RFC6830] Farinacci, D., Fuller, V., Meyer, D., and D. Lewis, "The
Locator/ID Separation Protocol (LISP)", RFC 6830,
DOI 10.17487/RFC6830, January 2013,
<<https://www.rfc-editor.org/info/rfc6830>>.
- [RFC7490] Bryant, S., Filsfils, C., Previdi, S., Shand, M., and N.
So, "Remote Loop-Free Alternate (LFA) Fast Reroute (FRR)",
RFC 7490, DOI 10.17487/RFC7490, April 2015,
<<https://www.rfc-editor.org/info/rfc7490>>.
- [RFC7752] Gredler, H., Ed., Medved, J., Previdi, S., Farrel, A., and
S. Ray, "North-Bound Distribution of Link-State and
Traffic Engineering (TE) Information Using BGP", RFC 7752,
DOI 10.17487/RFC7752, March 2016,
<<https://www.rfc-editor.org/info/rfc7752>>.
- [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>>.

- [RFC8370] Beeram, V., Ed., Minei, I., Shakir, R., Pacella, D., and T. Saad, "Techniques to Improve the Scalability of RSVP-TE Deployments", RFC 8370, DOI 10.17487/RFC8370, May 2018, <<https://www.rfc-editor.org/info/rfc8370>>.
- [RFC8453] Ceccarelli, D., Ed. and Y. Lee, Ed., "Framework for Abstraction and Control of TE Networks (ACTN)", RFC 8453, DOI 10.17487/RFC8453, August 2018, <<https://www.rfc-editor.org/info/rfc8453>>.
- [TS.23.401-3GPP]
3rd Generation Partnership Project (3GPP), "Procedures for 4G/LTE System; 3GPP TS 23.401, v15.4.0", June 2018.
- [TS.23.501-3GPP]
3rd Generation Partnership Project (3GPP), "System Architecture for 5G System; Stage 2, 3GPP TS 23.501 v2.0.1", December 2017.
- [TS.23.502-3GPP]
3rd Generation Partnership Project (3GPP), "Procedures for 5G System; Stage 2, 3GPP TS 23.502, v2.0.0", December 2017.
- [TS.23.503-3GPP]
3rd Generation Partnership Project (3GPP), "Policy and Charging Control System for 5G Framework; Stage 2, 3GPP TS 23.503 v1.0.0", December 2017.
- [TS.28.533-3GPP]
3rd Generation Partnership Project (3GPP), "Management and Orchestration Architecture Framework (Release 15)", June 2018.
- [TS.29.281-3GPP]
3rd Generation Partnership Project (3GPP), "GPRS Tunneling Protocol User Plane (GTPv1-U), 3GPP TS 29.281 v15.1.0", December 2018.

Appendix A. New Control Plane and User Planes

A.1. Slice aware Mobility: Discrete Approach

In this approach transport network functionality from the 5G-AN to UPF is discrete and 5GS is not aware of the underlying transport network and the resources available. Deployment specific mapping function is used to map the GTP-U encapsulated traffic at the 5G-AN (e.g. gNB) in UL and UPF in DL direction to the appropriate transport

slice or transport Traffic Engineered (TE) paths. These TE paths can be established using RSVP-TE [RFC3209] for MPLS underlay, SR [I-D.ietf-spring-segment-routing] for both MPLS and IPv6 underlay or PPR [I-D.chunduri-lsr-isis-preferred-path-routing] with MPLS, IPv6 with SRH, native IPv6 and native IPv4 underlays.

As per [TS.23.501-3GPP] and [TS.23.502-3GPP] the SMF controls the user plane traffic forwarding rules in the UPF. The UPFs have a concept of a "Network Instance" which logically abstracts the underlying transport path. When the SMF creates the packet detection rules (PDR) and forwarding action rules (FAR) for a PDU session at the UPF, the SMF identifies the network instance through which the packet matching the PDR has to be forwarded. A network instance can be mapped to a TE path at the UPF. In this approach, TNF as shown in Figure 1 need not be part of the 5G Service Based Interface (SBI). Only management plane functionality is needed to create, monitor, manage and delete (life cycle management) the transport TE paths/transport slices from the 5G-AN to the UPF (on N3/N9 interfaces). The management plane functionality also provides the mapping of such TE paths to a network instance identifier to the SMF. The SMF uses this mapping to install appropriate FARs in the UPF. This approach provides partial integration of the transport network into 5GS with some benefits.

One of the limitations of this approach is the inability of the 5GS procedures to know, if underlying transport resources are available for the traffic type being carried in PDU session before making certain decisions in the 5G CP. One example scenario/decision could be, a target 5G-AN selection during a N2 mobility event, without knowing if the target 5G-AN is having a underlay transport slice resource for the S-NSSAI and 5QI of the PDU session. The Integrated approach specified below can mitigate this.

Appendix B. PPR with various 5G Mobility procedures

PPR fulfills the needs of 5GS to transport the user plane traffic from 5G-AN to UPF in all 3 SSC modes defined [TS.23.501-3GPP]. This is done in keeping the backhaul network at par with 5G slicing requirements that are applicable to Radio and virtualized core network to create a truly end-to-end slice path for 5G traffic. When UE moves across the 5G-AN (e.g. from one gNB to another gNB), there is no transport network reconfiguration required with the approach above.

SSC mode would be specified/defaulted by SMF. No change in the mode once connection is initiated and this property is not altered here.

B.1. SSC Model

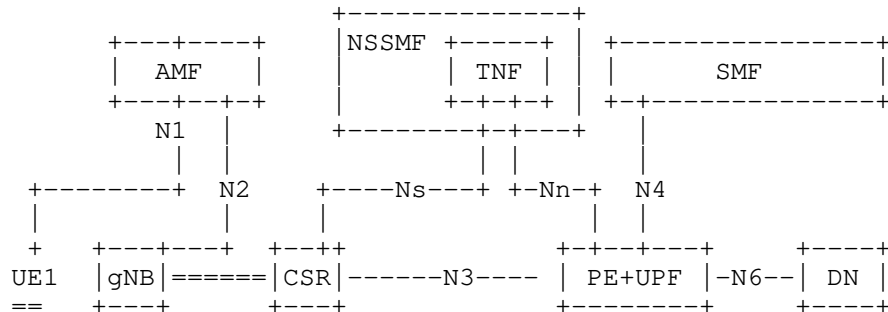


Figure 3: SSC Model with integrated Transport Slice Function

After UE1 moved to another gNB in the same UPF serving area

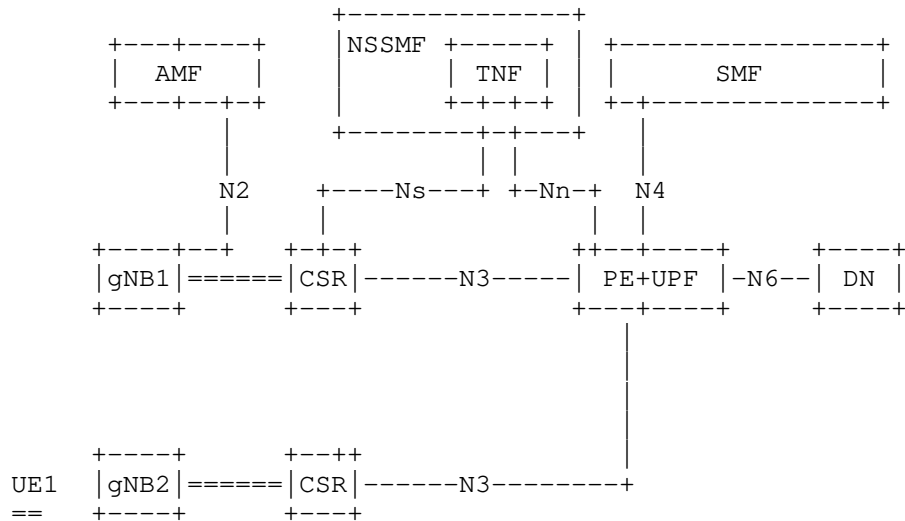


Figure 4: SSC Model with integrated Transport Slice Function

In this mode, IP address at the UE is preserved during mobility events. This is similar to 4G/LTE mechanism and for respective slices, corresponding PPR-ID (TE Path) has to be assigned to the packet at UL and DL direction. During Xn mobility as shown above,

source gNB has to additionally ensure transport path's resources from TNF are available at the target gNB apart from radio resources check (at decision and request phase of Xn/N2 mobility scenario).

B.2. SSC Mode2

In this case, if IP Address is changed during mobility (different UPF area), then corresponding PDU session is released. No session continuity from the network is provided and this is designed as an application offload and application manages the session continuity, if needed. For PDU Session, Service Request and Mobility cases mechanism to select the transport resource and the PPR-ID (TE Path) is similar to SSC Model.

B.3. SSC Mode3

In this mode, new IP address may be assigned because of UE moved to another UPF coverage area. Network ensures UE suffers no loss of 'connectivity'. A connection through new PDU session anchor point is established before the connection is terminated for better service continuity. There are two ways in which this happens.

- o Change of SSC Mode 3 PDU Session Anchor with multiple PDU Sessions.
- o Change of SSC Mode 3 PDU Session Anchor with IPv6 multi-homed PDU Session.

In the first mode, from user plane perspective, the two PDU sessions are independent and the use of PPR-ID by gNB and UPFs is exactly similar to SSC Mode 1 described above. The following paragraphs describe the IPv6 multi-homed PDU session case for SSC Mode 3.

- o For Service continuity with multi-homed PDU session same transport network characteristics of the original PDU session (both on N3 and N9) need to be observed for the newly configured IPv6 prefixes.

Authors' Addresses

Uma Chunduri (editor)
Futurewei
2330 Central Expressway
Santa Clara, CA 95050
USA

Email: umac.ietf@gmail.com

Richard Li
Futurewei
2330 Central Expressway
Santa Clara, CA 95050
USA

Email: richard.li@futurewei.com

Sridhar Bhaskaran
AltioStar

Email: sridharb@altioStar.com

John Kaippallimalil
Futurewei

Email: john.kaippallimalil@futurewei.com

Jeff Tantsura
Apstra, Inc.

Email: jefftant.ietf@gmail.com

Luis M. Contreras
Telefonica
Sur-3 building, 3rd floor
Madrid 28050
Spain

Email: luismiguel.contrerasmurillo@telefonica.com

Praveen Muley
Nokia
440 North Bernardo Ave
Mountain View, CA 94043
USA

Email: praveen.muley@nokia.com

DMM Working Group
Internet-Draft
Intended status: Standards Track
Expires: October 13, 2019

S. Matsushima
SoftBank
L. Bertz
Sprint
M. Liebsch
NEC
S. Gundavelli
Cisco
D. Moses
Intel Corporation
C. Perkins
Futurewei
April 11, 2019

YANG for Protocol for Forwarding Policy Configuration (FPC)
draft-ietf-dmm-fpc-yang-00

Abstract

This document provides YANG modules to exhibit a data model for the information models defining 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.

Status of This Memo

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

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

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

This Internet-Draft will expire on October 13, 2019.

Copyright Notice

Copyright (c) 2019 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 (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Terminology	2
3. YANG Data Model for the FPC protocol	3
3.1. FPC YANG Model	5
3.2. FPC YANG Settings and Extensions Model	27
3.3. PMIP QoS Model	39
3.4. Traffic Selectors YANG Model	46
3.5. RFC 5777 Classifier YANG Model	54
4. FPC YANG Tree Structure	62
5. Work Team Participants	80
6. References	80
6.1. Normative References	80
6.2. Informative References	81
Authors' Addresses	81

1. Introduction

This document provides YANG modules to exhibit a data model for the information models defining Forwarding Policy Configuration (FPC) [I-D.ietf-dmm-fpc-cpdp] 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.

2. Terminology

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

following FPC-specific terms used in this document are defined in [I-D.ietf-dmm-fpc-cpdp].

- o Domain
- o DPN
- o FPC Agent
- o FPC Client
- o Mobility Context
- o Monitor
- o Policy
- o Template
- o Tenant
- o Topology

3. 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.

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 in a 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 [RFC7222]
- 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)
- o qosattribute (ietf-qos-pmip module)
- o qosoption (ietf-qos-pmip module)
- o Allocation-Retention-Priority-Value (ietf-qos-pmip module)
- o trafficselector (ietf-traffic-selector-types)
- o classifier (ietf-diam-trafficclassifier)
- o packet-filter (ietf-dmm-fpc-settingsext)
- o instructions (ietf-dmm-fpc-settingsext)
- o fpc-descriptor-value (ietf-dmm-fpc-settingsext)
- o fpc-action-value (ietf-dmm-fpc-settingsext)

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

A DPN 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.

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.

3.1. 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-cdp-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 template.";  
  }  
  leaf entity-state {
```



```
    type enumeration {
      enum initial {
        description "Initial Configuration";
      }
      enum partially-configured {
        description "Partial Configuration";
      }
      enum configured {
        description "Configured";
      }
      enum active {
        description "Active";
      }
    }
    default initial;
    description "Entity State";
  }
  leaf version {
    type uint32;
    description "Template Version";
  }
  description "Teemplate 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 Occurrence";
}
identity low-threshold-crossed {
  base "notification-cause";
  description "Subscribed Event Occurrence";
}
identity high-threshold-crossed {
  base "notification-cause";
  description "Subscribed Event Occurrence";
}
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>
```

3.2. 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 fpabase;

  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 definitions 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>
```

3.3. 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 parameters 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>

```

3.4. 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 common 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-label 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>

```

3.5. RFC 5777 Classifier YANG Model

This module defines the RFC 5777 Classifier.

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>

```

4. 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 1: YANG FPC Agent Tree

5. 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.

6. References

6.1. 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>>.
- [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>>.
- [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>>.
- [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>>.

6.2. Informative References

- [I-D.ietf-dmm-fpc-cpdp]
Matsushima, S., Bertz, L., Liebsch, M., Gundavelli, S.,
Moses, D., and C. Perkins, "Protocol for Forwarding Policy
Configuration (FPC) in DMM", draft-ietf-dmm-fpc-cpdp-12
(work in progress), June 2018.
- [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>>.
- [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>>.

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

DMM Working Group
Internet-Draft
Intended status: Standards Track
Expires: 10 November 2022

S. Matsushima, Ed.
SoftBank
C. Filsfils
M. Kohno
P. Camarillo, Ed.
Cisco Systems, Inc.
D. Voyer
Bell Canada
C.E. Perkins
Lupin Lodge
9 May 2022

Segment Routing IPv6 for Mobile User Plane
draft-ietf-dmm-srv6-mobile-uplane-21

Abstract

This document specifies the applicability of SRv6 (Segment Routing IPv6) to the user-plane of mobile networks. The network programming nature of SRv6 accomplishes mobile user-plane functions in a simple manner. The statelessness of SRv6 and its ability to control both service layer path and underlying transport can be beneficial to the mobile user-plane, providing flexibility, end-to-end network slicing, and SLA control for various applications.

Status of This Memo

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

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

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

This Internet-Draft will expire on 10 November 2022.

Copyright Notice

Copyright (c) 2022 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 (<https://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 Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

Table of Contents

1. Introduction	3
2. Conventions and Terminology	3
2.1. Terminology	3
2.2. Conventions	4
2.3. Predefined SRv6 Endpoint Behaviors	4
3. Motivation	5
4. 3GPP Reference Architecture	5
5. User-plane modes	6
5.1. Traditional mode	7
5.1.1. Packet flow - Uplink	8
5.1.2. Packet flow - Downlink	9
5.2. Enhanced mode	9
5.2.1. Packet flow - Uplink	10
5.2.2. Packet flow - Downlink	11
5.2.3. Scalability	12
5.3. Enhanced mode with unchanged gNB GTP behavior	12
5.3.1. Interworking with IPv6 GTP	12
5.3.2. Interworking with IPv4 GTP	15
5.3.3. Extensions to the interworking mechanisms	18
5.4. SRv6 Drop-in Interworking	18
6. SRv6 Segment Endpoint Mobility Behaviors	19
6.1. Args.Mob.Session	20
6.2. End.MAP	20
6.3. End.M.GTP6.D	21
6.4. End.M.GTP6.D.Di	22
6.5. End.M.GTP6.E	23
6.6. End.M.GTP4.E	24
6.7. H.M.GTP4.D	25
6.8. End.Limit: Rate Limiting behavior	26
7. SRv6 supported 3GPP PDU session types	27
8. Network Slicing Considerations	27
9. Control Plane Considerations	27
10. Security Considerations	28
11. IANA Considerations	28
12. Acknowledgements	29
13. Contributors	29
14. References	29

14.1. Normative References	29
14.2. Informative References	30
Appendix A. Implementations	32
Authors' Addresses	32

1. Introduction

In mobile networks, mobility systems provide connectivity over a wireless link to stationary and non-stationary nodes. The user-plane establishes a tunnel between the mobile node and its anchor node over IP-based backhaul and core networks.

This document specifies the applicability of SRv6 (Segment Routing IPv6) to mobile networks.

Segment Routing [RFC8402] is a source routing architecture: a node steers a packet through an ordered list of instructions called "segments". A segment can represent any instruction, topological or service based.

SRv6 applied to mobile networks enables a source-routing based mobile architecture, where operators can explicitly indicate a route for the packets to and from the mobile node. The SRv6 Endpoint nodes serve as mobile user-plane anchors.

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].

2.1. Terminology

- * CNF: Cloud-native Network Function
- * NFV: Network Function Virtualization
- * PDU: Packet Data Unit
- * PDU Session: Context of a UE connects to a mobile network.
- * UE: User Equipment
- * UPF: User Plane Function
- * VNF: Virtual Network Function (including CNFs)

The following terms used within this document are defined in [RFC8402]: Segment Routing, SR Domain, Segment ID (SID), SRv6, SRv6 SID, Active Segment, SR Policy, Prefix SID, Adjacency SID and Binding SID.

The following terms used within this document are defined in [RFC8754]: SRH, SR Source Node, Transit Node, SR Segment Endpoint Node and Reduced SRH.

The following terms used within this document are defined in [RFC8986]: NH, SL, FIB, SA, DA, SRv6 SID behavior, SRv6 Segment Endpoint Behavior.

2.2. Conventions

An SR Policy is resolved to a SID list. A SID list is represented as <S1, S2, S3> where S1 is the first SID to visit, S2 is the second SID to visit, and S3 is the last SID to visit along the SR path.

(SA,DA) (S3, S2, S1; SL) represents an IPv6 packet with:

- * Source Address is SA, Destination Address is DA, and next-header is SRH
- * SRH with SID list <S1, S2, S3> with Segments Left = SL
- * Note the difference between the <> and () symbols: <S1, S2, S3> represents a SID list where S1 is the first SID and S3 is the last SID to traverse. (S3, S2, S1; SL) represents the same SID list but encoded in the SRH format where the rightmost SID in the SRH is the first SID and the leftmost SID in the SRH is the last SID. When referring to an SR policy in a high-level use-case, it is simpler to use the <S1, S2, S3> notation. When referring to an illustration of the detailed packet behavior, the (S3, S2, S1; SL) notation is more convenient.
- * The payload of the packet is omitted.

SRH[n]: A shorter representation of Segment List[n], as defined in [RFC8754]. SRH[SL] can be different from the DA of the IPv6 header.

- * gNB::1 is an IPv6 address (SID) assigned to the gNB.
- * U1::1 is an IPv6 address (SID) assigned to UPF1.
- * U2::1 is an IPv6 address (SID) assigned to UPF2.
- * U2:: is the Locator of UPF2.

2.3. Predefined SRv6 Endpoint Behaviors

The following SRv6 Endpoint Behaviors are defined in [RFC8986].

- * End.DT4: Decapsulation and Specific IPv4 Table Lookup
- * End.DT6: Decapsulation and Specific IPv6 Table Lookup
- * End.DT46: Decapsulation and Specific IP Table Lookup
- * End.DX4: Decapsulation and IPv4 Cross-Connect
- * End.DX6: Decapsulation and IPv6 Cross-Connect
- * End.DX2: Decapsulation and L2 Cross-Connect

* End.T: Endpoint with specific IPv6 Table Lookup

This document defines new SRv6 Segment Endpoint Behaviors in Section 6.

3. Motivation

Mobile networks are becoming more challenging to operate. On one hand, traffic is constantly growing, and latency requirements are tighter; on the other-hand, there are new use-cases like distributed NFVi that are also challenging network operations.

The current architecture of mobile networks does not take into account the underlying transport. The user-plane is rigidly fragmented into radio access, core and service networks, connected by tunneling according to user-plane roles such as access and anchor nodes. These factors have made it difficult for the operator to optimize and operate the data-path.

In the meantime, applications have shifted to use IPv6, and network operators have started adopting IPv6 as their IP transport. SRv6, the IPv6 dataplane instantiation of Segment Routing [RFC8402], integrates both the application data-path and the underlying transport layer into a single protocol, allowing operators to optimize the network in a simplified manner and removing forwarding state from the network. It is also suitable for virtualized environments, like VNF/CNF to VNF/CNF networking. SRv6 has been deployed in dozens of networks [I-D.matsushima-spring-srv6-deployment-status].

SRv6 defines the network-programming concept [RFC8986]. Applied to mobility, SRv6 can provide the user-plane behaviors needed for mobility management. SRv6 takes advantage of the underlying transport awareness and flexibility together with the ability to also include services to optimize the end-to-end mobile dataplane.

The use-cases for SRv6 mobility are discussed in [I-D.camarilloelmalaky-springdmm-srv6-mob-usecases], and the architectural benefits are discussed in [I-D.kohno-dmm-srv6mob-arch].

4. 3GPP Reference Architecture

This section presents a reference architecture and possible deployment scenarios.

Figure 1 shows a reference diagram from the 5G packet core architecture [TS.23501].

The user plane described in this document does not depend on any specific architecture. The 5G packet core architecture as shown is based on the latest 3GPP standards at the time of writing this draft.

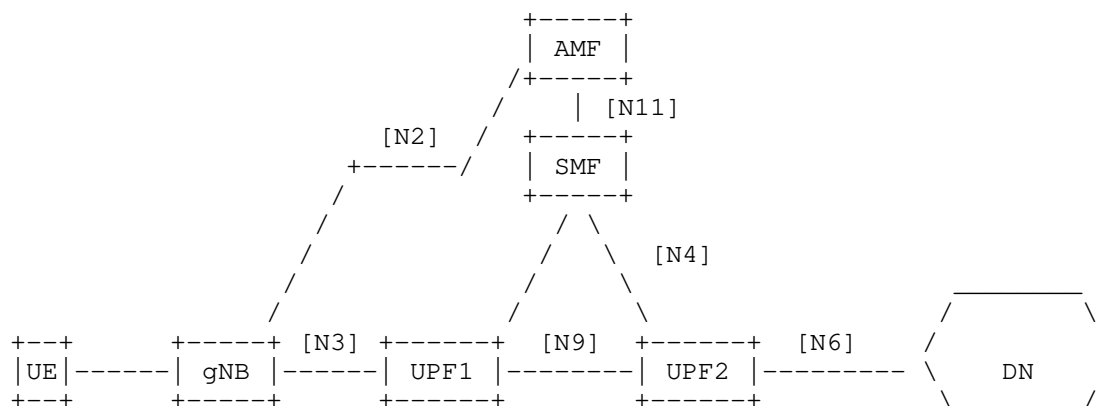


Figure 1: 3GPP 5G Reference Architecture

- * UE: User Endpoint
- * gNB: gNodeB with N3 interface towards packet core (and N2 for control plane)
- * UPF1: UPF with Interfaces N3 and N9 (and N4 for control plane)
- * UPF2: UPF with Interfaces N9 and N6 (and N4 for control plane)
- * SMF: Session Management Function
- * AMF: Access and Mobility Management Function
- * DN: Data Network e.g. operator services, Internet access

This reference diagram does not depict a UPF that is only connected to N9 interfaces, although the mechanisms defined in this document also work in such case.

Each session from a UE gets assigned to a UPF. Sometimes multiple UPFs may be used, providing richer service functions. A UE gets its IP address from the DHCP block of its UPF. The UPF advertises that IP address block toward the Internet, ensuring that return traffic is routed to the right UPF.

5. User-plane modes

This section introduces an SRv6 based mobile user-plane.

In order to simplify the adoption of SRv6, we present two different "modes" that vary with respect to the use of SRv6. The first one is the "Traditional mode", which inherits the current 3GPP mobile architecture. In this mode GTP-U protocol [TS.29281] is replaced by

SRv6, however the N3, N9 and N6 interfaces are still point-to-point interfaces with no intermediate waypoints as in the current mobile network architecture.

The second mode is the "Enhanced mode". This is an evolution from the "Traditional mode". In this mode the N3, N9 or N6 interfaces have intermediate waypoints -SIDs- that are used for Traffic Engineering or VNF purposes transparent to 3GPP functionalities. This results in optimal end-to-end policies across the mobile network with transport and services awareness.

In both, the Traditional and the Enhanced modes, we assume that the gNB as well as the UPFs are SR-aware (N3, N9 and -potentially- N6 interfaces are SRv6).

In addition to those two modes, we introduce two mechanisms for interworking with legacy access networks (those where the N3 interface is unmodified). In this document we introduce them as a variant to the Enhanced mode, however they are equally applicable to the Traditional mode.

One of these mechanisms is designed to interwork with legacy gNBs using GTP/IPv4. The second mechanism is designed to interwork with legacy gNBs using GTP/IPv6.

This document uses SRv6 Segment Endpoint Behaviors defined in [RFC8986] as well as new SRv6 Segment Endpoint Behaviors designed for the mobile user plane that are defined in this document in Section 6.

Note that the modes discussed throughout this section (with the exception of Section 5.4) only have informational purpose to implementors as well as operators deploying this technology. Indeed, it is expected that the operator defines his own operational model that best suits their needs.

5.1. Traditional mode

In the traditional mode, the existing mobile UPFs remain unchanged with the sole exception of the use of SRv6 as the data plane instead of GTP-U. There is no impact to the rest of the mobile system.

In existing 3GPP mobile networks, a PDU Session is mapped 1-for-1 with a specific GTP tunnel (TEID). This 1-for-1 mapping is mirrored here to replace GTP encapsulation with the SRv6 encapsulation, while not changing anything else. There will be a unique SRv6 SID associated with each PDU Session, and the SID list only contains a single SID.

The traditional mode minimizes the changes required to the mobile system; hence it is a good starting point for forming a common ground.

The gNB/UPF control-plane (N2/N4 interface) is unchanged, specifically a single IPv6 address is provided to the gNB. The same control plane signalling is used, and the gNB/UPF decides to use SRv6 based on signaled GTP-U parameters per local policy. The only information from the GTP-U parameters used for the SRv6 policy is the TEID, QFI, and the IPv6 Destination Address.

Our example topology is shown in Figure 2. The gNB and the UPFs are SR-aware. In the descriptions of the uplink and downlink packet flow, A is an IPv6 address of the UE, and Z is an IPv6 address reachable within the Data Network DN. A new SRv6 Endpoint Behavior, End.MAP, defined in Section 6.2, is used.

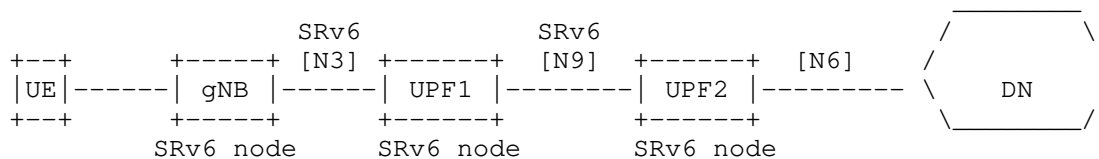


Figure 2: Traditional mode - example topology

5.1.1. Packet flow - Uplink

The uplink packet flow is as follows:

```

UE_out   : (A,Z)
gNB_out  : (gNB, U1::1) (A,Z)    -> H.Encaps.Red <U1::1>
UPF1_out : (gNB, U2::1) (A,Z)    -> End.MAP
UPF2_out : (A,Z)                 -> End.DT4 or End.DT6

```

When the UE packet arrives at the gNB, the gNB performs a H.Encaps.Red operation. Since there is only one SID, there is no need to push an SRH. gNB only adds an outer IPv6 header with IPv6 DA U1::1. gNB obtains the SID U1::1 from the existing control plane (N2 interface). U1::1 represents an anchoring SID specific for that session at UPF1.

When the packet arrives at UPF1, the SID U1::1 is associated with the End.MAP SRv6 Endpoint Behavior. End.MAP replaces U1::1 by U2::1, that belongs to the next UPF (U2).

When the packet arrives at UPF2, the SID U2::1 corresponds to an End.DT4/End.DT6/End.DT46 SRv6 Endpoint Behavior. UPF2 decapsulates the packet, performs a lookup in a specific table associated with that mobile network and forwards the packet toward the data network (DN).

5.1.2. Packet flow - Downlink

The downlink packet flow is as follows:

```
UPF2_in : (Z,A)
UPF2_out: (U2::, U1::2) (Z,A)    -> H.Encaps.Red <U1::2>
UPF1_out: (U2::, gNB::1) (Z,A)   -> End.MAP
gNB_out  : (Z,A)                 -> End.DX4, End.DX6, End.DX2
```

When the packet arrives at the UPF2, the UPF2 maps that flow into a PDU Session. This PDU Session is associated with the segment endpoint <U1::2>. UPF2 performs a H.Encaps.Red operation, encapsulating the packet into a new IPv6 header with no SRH since there is only one SID.

Upon packet arrival on UPF1, the SID U1::2 is a local SID associated with the End.MAP SRv6 Endpoint Behavior. It maps the SID to the next anchoring point and replaces U1::2 by gNB::1, that belongs to the next hop.

Upon packet arrival on gNB, the SID gNB::1 corresponds to an End.DX4, End.DX6 or End.DX2 behavior (depending on the PDU Session Type). The gNB decapsulates the packet, removing the IPv6 header and all its extensions headers, and forwards the traffic toward the UE.

5.2. Enhanced mode

Enhanced mode improves scalability, provides traffic engineering capabilities, and allows service programming [I-D.ietf-spring-sr-service-programming], thanks to the use of multiple SIDs in the SID list (instead of a direct connectivity in between UPFs with no intermediate waypoints as in Traditional Mode).

Thus, the main difference is that the SR policy MAY include SIDs for traffic engineering and service programming in addition to the anchoring SIDs at UPFs.

Additionally in this mode the operator may choose to aggregate several devices under the same SID list (e.g., stationary residential meters connected to the same cell) to improve scalability.

The gNB/UPF control-plane (N2/N4 interface) is unchanged, specifically a single IPv6 address is provided to the gNB. A local policy instructs the gNB to use SRv6.

The gNB MAY resolve the IP address received via the control plane into a SID list using a mechanism like PCEP, DNS-lookup, LISP control-plane or others. The resolution mechanism is out of the scope of this document.

Note that the SIDs MAY use the arguments Args.Mob.Session if required by the UPFs.

Figure 3 shows an Enhanced mode topology. The gNB and the UPF are SR-aware. The Figure shows two service segments, S1 and C1. S1 represents a VNF in the network, and C1 represents an intermediate router used for Traffic Engineering purposes to enforce a low-latency path in the network. Note that neither S1 nor C1 are required to have an N4 interface.

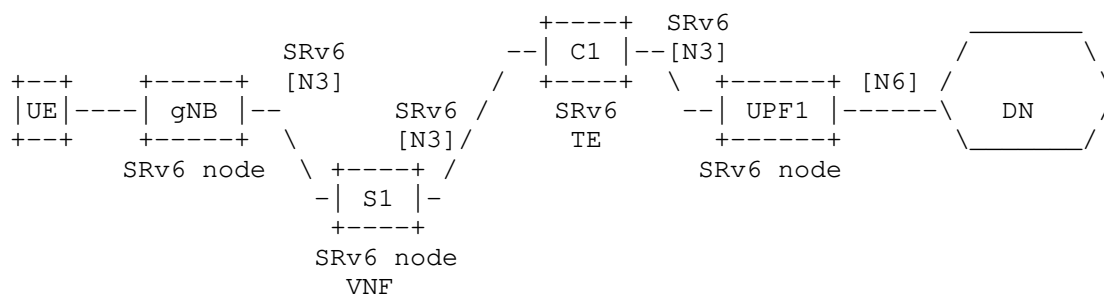


Figure 3: Enhanced mode - Example topology

5.2.1. Packet flow - Uplink

The uplink packet flow is as follows:

```

UE_out   : (A,Z)
gNB_out  : (gNB, S1) (U1::1, C1; SL=2) (A,Z)->H.Encaps.Red<S1,C1,U1::1>
S1_out   : (gNB, C1) (U1::1, C1; SL=1) (A,Z)
C1_out   : (gNB, U1::1) (A,Z)                ->End with PSP
UPF1_out : (A,Z)                            ->End.DT4,End.DT6,End.DT2U

```

UE sends its packet (A,Z) on a specific bearer to its gNB. gNB's control plane associates that session from the UE(A) with the IPv6 address B. gNB's control plane does a lookup on B to find the related SID list <S1, C1, U1::1>.

When gNB transmits the packet, it contains all the segments of the SR policy. The SR policy includes segments for traffic engineering (C1) and for service programming (S1).

Nodes S1 and C1 perform their related Endpoint functionality and forward the packet.

When the packet arrives at UPF1, the active segment (U1::1) is an End.DT4/End.DT6/End.DT2U which performs the decapsulation (removing the IPv6 header with all its extension headers) and forwards toward the data network.

5.2.2. Packet flow - Downlink

The downlink packet flow is as follows:

```

UPF1_in : (Z,A)                                ->UPF1 maps the flow w/
                                                SID list <C1,S1, gNB>
UPF1_out: (U1::1, C1) (gNB::1, S1; SL=2) (Z,A) ->H.Encaps.Red
C1_out  : (U1::1, S1) (gNB::1, S1; SL=1) (Z,A)
S1_out  : (U1::1, gNB::1) (Z,A)                ->End with PSP
gNB_out : (Z,A)                                ->End.DX4/End.DX6/End.DX2

```

When the packet arrives at the UPF1, the UPF1 maps that particular flow into a UE PDU Session. This UE PDU Session is associated with the policy <C1, S1, gNB>. The UPF1 performs a H.Encaps.Red operation, encapsulating the packet into a new IPv6 header with its corresponding SRH.

The nodes C1 and S1 perform their related Endpoint processing.

Once the packet arrives at the gNB, the IPv6 DA corresponds to an End.DX4, End.DX6 or End.DX2 behavior at the gNB (depending on the underlying traffic). The gNB decapsulates the packet, removing the IPv6 header, and forwards the traffic towards the UE. The SID gNB::1 is one example of a SID associated to this service.

Note that there are several means to provide the UE session aggregation. The decision on which one to use is a local decision made by the operator. One option is to use the Args.Mob.Session (Section 6.1). Another option comprises the gNB performing an IP lookup on the inner packet by using the End.DT4, End.DT6, and End.DT2 behaviors.

5.2.3. Scalability

The Enhanced Mode improves since it allows the aggregation of several UEs under the same SID list. For example, in the case of stationary residential meters that are connected to the same cell, all such devices can share the same SID list. This improves scalability compared to Traditional Mode (unique SID per UE) and compared to GTP-U (dedicated TEID per UE).

5.3. Enhanced mode with unchanged gNB GTP behavior

This section describes two mechanisms for interworking with legacy gNBs that still use GTP: one for IPv4, and another for IPv6.

In the interworking scenarios as illustrated in Figure 4, the gNB does not support SRv6. The gNB supports GTP encapsulation over IPv4 or IPv6. To achieve interworking, an SR Gateway (SRGW) entity is added. The SRGW maps the GTP traffic into SRv6.

The SRGW is not an anchor point and maintains very little state. For this reason, both IPv4 and IPv6 methods scale to millions of UEs.

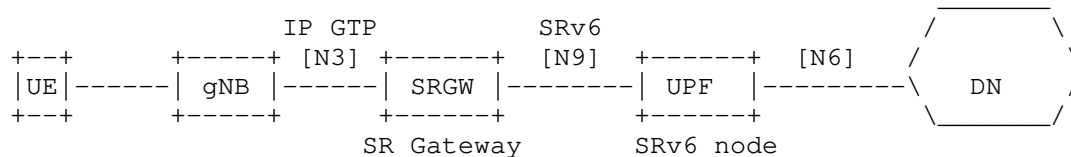


Figure 4: Example topology for interworking

Both of the mechanisms described in this section are applicable to either the Traditional Mode or the Enhanced Mode.

5.3.1. Interworking with IPv6 GTP

In this interworking mode the gNB at the N3 interface uses GTP over IPv6.

Key points:

- * The gNB is unchanged (control-plane or user-plane) and encapsulates into GTP (N3 interface is not modified).
- * The 5G Control-Plane towards the gNB (N2 interface) is unmodified, though multiple UPF addresses need to be used - one IPv6 address (i.e. a BSID at the SRGW) is needed per <SLA, PDU session type>. The SRv6 SID is different depending on the required <SLA, PDU session type> combination.

- * In the uplink, the SRGW removes GTP, finds the SID list related to the IPv6 DA, and adds SRH with the SID list.
- * There is no state for the downlink at the SRGW.
- * There is simple state in the uplink at the SRGW; using Enhanced mode results in fewer SR policies on this node. An SR policy is shared across UEs as long as they belong to the same context (i.e., tenant). A set of many different policies (i.e., different SLAs) increases the amount of state required.
- * When a packet from the UE leaves the gNB, it is SR-routed. This simplifies network slicing [I-D.ietf-lsr-flex-algo].
- * In the uplink, the SRv6 BSID steers traffic into an SR policy when it arrives at the SRGW.

An example topology is shown in Figure 5.

S1 and C1 are two service segments. S1 represents a VNF in the network, and C1 represents a router configured for Traffic Engineering.

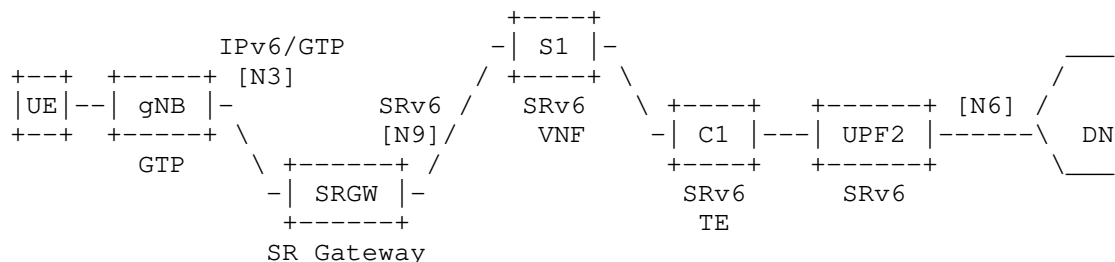


Figure 5: Enhanced mode with unchanged gNB IPv6/GTP behavior

5.3.1.1. Packet flow - Uplink

The uplink packet flow is as follows:

```

UE_out   : (A,Z)
gNB_out  : (gNB, B) (GTP: TEID T) (A,Z)      -> Interface N3 unmodified
                                                (IPv6/GTP)
SRGW_out : (SRGW, S1) (U2::T, C1; SL=2) (A,Z) -> B is an End.M.GTP6.D
                                                SID at the SRGW
S1_out   : (SRGW, C1) (U2::T, C1; SL=1) (A,Z)
C1_out   : (SRGW, U2::T) (A,Z)                -> End with PSP
UPF2_out : (A,Z)                             -> End.DT4 or End.DT6

```

The UE sends a packet destined to Z toward the gNB on a specific bearer for that session. The gNB, which is unmodified, encapsulates the packet into IPv6, UDP, and GTP headers. The IPv6 DA B, and the GTP TEID T are the ones received in the N2 interface.

The IPv6 address that was signaled over the N2 interface for that UE PDU Session, B, is now the IPv6 DA. B is an SRv6 Binding SID at the SRGW. Hence the packet is routed to the SRGW.

When the packet arrives at the SRGW, the SRGW identifies B as an End.M.GTP6.D Binding SID (see Section 6.3). Hence, the SRGW removes the IPv6, UDP, and GTP headers, and pushes an IPv6 header with its own SRH containing the SIDs bound to the SR policy associated with this BindingSID. There at least one instance of the End.M.GTP6.D SID per PDU type.

S1 and C1 perform their related Endpoint functionality and forward the packet.

When the packet arrives at UPF2, the active segment is (U2::T) which is bound to End.DT4/6. UPF2 then decapsulates (removing the outer IPv6 header with all its extension headers) and forwards the packet toward the data network.

5.3.1.2. Packet flow - Downlink

The downlink packet flow is as follows:

```

UPF2_in : (Z,A)                                -> UPF2 maps the flow with
                                                <C1, S1, SRGW::TEID,gNB>
UPF2_out: (U2::1, C1)(gNB, SRGW::TEID, S1; SL=3)(Z,A) -> H.Encaps.Red
C1_out   : (U2::1, S1)(gNB, SRGW::TEID, S1; SL=2)(Z,A)
S1_out   : (U2::1, SRGW::TEID)(gNB, SRGW::TEID, S1, SL=1)(Z,A)
SRGW_out : (SRGW, gNB)(GTP: TEID=T)(Z,A)      -> SRGW/96 is End.M.GTP6.E
gNB_out  : (Z,A)

```

When a packet destined to A arrives at the UPF2, the UPF2 performs a lookup in the table associated to A and finds the SID list <C1, S1, SRGW::TEID, gNB>. The UPF2 performs an H.Encaps.Red operation, encapsulating the packet into a new IPv6 header with its corresponding SRH.

C1 and S1 perform their related Endpoint processing.

Once the packet arrives at the SRGW, the SRGW identifies the active SID as an End.M.GTP6.E function. The SRGW removes the IPv6 header and all its extensions headers. The SRGW generates new IPv6, UDP, and GTP headers. The new IPv6 DA is the gNB which is the last SID in the received SRH. The TEID in the generated GTP header is an argument of the received End.M.GTP6.E SID. The SRGW pushes the headers to the packet and forwards the packet toward the gNB. There is one instance of the End.M.GTP6.E SID per PDU type.

Once the packet arrives at the gNB, the packet is a regular IPv6/GTP packet. The gNB looks for the specific radio bearer for that TEID and forward it on the bearer. This gNB behavior is not modified from current and previous generations.

5.3.1.3. Scalability

For the downlink traffic, the SRGW is stateless. All the state is in the SRH pushed by the UPF2. The UPF2 must have the UE states since it is the UE's session anchor point.

For the uplink traffic, the state at the SRGW does not necessarily need to be unique per PDU Session; the SR policy can be shared among UEs. This enables more scalable SRGW deployments compared to a solution holding millions of states, one or more per UE.

5.3.2. Interworking with IPv4 GTP

In this interworking mode the gNB uses GTP over IPv4 in the N3 interface

Key points:

- * The gNB is unchanged and encapsulates packets into GTP (the N3 interface is not modified).
- * N2 signaling is not changed, though multiple UPF addresses need to be provided – one for each PDU Session Type.
- * In the uplink, traffic is classified by SRGW's classification engine and steered into an SR policy. The SRGW may be implemented in a UPF or as a separate entity. How the classification engine rules are set up is outside the scope of this document, though one example is using BGP signaling from a Mobile User Plane Controller [I-D.mhkk-dmm-srv6mup-architecture].
- * SRGW removes GTP, finds the SID list related to DA, and adds an SRH with the SID list.

An example topology is shown in Figure 6. In this mode the gNB is an unmodified gNB using IPv4/GTP. The UPFs are SR-aware. As before, the SRGW maps the IPv4/GTP traffic to SRv6.

S1 and C1 are two service segment endpoints. S1 represents a VNF in the network, and C1 represents a router configured for Traffic Engineering.

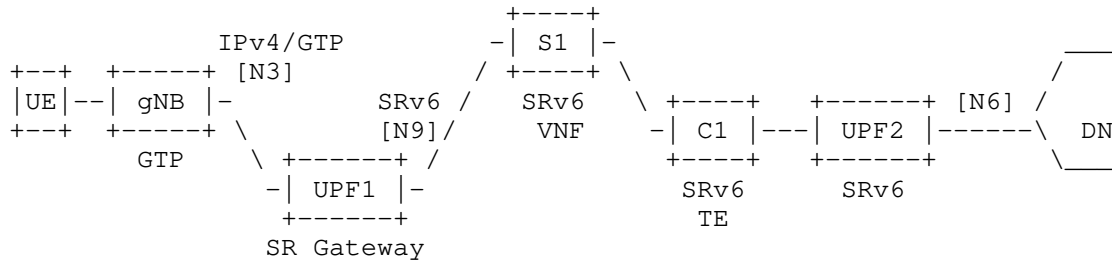


Figure 6: Enhanced mode with unchanged gNB IPv4/GTP behavior

5.3.2.1. Packet flow - Uplink

The uplink packet flow is as follows:

```

gNB_out : (gNB, B) (GTP: TEID T) (A, Z)      -> Interface N3
                                              unchanged IPv4/GTP
SRGW_out: (SRGW, S1) (U2::1, C1; SL=2) (A, Z) -> H.M.GTP4.D function
S1_out  : (SRGW, C1) (U2::1, C1; SL=1) (A, Z)
C1_out  : (SRGW, U2::1) (A, Z)                -> PSP
UPF2_out: (A, Z)                             -> End.DT4 or End.DT6
  
```

The UE sends a packet destined to Z toward the gNB on a specific bearer for that session. The gNB, which is unmodified, encapsulates the packet into a new IPv4, UDP, and GTP headers. The IPv4 DA, B, and the GTP TEID are the ones received at the N2 interface.

When the packet arrives at the SRGW for UPF1, the SRGW has an classification engine rule for incoming traffic from the gNB, that steers the traffic into an SR policy by using the function H.M.GTP4.D. The SRGW removes the IPv4, UDP, and GTP headers and pushes an IPv6 header with its own SRH containing the SIDs related to the SR policy associated with this traffic. The SRGW forwards according to the new IPv6 DA.

S1 and C1 perform their related Endpoint functionality and forward the packet.

When the packet arrives at UPF2, the active segment is (U2::1) which is bound to End.DT4/6 which performs the decapsulation (removing the outer IPv6 header with all its extension headers) and forwards toward the data network.

Note that the interworking mechanisms for IPv4/GTP and IPv6/GTP differs. This is due to the fact that in IPv6/GTP we can leverage the remote steering capabilities provided by the Segment Routing BSID. In IPv4 this construct is not available, and building a similar mechanism would require a significant address consumption.

5.3.2.2. Packet flow - Downlink

The downlink packet flow is as follows:

```

UPF2_in : (Z,A)                                -> UPF2 maps flow with SID
                                                <C1, S1,GW::SA:DA:TEID>
UPF2_out: (U2::1, C1) (GW::SA:DA:TEID, S1; SL=2) (Z,A) ->H.Encaps.Red
C1_out   : (U2::1, S1) (GW::SA:DA:TEID, S1; SL=1) (Z,A)
S1_out   : (U2::1, GW::SA:DA:TEID) (Z,A)
SRGW_out: (GW, gNB) (GTP: TEID=T) (Z,A)         -> End.M.GTP4.E
gNB_out  : (Z,A)

```

When a packet destined to A arrives at the UPF2, the UPF2 performs a lookup in the table associated to A and finds the SID list <C1, S1, SRGW::SA:DA:TEID>. The UPF2 performs a H.Encaps.Red operation, encapsulating the packet into a new IPv6 header with its corresponding SRH.

The nodes C1 and S1 perform their related Endpoint processing.

Once the packet arrives at the SRGW, the SRGW identifies the active SID as an End.M.GTP4.E function. The SRGW removes the IPv6 header and all its extensions headers. The SRGW generates an IPv4, UDP, and GTP headers. The IPv4 SA and DA are received as SID arguments. The TEID in the generated GTP header is also the arguments of the received End.M.GTP4.E SID. The SRGW pushes the headers to the packet and forwards the packet toward the gNB.

When the packet arrives at the gNB, the packet is a regular IPv4/GTP packet. The gNB looks for the specific radio bearer for that TEID and forwards it on the bearer. This gNB behavior is not modified from current and previous generations.

5.3.2.3. Scalability

For the downlink traffic, the SRGW is stateless. All the state is in the SRH pushed by the UPF2. The UPF must have this UE-base state anyway (since it is its anchor point).

For the uplink traffic, the state at the SRGW is dedicated on a per UE/session basis according to a classification engine. There is state for steering the different sessions in the form of an SR Policy. However, SR policies are shared among several UE/sessions.

5.3.3. Extensions to the interworking mechanisms

In this section we presented two mechanisms for interworking with gNBs and UPFs that do not support SRv6. These mechanisms are used to support GTP over IPv4 and IPv6.

Even though we have presented these methods as an extension to the "Enhanced mode", it is straightforward in its applicability to the "Traditional mode".

5.4. SRv6 Drop-in Interworking

In this section we introduce another mode useful for legacy gNB and UPFs that still operate with GTP-U. This mode provides an SRv6-enabled user plane in between two GTP-U tunnel endpoints.

In this mode we employ two SRGWs that map GTP-U traffic to SRv6 and vice-versa.

Unlike other interworking modes, in this mode both of the mobility overlay endpoints use GTP-U. Two SRGWs are deployed in either N3 or N9 interface to realize an intermediate SR policy.

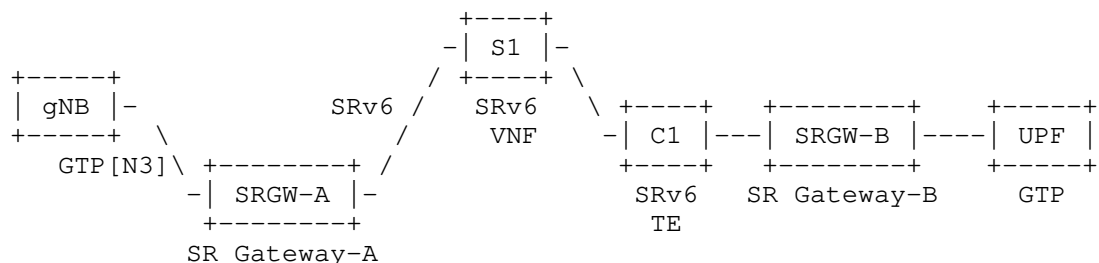


Figure 7: Example topology for SRv6 Drop-in mode

The packet flow of Figure 7 is as follows:

```

gNB_out : (gNB, U::1) (GTP: TEID T) (A,Z)
GW-A_out: (GW-A, S1) (U::1, SGB::TEID, C1; SL=3) (A,Z) ->U::1 is an
                                                    End.M.GTP6.D.Di
                                                    SID at SRGW-A
S1_out   : (GW-A, C1) (U::1, SGB::TEID, C1; SL=2) (A,Z)
C1_out   : (GW-A, SGB::TEID) (U::1, SGB::TEID, C1; SL=1) (A,Z)
GW-B_out: (GW-B, U::1) (GTP: TEID T) (A,Z) ->SGB::TEID is an
                                                    End.M.GTP6.E
                                                    SID at SRGW-B

UPF_out  : (A,Z)

```

When a packet destined to Z is sent to the gNB, which is unmodified (control-plane and user-plane remain GTP-U), gNB performs encapsulation into a new IP, UDP, and GTP headers. The IPv6 DA, U::1, and the GTP TEID are the ones received at the N2 interface.

The IPv6 address that was signaled over the N2 interface for that PDU Session, U::1, is now the IPv6 DA. U::1 is an SRv6 Binding SID at SRGW-A. Hence the packet is routed to the SRGW.

When the packet arrives at SRGW-A, the SRGW identifies U::1 as an End.M.GTP6.D.Di Binding SID (see Section 6.4). Hence, the SRGW removes the IPv6, UDP, and GTP headers, and pushes an IPv6 header with its own SRH containing the SIDs bound to the SR policy associated with this Binding SID. There is one instance of the End.M.GTP6.D.Di SID per PDU type.

S1 and C1 perform their related Endpoint functionality and forward the packet.

Once the packet arrives at SRGW-B, the SRGW identifies the active SID as an End.M.GTP6.E function. The SRGW removes the IPv6 header and all its extensions headers. The SRGW generates new IPv6, UDP, and GTP headers. The new IPv6 DA is U::1 which is the last SID in the received SRH. The TEID in the generated GTP header is an argument of the received End.M.GTP6.E SID. The SRGW pushes the headers to the packet and forwards the packet toward UPF. There is one instance of the End.M.GTP6.E SID per PDU type.

Once the packet arrives at UPF, the packet is a regular IPv6/GTP packet. The UPF looks for the specific rule for that TEID to forward the packet. This UPF behavior is not modified from current and previous generations.

6. SRv6 Segment Endpoint Mobility Behaviors

6.1. Args.Mob.Session

Args.Mob.Session provide per-session information for charging, buffering and lawful intercept (among others) required by some mobile nodes. The Args.Mob.Session argument format is used in combination with End.Map, End.DT4/End.DT6/End.DT46 and End.DX4/End.DX6/End.DX2 behaviors. Note that proposed format is applicable for 5G networks, while similar formats could be used for legacy networks.

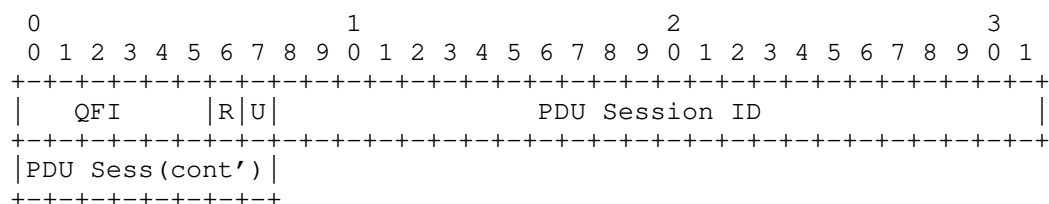


Figure 8: Args.Mob.Session format

- * QFI: QoS Flow Identifier [TS.38415]
- * R: Reflective QoS Indication [TS.23501]. This parameter indicates the activation of reflective QoS towards the UE for the transferred packet. Reflective QoS enables the UE to map UL User Plane traffic to QoS Flows without SMF provided QoS rules.
- * U: Unused and for future use. MUST be 0 on transmission and ignored on receipt.
- * PDU Session ID: Identifier of PDU Session. The GTP-U equivalent is TEID.

Arg.Mob.Session is required in case that one SID aggregates multiple PDU Sessions. Since the SRv6 SID is likely NOT to be instantiated per PDU session, Args.Mob.Session helps the UPF to perform the behaviors which require per QFI and/or per PDU Session granularity.

Note that the encoding of user-plane messages (e.g., Echo Request, Echo Reply, Error Indication and End Marker) is out of the scope of this draft. [I-D.murakami-dmm-user-plane-message-encoding] defines one possible encoding.

6.2. End.MAP

The "Endpoint behavior with SID mapping" behavior (End.MAP for short) is used in several scenarios. Particularly in mobility, End.MAP is used by the intermediate UPFs.

When node N receives a packet whose IPv6 DA is D and D is a local End.MAP SID, N does:


```
S01. If (IPv6 Hop Limit <= 1) {
S02.   Send an ICMP Time Exceeded message to the Source Address,
      Code 0 (Hop limit exceeded in transit),
      interrupt packet processing, and discard the packet.
S03. }
S04. Decrement IPv6 Hop Limit by 1
S05. Update the IPv6 DA with the new mapped SID
S06. Submit the packet to the egress IPv6 FIB lookup for
      transmission to the new destination
```

Notes: The SIDs in the SRH are not modified.

6.3. End.M.GTP6.D

The "Endpoint behavior with IPv6/GTP decapsulation into SR policy" behavior (End.M.GTP6.D for short) is used in interworking scenario for the uplink towards SRGW from the legacy gNB using IPv6/GTP. Any SID instance of this behavior is associated with an SR Policy B and an IPv6 Source Address S.

When the SR Gateway node N receives a packet destined to D and D is a local End.M.GTP6.D SID, N does:

```
S01. When an SRH is processed {
S02.   If (Segments Left != 0) {
S03.     Send an ICMP Parameter Problem to the Source Address,
      Code 0 (Erroneous header field encountered),
      Pointer set to the Segments Left field,
      interrupt packet processing, and discard the packet.
S04.   }
S05.   Proceed to process the next header in the packet
S06. }
```

When processing the Upper-layer header of a packet matching a FIB entry locally instantiated as an End.M.GTP6.D SID, N does:

```
S01. If (Next Header (NH) == UDP & UDP_Dest_port == GTP) {
S02.   Copy the GTP TEID and QFI to buffer memory
S03.   Pop the IPv6, UDP, and GTP Headers
S04.   Push a new IPv6 header with its own SRH containing B
S05.   Set the outer IPv6 SA to S
S06.   Set the outer IPv6 DA to the first SID of B
S07.   Set the outer Payload Length, Traffic Class, Flow Label,
       Hop Limit, and Next-Header (NH) fields
S08.   Write in the SRH[0] the Args.Mob.Session based on
       the information of buffer memory
S09.   Submit the packet to the egress IPv6 FIB lookup and
       transmission to the new destination
S10. } Else {
S11.   Process as per [RFC8986] Section 4.1.1
S12. }
```

Notes: S07. The NH is set based on the SID parameter. There is one instantiation of the End.M.GTP6.D SID per PDU Session Type, hence the NH is already known in advance. For the IPv4v6 PDU Session Type, in addition we inspect the first nibble of the PDU to know the NH value.

The last segment (S3 in above example) SHOULD be followed by an Arg.Mob.Session argument space which is used to provide the session identifiers.

6.4. End.M.GTP6.D.Di

The "Endpoint behavior with IPv6/GTP decapsulation into SR policy for Drop-in Mode" behavior (End.M.GTP6.D.Di for short) is used in SRv6 drop-in interworking scenario described in Section 5.4. The difference between End.M.GTP6.D as another variant of IPv6/GTP decapsulation function is that the original IPv6 DA of GTP packet is preserved as the last SID in SRH.

Any SID instance of this behavior is associated with an SR Policy B and an IPv6 Source Address S.

When the SR Gateway node N receives a packet destined to D and D is a local End.M.GTP6.D.Di SID, N does:

```
S01. When an SRH is processed {
S02.   If (Segments Left != 0) {
S03.     Send an ICMP Parameter Problem to the Source Address,
        Code 0 (Erroneous header field encountered),
        Pointer set to the Segments Left field,
        interrupt packet processing, and discard the packet.
S04.   }
S05.   Proceed to process the next header in the packet
S06. }
```

When processing the Upper-layer header of a packet matching a FIB entry locally instantiated as an End.M.GTP6.Di SID, N does:

```
S01. If (Next Header = UDP & UDP_Dest_port = GTP) {
S02.   Copy D to buffer memory
S03.   Pop the IPv6, UDP, and GTP Headers
S04.   Push a new IPv6 header with its own SRH containing B
S05.   Set the outer IPv6 SA to S
S06.   Set the outer IPv6 DA to the first SID of B
S07.   Set the outer Payload Length, Traffic Class, Flow Label,
        Hop Limit, and Next-Header fields
S08.   Prepend D to the SRH (as SRH[0]) and set SL accordingly
S09.   Submit the packet to the egress IPv6 FIB lookup and
        transmission to the new destination
S10. } Else {
S11.   Process as per [RFC8986] Section 4.1.1
S12. }
```

Notes: S07. The NH is set based on the SID parameter. There is one instantiation of the End.M.GTP6.D SID per PDU Session Type, hence the NH is already known in advance. For the IPv4v6 PDU Session Type, in addition we inspect the first nibble of the PDU to know the NH value.

S SHOULD be an End.M.GTP6.E SID instantiated at the SR gateway.

6.5. End.M.GTP6.E

The "Endpoint behavior with encapsulation for IPv6/GTP tunnel" behavior (End.M.GTP6.E for short) is used among others in the interworking scenario for the downlink toward the legacy gNB using IPv6/GTP.

The prefix of End.M.GTP6.E SID MUST be followed by the Arg.Mob.Session argument space which is used to provide the session identifiers.

When the SR Gateway node N receives a packet destined to D, and D is a local End.M.GTP6.E SID, N does the following:

```
S01. When an SRH is processed {
S02.   If (Segments Left != 1) {
S03.     Send an ICMP Parameter Problem to the Source Address,
        Code 0 (Erroneous header field encountered),
        Pointer set to the Segments Left field,
        interrupt packet processing, and discard the packet.
S04.   }
S05.   Proceed to process the next header in the packet
S06. }
```

When processing the Upper-layer header of a packet matching a FIB entry locally instantiated as an End.M.GTP6.E SID, N does:

```
S01.   Copy SRH[0] and D to buffer memory
S02.   Pop the IPv6 header and all its extension headers
S03.   Push a new IPv6 header with a UDP/GTP Header
S04.   Set the outer IPv6 SA to S
S05.   Set the outer IPv6 DA from buffer memory
S06.   Set the outer Payload Length, Traffic Class, Flow Label,
        Hop Limit, and Next-Header fields
S07.   Set the GTP TEID (from buffer memory)
S08.   Submit the packet to the egress IPv6 FIB lookup and
        transmission to the new destination
```

Notes: An End.M.GTP6.E SID MUST always be the penultimate SID. The TEID is extracted from the argument space of the current SID.

The source address S SHOULD be an End.M.GTP6.D SID instantiated at an SR gateway.

6.6. End.M.GTP4.E

The "Endpoint behavior with encapsulation for IPv4/GTP tunnel" behavior (End.M.GTP4.E for short) is used in the downlink when doing interworking with legacy gNB using IPv4/GTP.

When the SR Gateway node N receives a packet destined to S and S is a local End.M.GTP4.E SID, N does:

```
S01. When an SRH is processed {
S02.   If (Segments Left != 0) {
S03.     Send an ICMP Parameter Problem to the Source Address,
        Code 0 (Erroneous header field encountered),
        Pointer set to the Segments Left field,
        interrupt packet processing, and discard the packet.
S04.   }
S05.   Proceed to process the next header in the packet
S06. }
```

When processing the Upper-layer header of a packet matching a FIB entry locally instantiated as an End.M.GTP4.E SID, N does:

- S01. Store the IPv6 DA and SA in buffer memory
- S02. Pop the IPv6 header and all its extension headers
- S03. Push a new IPv4 header with a UDP/GTP Header
- S04. Set the outer IPv4 SA and DA (from buffer memory)
- S05. Set the outer Total Length, DSCP, Time To Live, and Next-Header fields
- S06. Set the GTP TEID (from buffer memory)
- S07. Submit the packet to the egress IPv6 FIB lookup and transmission to the new destination

Notes: The End.M.GTP4.E SID in S has the following format:

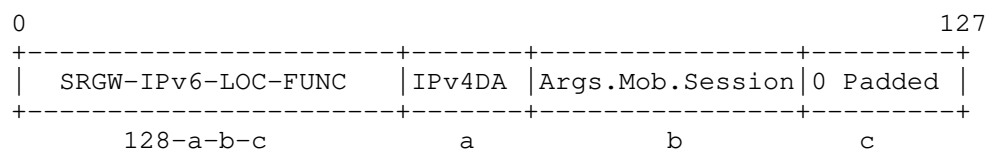


Figure 9: End.M.GTP4.E SID Encoding

The IPv6 Source Address has the following format:

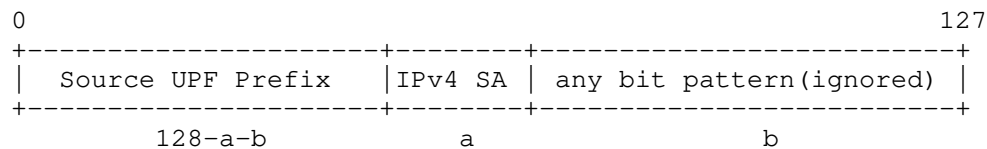


Figure 10: IPv6 SA Encoding for End.M.GTP4.E

6.7. H.M.GTP4.D

The "SR Policy Headend with tunnel decapsulation and map to an SRv6 policy" behavior (H.M.GTP4.D for short) is used in the direction from legacy IPv4 user-plane to SRv6 user-plane network.

When the SR Gateway node N receives a packet destined to a IW-IPv4-Prefix, N does:

```

S01. IF Payload == UDP/GTP THEN
S02.   Pop the outer IPv4 header and UDP/GTP headers
S03.   Copy IPv4 DA, TEID to form SID B
S04.   Copy IPv4 SA to form IPv6 SA B'
S05.   Encapsulate the packet into a new IPv6 header   ;;Ref1
S06.   Set the IPv6 DA = B
S07.   Forward along the shortest path to B
S08. ELSE
S09.   Drop the packet

```

Ref1: The NH value is identified by inspecting the first nibble of the inner payload.

The SID B has the following format:

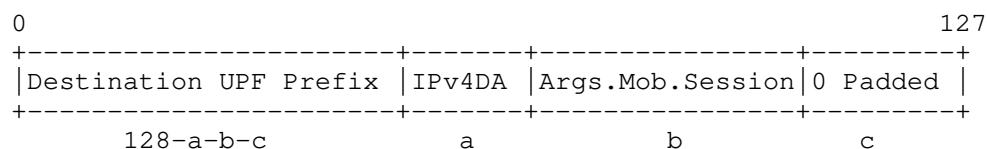


Figure 11: H.M.GTP4.D SID Encoding

The SID B MAY be an SRv6 Binding SID instantiated at the first UPF (U1) to bind an SR policy [I-D.ietf-spring-segment-routing-policy].

6.8. End.Limit: Rate Limiting behavior

The mobile user-plane requires a rate-limit feature. For this purpose, we define a new behavior "End.Limit". The "End.Limit" behavior encodes in its arguments the rate limiting parameter that should be applied to this packet. Multiple flows of packets should have the same group identifier in the SID when those flows are in the same AMBR (Aggregate Maximum Bit Rate) group. The encoding format of the rate limit segment SID is as follows:

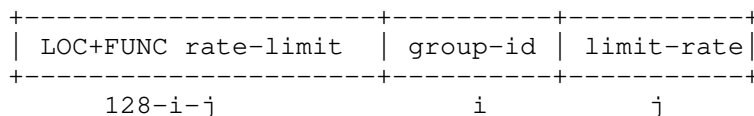


Figure 12: End.Limit: Rate limiting behavior argument format

If the limit-rate bits are set to zero, the node should not do rate limiting unless static configuration or control-plane sets the limit rate associated to the SID.

7. SRv6 supported 3GPP PDU session types

The 3GPP [TS.23501] defines the following PDU session types:

- * IPv4
- * IPv6
- * IPv4v6
- * Ethernet
- * Unstructured

SRv6 supports the 3GPP PDU session types without any protocol overhead by using the corresponding SRv6 behaviors (End.DX4, End.DT4 for IPv4 PDU sessions; End.DX6, End.DT6, End.T for IPv6 PDU sessions; End.DT46 for IPv4v6 PDU sessions; End.DX2 for L2 and Unstructured PDU sessions).

8. Network Slicing Considerations

A mobile network may be required to implement "network slices", which logically separate network resources. User-plane behaviors represented as SRv6 segments would be part of a slice.

[I-D.ietf-spring-segment-routing-policy] describes a solution to build basic network slices with SR. Depending on the requirements, these slices can be further refined by adopting the mechanisms from:

- * IGP Flex-Algo [I-D.ietf-lsr-flex-algo]
- * Inter-Domain policies
[I-D.ietf-spring-segment-routing-central-epe]

Furthermore, these can be combined with ODN/AS (On Demand Nexthop/ Automated Steering) [I-D.ietf-spring-segment-routing-policy] for automated slice provisioning and traffic steering.

Further details on how these tools can be used to create end to end network slices are documented in [I-D.ali-spring-network-slicing-building-blocks].

9. Control Plane Considerations

This document focuses on user-plane behavior and its independence from the control plane. While the SRv6 mobile user-plane behaviors may be utilized in emerging architectures, such as [I-D.gundavelli-dmm-mfa], [I-D.mhkk-dmm-srv6mup-architecture] for example, require control plane support for the user-plane, this document does not impose any change to the existent mobility control plane.

Section 11 allocates SRv6 Segment Endpoint Behavior codepoints for the new behaviors defined in this document.

10. Security Considerations

The security considerations for Segment Routing are discussed in [RFC8402]. More specifically for SRv6 the security considerations and the mechanisms for securing an SR domain are discussed in [RFC8754]. Together, they describe the required security mechanisms that allow establishment of an SR domain of trust to operate SRv6-based services for internal traffic while preventing any external traffic from accessing or exploiting the SRv6-based services.

The technology described in this document is applied to a mobile network that is within the SR Domain.

This document introduces new SRv6 Endpoint Behaviors. Those behaviors do not need any special security consideration given that it is deployed within that SR Domain.

11. IANA Considerations

The following values have been allocated within the "SRv6 Endpoint Behaviors" [RFC8986] sub-registry belonging to the top-level "Segment Routing Parameters" registry:

Value	Hex	Endpoint behavior	Reference
40	0x0028	End.MAP	[This.ID]
41	0x0029	End.Limit	[This.ID]
69	0x0045	End.M.GTP6.D	[This.ID]
70	0x0046	End.M.GTP6.Di	[This.ID]
71	0x0047	End.M.GTP6.E	[This.ID]
72	0x0048	End.M.GTP4.E	[This.ID]

Table 1: SRv6 Mobile User-plane Endpoint Behavior Types

12. Acknowledgements

The authors would like to thank Daisuke Yokota, Bart Peirens, Ryokichi Onishi, Kentaro Ebisawa, Peter Bosch, Darren Dukes, Francois Clad, Sri Gundavelli, Sridhar Bhaskaran, Arashmid Akhavain, Ravi Shekhar, Aeneas Dodd-Noble, Carlos Jesus Bernardos, Dirk v. Hugo and Jeffrey Zhang for their useful comments of this work.

13. Contributors

Kentaro Ebisawa Toyota Motor Corporation Japan

Email: ebisawa@toyota-tokyo.tech

Tetsuya Murakami Arrcus, Inc. United States of America

Email: tetsuya.ietf@gmail.com

14. References

14.1. Normative References

- [I-D.ietf-spring-segment-routing-policy]
Filsfils, C., Talaulikar, K., Voyer, D., Bogdanov, A., and P. Mattes, "Segment Routing Policy Architecture", Work in Progress, Internet-Draft, draft-ietf-spring-segment-routing-policy-22, 22 March 2022, <<https://datatracker.ietf.org/doc/html/draft-ietf-spring-segment-routing-policy-22>>.
- [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>>.
- [RFC8402] Filsfils, C., Ed., Previdi, S., Ed., Ginsberg, L., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", RFC 8402, DOI 10.17487/RFC8402, July 2018, <<https://www.rfc-editor.org/info/rfc8402>>.
- [RFC8754] Filsfils, C., Ed., Dukes, D., Ed., Previdi, S., Leddy, J., Matsushima, S., and D. Voyer, "IPv6 Segment Routing Header (SRH)", RFC 8754, DOI 10.17487/RFC8754, March 2020, <<https://www.rfc-editor.org/info/rfc8754>>.

- [RFC8986] Filsfils, C., Ed., Camarillo, P., Ed., Leddy, J., Voyer, D., Matsushima, S., and Z. Li, "Segment Routing over IPv6 (SRv6) Network Programming", RFC 8986, DOI 10.17487/RFC8986, February 2021, <<https://www.rfc-editor.org/info/rfc8986>>.
- [TS.23501] 3GPP, "System Architecture for the 5G System", 3GPP TS 23.501 15.0.0, November 2017.

14.2. Informative References

- [I-D.ali-spring-network-slicing-building-blocks]
Ali, Z., Filsfils, C., Camarillo, P., and D. Voyer, "Building blocks for Slicing in Segment Routing Network", Work in Progress, Internet-Draft, draft-ali-spring-network-slicing-building-blocks-04, 21 February 2021, <<https://datatracker.ietf.org/doc/html/draft-ali-spring-network-slicing-building-blocks-04>>.
- [I-D.camarilloelmalky-springdmm-srv6-mob-usecases]
Garvia, P. C., Filsfils, C., Elmalky, H., Matsushima, S., Voyer, D., Cui, A., and B. Peirens, "SRv6 Mobility Use-Cases", Work in Progress, Internet-Draft, draft-camarilloelmalky-springdmm-srv6-mob-usecases-02, 15 August 2019, <<https://datatracker.ietf.org/doc/html/draft-camarilloelmalky-springdmm-srv6-mob-usecases-02>>.
- [I-D.gundavelli-dmm-mfa]
Gundavelli, S., Liebsch, M., and S. Matsushima, "Mobility-aware Floating Anchor (MFA)", Work in Progress, Internet-Draft, draft-gundavelli-dmm-mfa-01, 19 September 2018, <<https://datatracker.ietf.org/doc/html/draft-gundavelli-dmm-mfa-01>>.
- [I-D.ietf-lsr-flex-algo]
Psenak, P., Hegde, S., Filsfils, C., Talaulikar, K., and A. Gulko, "IGP Flexible Algorithm", Work in Progress, Internet-Draft, draft-ietf-lsr-flex-algo-19, 7 April 2022, <<https://datatracker.ietf.org/doc/html/draft-ietf-lsr-flex-algo-19>>.
- [I-D.ietf-spring-segment-routing-central-epe]
Filsfils, C., Previdi, S., Dawra, G., Aries, E., and D. Afanasiev, "Segment Routing Centralized BGP Egress Peer Engineering", Work in Progress, Internet-Draft, draft-ietf-spring-segment-routing-central-epe-10, 21 December 2017, <<https://datatracker.ietf.org/doc/html/draft-ietf-spring-segment-routing-central-epe-10>>.

- [I-D.ietf-spring-sr-service-programming]
Clad, F., Xu, X., Filsfils, C., Bernier, D., Li, C.,
Decraene, B., Ma, S., Yadlapalli, C., Henderickx, W., and
S. Salsano, "Service Programming with Segment Routing",
Work in Progress, Internet-Draft, draft-ietf-spring-sr-
service-programming-05, 10 September 2021,
<<https://datatracker.ietf.org/doc/html/draft-ietf-spring-sr-service-programming-05>>.
- [I-D.kohno-dmm-srv6mob-arch]
Kohno, M., Clad, F., Camarillo, P., and Z. Ali,
"Architecture Discussion on SRv6 Mobile User plane", Work
in Progress, Internet-Draft, draft-kohno-dmm-srv6mob-arch-
05, 8 November 2021,
<<https://datatracker.ietf.org/doc/html/draft-kohno-dmm-srv6mob-arch-05>>.
- [I-D.matsushima-spring-srv6-deployment-status]
Matsushima, S., Filsfils, C., Ali, Z., Li, Z., Rajaraman,
K., and A. Dhamija, "SRv6 Implementation and Deployment
Status", Work in Progress, Internet-Draft, draft-
matsushima-spring-srv6-deployment-status-15, 5 April 2022,
<<https://datatracker.ietf.org/doc/html/draft-matsushima-spring-srv6-deployment-status-15>>.
- [I-D.mhkk-dmm-srv6mup-architecture]
Matsushima, S., Horiba, K., Khan, A., Kawakami, Y.,
Murakami, T., Patel, K., Kohno, M., Kamata, T., Garvia, P.
C., Voyer, D., Zadok, S., Meilik, I., Agrawal, A.,
Perumal, K., and J. Horn, "Segment Routing IPv6 Mobile
User Plane Architecture for Distributed Mobility
Management", Work in Progress, Internet-Draft, draft-mhkk-
dmm-srv6mup-architecture-03, 20 March 2022,
<<https://datatracker.ietf.org/doc/html/draft-mhkk-dmm-srv6mup-architecture-03>>.
- [I-D.murakami-dmm-user-plane-message-encoding]
Murakami, T., Matsushima, S., Ebisawa, K., Camarillo, P.,
and R. Shekhar, "User Plane Message Encoding", Work in
Progress, Internet-Draft, draft-murakami-dmm-user-plane-
message-encoding-05, 5 March 2022,
<<https://datatracker.ietf.org/doc/html/draft-murakami-dmm-user-plane-message-encoding-05>>.
- [TS.29281] 3GPP, "General Packet Radio System (GPRS) Tunnelling
Protocol User Plane (GTPv1-U)", 3GPP TS 29.281 15.1.0,
December 2017.

[TS.38415] 3GPP, "Draft Specification for 5GS container (TS 38.415)",
3GPP R3-174510 0.0.0, August 2017.

Appendix A. Implementations

This document introduces new SRv6 Endpoint Behaviors. These behaviors have an open-source P4 implementation available in <https://github.com/ebiken/p4srv6>.

Additionally, a full implementation of this document is available in Linux Foundation FD.io VPP project since release 20.05. More information available here: https://docs.fd.io/vpp/20.05/d7/d3c/srv6_mobile_plugin_doc.html.

There are also experimental implementations in M-CORD NGIC and Open Air Interface (OAI).

Authors' Addresses

Satoru Matsushima (editor)
SoftBank
Japan
Email: satoru.matsushima@g.softbank.co.jp

Clarence Filsfils
Cisco Systems, Inc.
Belgium
Email: cf@cisco.com

Miya Kohno
Cisco Systems, Inc.
Japan
Email: mkohno@cisco.com

Pablo Camarillo Garvia (editor)
Cisco Systems, Inc.
Spain
Email: pcamaril@cisco.com

Daniel Voyer
Bell Canada
Canada
Email: daniel.voyer@bell.ca

Charles E. Perkins
Lupin Lodge
20600 Aldercroft Heights Rd.
Los Gatos, CA 95033
United States of America
Email: charliep@computer.org

DMM Working Group
Internet-Draft
Intended status: Informational
Expires: May 4, 2020

M. Kohno
F. Clad
P. Camarillo
Z. Ali
Cisco Systems, Inc.
November 1, 2019

Architecture Discussion on SRv6 Mobile User plane
draft-kohno-dmm-srv6mob-arch-00

Abstract

Layer separation is a powerful concept in system architecture. In the area of mobility, by separating GTP-U that is the overlay tunnel, and the IP transport network that is the underlay, the operation of the mobile network and the transport network can be separated, allowing them to evolve independently.

However, evolving individually at each layer promotes local optimization and may result in non-optimal solutions overall in the long run.

When a drastic architectural transition is required, for example, in the 5G era where various SLAs and completely new data intensive services are assumed, it is necessary to reconsider the architecture holistically, rather than from the viewpoint of individual layer.

One important value propositions of SRv6 mobile user plane is the possible optimization across the existing multiple layers.

This document discusses the architectural implications of applying SRv6 mobile user plane, especially regarding the possible optimization of existing layers. Then it takes 5G requirements and use cases as an example, and describes how these use cases are simply and effectively realized with the inter-layer optimization capability of SRv6 mobile user plane. Thus it show that SRv6 mobile use plane is a right architectural choice for the 5G era.

Status of This Memo

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

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

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

This Internet-Draft will expire on May 4, 2020.

Copyright Notice

Copyright (c) 2019 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 (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Architecture Consideration and Necessity of Inter-layer Optimization	4
3. Terminology	5
4. SRv6 mobile user plane and the 5G use cases	5
4.1. Network Slicing	5
4.2. Edge Computing	6
4.3. URLLC (Ultra-Reliable Low-Latency Communication) support	7
5. Security Considerations	8
6. IANA Considerations	8
7. Acknowledgements	8
8. References	8
8.1. Normative References	8
8.2. Informative References	9
Authors' Addresses	10

1. Introduction

Layer separation is a powerful concept in system architecture. In the area of mobility, by separating GTP-U that is the overlay tunnel, and the IP transport network that is the underlay, the operation of the mobile network and the transport network can be separated, allowing them to evolve independently.

However, evolving individually at each layer promotes local optimization and may result in non-optimal solutions overall in the long run.

The well-known aphorism of David J.Wheeler says:

"All problems in computer science can be solved by adding another level of indirection."

But, as a corollary, it also says:

"...that usually will create another problem."

In other words, excessive layer separation is not good for an overall architecture.

Existing practices have reasonable grounds, so it is usually recommended to follow them. But when a drastic architectural transition is required, for example, in the 5G era where various SLAs and completely new data intensive services are assumed, it is necessary to reconsider the architecture holistically, rather than from the viewpoint of individual layer.

SRv6 mobile user plane has been proposed as an alternative way to complement or replace GTP-U both in IETF [I-D.ietf-dmm-srv6-mobile-uplane] and 3GPP [TR.29892].

SRv6 has also an advantage if it is used as a mobile user plane, because of its flexibility through Service Programming functions and the use of metadata, in addition to the simple and stateless traffic steering capability.

The 3GPP data plane entities such as UPFs and service functions can be implemented either as virtual or physical appliances. The fact that SRv6 has been supported on various platforms including custom ASICs, commercially available NPUs, programmable switches, Smart NICs, Linux Kernel, virtual forwarders on server and container networking, will make the deployment flexible.

Also, the declarative programming nature of SRv6 will provide the necessary distinction to clarify basic reachability vs constraint path vs service path, whereas existing practices depended on the layer separation - service overlay and underlay. In other words, one of the most important value propositions of SRv6 mobile user plane is the possibility to perform cross-layer optimizations.

This document discusses the architectural implications of applying SRv6 mobile user plane, especially regarding the possible

optimization of existing layers. Then it takes 5G requirements and use cases as an example, and describes how these use cases are simply and effectively realized with the inter-layer optimization capability of SRv6 mobile user plane. Thus it show that SRv6 mobile use plane is a right architectural choice for the 5G era.

2. Architecture Consideration and Necessity of Inter-layer Optimization

Historically, Mobile and Transport Network have been designed, standardized and operated separately. GTP-U has been defined as the mobile user plane. This is an overlay tunnel that runs over the Transport Network. Therefore, the underlying network cannot be directly controlled.

5G requires variety of tight-SLA characteristics and flexible traffic steering towards various service functions. While 3GPP has been focused on the mobility overlay, how to map the overlay requirements into the transport network is out of the scope.

IETF has discussed mobile end-to-end slicing in different WGs [I-D.rocky-5g-transport-slice], [I-D.clt-dmm-tn-aware-mobility]. However, all these proposals are based on the current assumption that the underlying network is separated from the overlay and agnostic.

The evolution of architecture requires a review of conventional domain boundaries and practices. This way, inefficiencies caused by traditional practices can be reduced. For example, now that "CUPS" separated the Control Plane and User Plane, UPF, which is dedicated to forwarding, can be considered as an entity on the IP Transport Network.

And, as a matter of fact, layer reduction for efficiency has been done in other domains. Some data centers adopted native IP CLOS, avoiding using VXLAN for simplicity. Also, broadband subscriber managements were simplified by using IPoE instead of PPPoE / L2TP.

In the context of mobile operators, SRv6 provides end-to-end simpler network operations thus decreasing the OPEX. SRv6 has a potential to perform inter-layer optimizations and/or eliminate overlay tunnels, though it does not mandate to do so.

Note that SRv6 can also be applied to the mobility overlay, in which case it also has benefits as the tunnels are removed.

3. Terminology

The terminology used in this document leverages and conforms to [I-D.ietf-dmm-srv6-mobile-uplane].

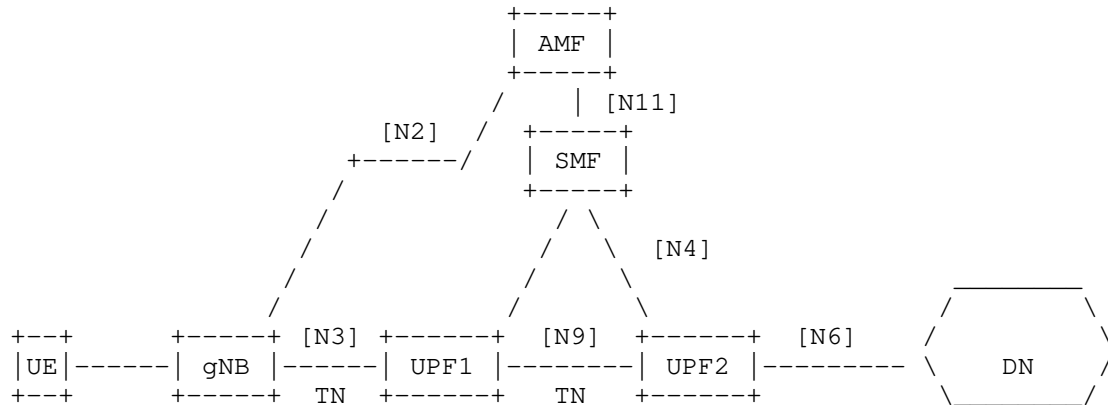


Figure 1: Reference Architecture

- UE : User Equipment
- gNB : gNodeB
- UPF : User Plane Function
- SMF : Session Management Function
- AMF : Access and Mobility Management Function
- 3GPP data plane entities : 3GPP entities responsible for data plane forwarding, i.e. gNB and UPF
- TN : Transport Network - IP network where 3GPP data plane entities connected
- DN : Data Network e.g. operator services, Internet access
- CUPS : Control Plane and User Plane Separation

4. SRv6 mobile user plane and the 5G use cases

4.1. Network Slicing

The SRv6 mobile user plane proposal specifies the Traditional mode and the Enhanced mode. The Traditional mode inherits the existing mobile user plane and minimize the impact to the existing the 3GPP architecture. The Enhanced mode can encode any required SID(s) for constraint path steering and service steering purpose, which enable efficient end-to-end network slicing.

How to build network slicing using the Segment Routing based technology is described in [I-D.ali-spring-network-slicing-building-blocks]

In the typical GTP-U over IP/MPLS/SR configuration, 3GPP data plane entity such as UPF is a CE to the transport networks PE. This results in the following facts:

- A certain Extra ID such as VLAN-ID is needed for segregating traffic and mapping it onto a designated slice.
- PE and the PE-CE connection is a single point of failure, so some form of PE redundancy (using routing protocols, MC-LAG, etc.) is required, which makes systems inefficient and complex.

In the past, this was unavoidable. But nowadays 3GPP dataa plane entities are implemented on servers or dedicated platforms which have virtualized infrastructure, and it is getting common that routing instances are implemented in such servers/platforms.

Furthermore, as stated in the introduction section, SRv6 have been implemented in various forms, it is natural for such servers/platforms to be SRv6 aware.

If the routing administrative domain must be separated between the 3GPP data plane entities and IP network, then BSID (BSID) may be used. With BSID, Topology information is abstracted and not exposed to the 3GPP data plane entities.

The BSID is bound to an SR policy, instantiation of which may involve a list of SIDs. Any packets received with active segment = BSID are steered onto the bound SR Policy, as defined in [RFC8402].

4.2. Edge Computing

Edge computing, where the computing workload is placed closer to users, is recognized as one of the key pillars to meet 5G's demanding key performance indicators (KPIs), especially with regard to low latency and bandwidth efficiency. The computing workload includes network services, security, analytics, content cache and various applications. UPF can also be viewed as a distributed network service.

SRv6's flexible traffic steering capabilities and the Network Programming concept of freely describing instructions and meta data are per se suitable for providing Edge Computing.

In addition, since SRv6 can be a common data plane regardless of the domains such as access, WAN, mobility and data center, Service Placement and Service Chain that used to be concentrated in Data Center can be expanded over a wide area.

Furthermore, with SRv6, session and QoS information can be exposed in IP header. It does not affect performance, thanks to the longest match mechanism in the IP routing. Only the services/applications who need the information for granular processing are to lookup. This also allows services/applications to be placed in between N9 paths if needed.

The draft implementation of Firewall using SRv6 meta data is discussed in [I-D.guichard-spring-srv6-simplified-firewall]

4.3. URLLC (Ultra-Reliable Low-Latency Communication) support

3GPP [TR.23725] investigates the key issues for meeting the URLLC requirements on latency, jitter and reliability in the 5G System. The solutions provided in such document are focused at improving the overlay protocol (GTP-U) and limit to provide a few hints into how to map such tight-SLA into the transport network. These hints are based on static configuration or static mapping for steering the overlay packet into the right transport SLA. Such solutions do not scale and hinder network economics.

Some of the issues can be solved more simply without GTP-U tunnel. SRv6 mobile user plane can expose session and QoS flow information in IP header as discussed in the previous section. This would make routing and forwarding path optimized for URLLC, much simpler than the case with GTP-U tunnel.

Another issue that deserves special mention is the ultra-reliability issue. In 3GPP, in order to support ultra-reliability, redundant user plane paths based on dual connectivity has been proposed. The proposal has two main options.

- Dual Connectivity based end-to-end Redundant User Plane Paths
- Support of redundant transmission on N3/N9 interfaces

In the case of the former, UE and hosts have RHF (Redundancy Handling Function). In sending, RFH is to replicate the traffic onto two GTP-U tunnels, and in receiving, RHF is to merge the traffic.

In the case of the latter, the 3GPP data plane entities are to replicate and merge the packets with the same sequence for specific QoS flow, which requires further enhancements.

SRv6 mobile user plane has some advantages for URLLC traffic. First, it can be used to enforce a low-latency path in the network by means of scalable Traffic Engineering. Additionally, SRv6 provides an automated reliability protection mechanism known as TI-LFA. This is

a sub-50ms FRR mechanism that provides protection regardless of the topology through the optimal backup path.

With the case that dual live-live path is required, the problem is not only the complexity but that the replication point and the merging point would be the single point of failure. The SRv6 mobile user plane also has an advantage in this respect, because any endpoints or 3GPP data plane nodes themselves can be the replication/merging point when they are SRv6 aware.

5. Security Considerations

TBD

6. IANA Considerations

NA

7. Acknowledgements

TBD

8. References

8.1. Normative References

[I-D.hegdeppsenak-isis-sr-flex-algo]

Psenak, P., Hegde, S., Filsfils, C., and A. Gulko, "ISIS Segment Routing Flexible Algorithm", draft-hegdeppsenak-isis-sr-flex-algo-02 (work in progress), February 2018.

[I-D.ietf-6man-segment-routing-header]

Filsfils, C., Dukes, D., Previdi, S., Leddy, J., Matsushima, S., and D. Voyer, "IPv6 Segment Routing Header (SRH)", draft-ietf-6man-segment-routing-header-26 (work in progress), October 2019.

[I-D.ietf-dmm-srv6-mobile-uplane]

Matsushima, S., Filsfils, C., Kohno, M., Camarillo, P., Voyer, D., and C. Perkins, "Segment Routing IPv6 for Mobile User Plane", draft-ietf-dmm-srv6-mobile-uplane-06 (work in progress), September 2019.

[I-D.ietf-spring-srv6-network-programming]

Filsfils, C., Camarillo, P., Leddy, J., Voyer, D., Matsushima, S., and Z. Li, "SRv6 Network Programming", draft-ietf-spring-srv6-network-programming-05 (work in progress), October 2019.

- [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>>.
- [RFC8402] Filsfils, C., Ed., Previdi, S., Ed., Ginsberg, L., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", RFC 8402, DOI 10.17487/RFC8402, July 2018, <<https://www.rfc-editor.org/info/rfc8402>>.

8.2. Informative References

- [I-D.ali-spring-network-slicing-building-blocks]
Ali, Z., Filsfils, C., Camarillo, P., and d. daniel.voyer@bell.ca, "Building blocks for Slicing in Segment Routing Network", draft-ali-spring-network-slicing-building-blocks-01 (work in progress), March 2019.
- [I-D.clt-dmm-tn-aware-mobility]
Chunduri, U., Li, R., Bhaskaran, S., Tantsura, J., Contreras, L., Muley, P., and J. Kaippallimalil, "Transport Network aware Mobility for 5G", draft-clt-dmm-tn-aware-mobility-04 (work in progress), July 2019.
- [I-D.filsfils-spring-srv6-interop]
Filsfils, C., Clad, F., Camarillo, P., Abdelsalam, A., Salsano, S., Bonaventure, O., Horn, J., and J. Liste, "SRv6 interoperability report", draft-filsfils-spring-srv6-interop-02 (work in progress), March 2019.
- [I-D.guichard-spring-srv6-simplified-firewall]
Guichard, J., Filsfils, C., daniel.bernier@bell.ca, d., Li, Z., Clad, F., Camarillo, P., and A. Abdelsalam, "Simplifying Firewall Rules with Network Programming and SRH Metadata", draft-guichard-spring-srv6-simplified-firewall-01 (work in progress), September 2019.
- [I-D.ietf-dmm-fpc-cpdp]
Matsushima, S., Bertz, L., Liebsch, M., Gundavelli, S., Moses, D., and C. Perkins, "Protocol for Forwarding Policy Configuration (FPC) in DMM", draft-ietf-dmm-fpc-cpdp-12 (work in progress), June 2018.

- [I-D.rokui-5g-transport-slice]
Rokui, R., Homma, S., Lopez, D., Foy, X., Contreras, L.,
Ordonez-Lucena, J., Martinez-Julia, P., Boucadair, M.,
Eardley, P., Makhijani, K., and H. Flinck, "5G Transport
Slice Connectivity Interface", draft-rokui-5g-transport-
slice-00 (work in progress), July 2019.
- [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>>.
- [TR.23725]
3GPP, "Study on enhancement of Ultra-Reliable Low-Latency
Communication (URLLC) support in the 5G Core network
(5GC)", 3GPP TR 23.725 16.2.0, June 2019.
- [TR.29892]
3GPP, "Study on User Plane Protocol in 5GC", 3GPP TR
29.892 16.1.0, April 2019.
- [TS.23501]
3GPP, "System Architecture for the 5G System", 3GPP TS
23.501 15.0.0, November 2017.
- [TS.29244]
3GPP, "Interface between the Control Plane and the User
Plane Nodes", 3GPP TS 29.244 15.0.0, December 2017.
- [TS.29281]
3GPP, "General Packet Radio System (GPRS) Tunnelling
Protocol User Plane (GTPv1-U)", 3GPP TS 29.281 15.1.0,
December 2017.

Authors' Addresses

Miya Kohno
Cisco Systems, Inc.
Japan

Email: mkohno@cisco.com

Francois Clad
Cisco Systems, Inc.
France

Email: fclad@cisco.com

Pablo Camarillo Garvia
Cisco Systems, Inc.
Spain

Email: pcamaril@cisco.com

Zafar Ali
Cisco Systems, Inc.
Canada

Email: zali@cisco.com

Internet Engineering Task Force
Internet-Draft
Intended status: Informational
Expires: September 6, 2022

T. Murakami, Ed.
Arrcus, Inc
S. Matsushima
SoftBank
K. Ebisawa
Toyota Motor Corporation
P. Camarillo
R. Shekhar
Cisco Systems, Inc.
March 5, 2022

User Plane Message Encoding
draft-murakami-dmm-user-plane-message-encoding-05

Abstract

This document defines the encoding of User Plane messages into Segment Routing Header (SRH). The SRH carries the User Plane messages over SRv6 Network.

Status of This Memo

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

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

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

This Internet-Draft will expire on September 6, 2022.

Copyright Notice

Copyright (c) 2022 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 (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect

to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Requirements Language	3
3. Conventions and Terminology	3
4. Motivation	3
5. User Plane Message encoding into SRH	3
5.1. GTP-U Header format	4
5.2. Args.Mob.Upmsg	4
5.3. Encoding of Tags Field	5
5.4. User Plane message Information Element Support	6
5.5. SID flavor consideration	7
6. Security Considerations	8
7. IANA Consideration	8
8. Acknowledgements	8
9. References	8
9.1. Normative References	8
9.2. Informative References	9
Authors' Addresses	9

1. Introduction

3GPP defines User Plane function (UPF) and the protocol messages that it supports. The User Plane messages support in-band signalling for path and tunnel management. Currently, User Plane messages are defined in TS 29.281 [TS29281].

When applying SRv6 (Segment Routing IPv6) to the user plane of mobile networks, based on draft-ietf-dmm-srv6-mobile-uplane [I-D.ietf-dmm-srv6-mobile-uplane]. User Plane messages must be carried over SRv6 network. This document defines which User Plane message must be encoded to SRv6 and also defines how to encode the User Plane messages into SRH.

In addition, SRH is mandatory at the ultimate segment upon carrying the User Plane messages because User Plane message is encoded into SRH. Hence, this document considers how to deal with the encoding of User Plane messages into SRH when PSP is applied that SRH is popped out at the penultimate segment.

2. Requirements Language

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

3. Conventions and Terminology

SRv6:	Segment Routing IPv6.
GTP-U:	GPRS Tunneling Protocol User Plane.
UPF:	User Plane Function.
SRH:	IPv6 Segment Routing Header.
PSP:	Penultimate Segment POP of the SRH.
USP:	Ultimate Segment Pop of the SRH.

4. Motivation

3GPP User Plane needs to support the user plane messages associated with a GTP-U tunnel defined in [TS29281]. In the case of SRv6 User Plane [I-D.ietf-dmm-srv6-mobile-uplane], those messages are also required when the user plane interworks with GTP-U.

IPv6 Segment Routing Header (SRH) [RFC8754] is used for SRv6 User Plane. SRH is able to associate additional information to the segments. The Tag field of SRH is capable to indicate different properties within a SID. SRH TLV is capable to provide meta-data to the endpoint node.

The above capability of SRH motivates us to map the user plane messages into it because of the same encapsulation with the packets of carrying client packets. It introduces no additional headers or extension headers to be chained in the packet just for carrying the user plane messages.

5. User Plane Message encoding into SRH

This section defines how to encode the User Plane messages into SRH in order to carry the User Plane messages over SRv6 network.

5.1. GTP-U Header format

3GPP defines GTP-U Header format as shown below.

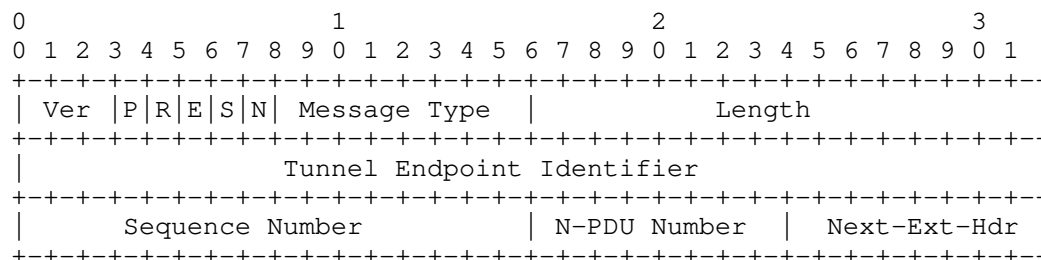


Figure 1: GTP-U Header format

User Plane message type is encoded in Message Type field of GTP-U Header. The following User Plane messages must be carried over SRv6 network at least. The value of each User Plane message type is defined as shown below.

Echo Request:	1
Echo Reply:	2
Error Indication:	26
End Marker:	254

5.2. Args.Mob.Upmsg

draft-ietf-dmm-srv6-mobile-uplane [I-D.ietf-dmm-srv6-mobile-uplane] defines the format of Args.Mob.Session argument which is used in SRv6 SID Mobility Functions in order to carry the PDU Session identifier. The format of Args.Mob.Session is defined as shown below.

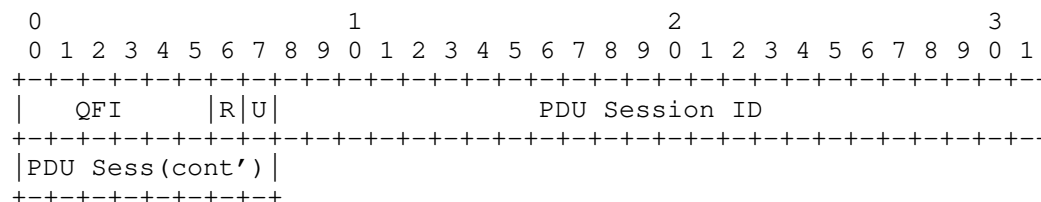


Figure 2: Args.Mob.Session format

In case of Echo Request, Echo Reply and Error Indication, Sequence Number in GTP-U header needs to be carried. Similar to draft-ietf-

dmm-srv6-mobile-uplane [I-D.ietf-dmm-srv6-mobile-uplane], the new arguments to carry Sequence number for Echo Request, Echo Reply and Error Indication message needs to be defined. For this, the following Args.Mobs.Upmsg should be defined newly to carry Sequence number.

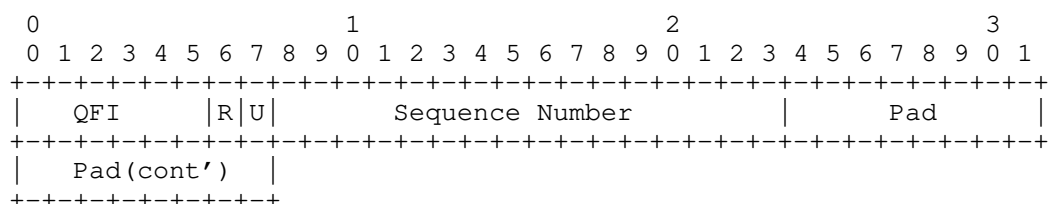


Figure 3: Args.Mob.Upmsg format for Echo Request, Echo Reply and Error Indication

QFI bit, R bit, U bit and 16-bit Sequence Number is encoded in Args.Mobs.Upmsg. The remaining bits followed by Sequence Number must be padded in 0.

In case of End Marker, TEID shall be used as PDU Session ID same as draft-ietf-dmm-srv6-mobile-uplane [I-D.ietf-dmm-srv6-mobile-uplane]. Hence, for End Marker, Args.Mobs.Session should be used to carry TEID as PDU Session ID.

5.3. Encoding of Tags Field

The Segment Routing Header is defined in IPv6 Segment Routing Header (SRH) [RFC8754]. This draft defines 16 bits Tag field but does not define the format or use of this Tag field in the Segment Routing Header.

The User Plane message type encoding is defined in TS 29.281 [TS29281]. Based on this definition, the User Plane message type must be encoded into the Tag field in the Segment Routing Header in order to indicate the type of the user plane messages for at least Echo Request, Echo Reply, Error Indication or End Marker.

Only UPF must process the Tag field where the user plane message is encoded. In addition, when the user plane message is encoded in the Tag field, the UPF should not encode any segments in the Segment Routing Header whose function modifies the Tag field value. Any other transport router implementing SRv6 must ignore the Tag field upon processing the Segment Routing Header.

The user plane messages must be encoded into the Tag field as shown below.

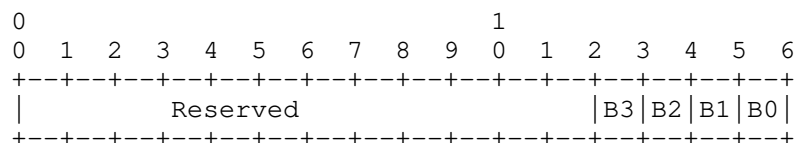


Figure 4: Tag Field Encoding

- Bit 0 [B0]: End Marker
- Bit 1 [B1]: Error Indication
- Bit 2 [B2]: Echo Request
- Bit 3 [B3]: Echo Reply

End Marker, Echo Request and Echo reply messages do not require any additional information elements. However, Error Indication message requires the additional information elements like Tunnel Endpoint Identifier Data IE, GSN Address, etc. These additional information elements can be encoded into the SRH TLV that is defined in the next section.

5.4. User Plane message Information Element Support

End Maker, Echo Request and Echo Reply messages do not require any additional information elements. However, Error Indication message requires additional 3GPP IEs (Information Element). These additional information elements must be carried over SRv6 network as well. However SRv6 SID has limited space only. Hence it cannot carry a lot of information elements.

In order to carry more information elements, SRH TLV shall be leveraged. SRH TLV is defined in IPv6 Segment Routing Header (SRH) [RFC8754] in order to carry the meta-data for the segment processing. In order to carry additional User Plane messages like 3GPP IEs, the new type named as "User Plane Container" must be defined as the new SRH TLV. The "User Plane Container" can carry additional User Plane messages which includes multiple 3GPP IEs with 1 sub-TLV.

```

      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      | User Plane message sub-TLV |
+-----+-----+-----+-----+-----+-----+-----+-----+
//                      User Plane message sub-TLV                      //
+-----+-----+-----+-----+-----+-----+-----+-----+

```

User Plane Container TLV

Type: to be assigned by IANA

Length: Length of User Plane message sub-TLV

User Plane message sub-TLV: User Plane message sub-TLV 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
+-----+-----+-----+-----+-----+-----+-----+-----+
|          Type          |      Length      |          Value          |
+-----+-----+-----+-----+-----+-----+-----+-----+

```

User Plane message sub-TLV

Type: Type of User Plane message sub-TLV

3GPP IE sub-TLV: 0x01

Length: Length of Value

Value: User Plane Message data

3GPP IE sub-TLV: multiple 3GG IEs

5.5. SID flavor consideration

This section considers SID flavor of where the SRH is popped out at either the penultimate or the ultimate segment.

In order to carry User Plane message over SRv6 network, SRH must be sustained over entire SRv6 network because User Plane message type and required information elements are encoded into SRH. If the

penultimate segment is popping out SRH, i.e., PSP, User Plane message can not be carried in entire SRv6 network.

In order to avoid this problem, USP is recommended in SRv6 Mobile network. In this case, SRH is never popped out and User Plane message can be sustained over entire SRv6 network.

However, if PSP needs to be enabled in SRv6 network, it is also a possible solution to encap another SRH which carries User Plane message along with the outer IPv6 or SRH.

6. Security Considerations

This document does not raise any additional security issues. This document just define the mechanisms for mapping between user plane message (GTP-U message) and SRH in SRv6. Basically, since this document is using SRH defined in [RFC8754] to carry user plane message, same security consideration stated in [RFC8754] shall be applied.

7. IANA Consideration

The type value of SRH TLV for User Plane Container must be assigned by IANA.

8. Acknowledgements

9. References

9.1. 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>>.
- [RFC2460] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, DOI 10.17487/RFC2460, December 1998, <<https://www.rfc-editor.org/info/rfc2460>>.
- [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>>.

9.2. Informative References

- [I-D.ietf-dmm-srv6-mobile-uplane]
Matsushima, S., Filsfils, C., Kohno, M., Garvia, P. C., Voyer, D., and C. E. Perkins, "Segment Routing IPv6 for Mobile User Plane", draft-ietf-dmm-srv6-mobile-uplane-18 (work in progress), February 2022.
- [RFC1918] Rekhter, Y., Moskowitz, B., Karrenberg, D., de Groot, G., and E. Lear, "Address Allocation for Private Internets", BCP 5, RFC 1918, DOI 10.17487/RFC1918, February 1996, <<https://www.rfc-editor.org/info/rfc1918>>.
- [RFC3513] Hinden, R. and S. Deering, "Internet Protocol Version 6 (IPv6) Addressing Architecture", RFC 3513, DOI 10.17487/RFC3513, April 2003, <<https://www.rfc-editor.org/info/rfc3513>>.
- [RFC8754] Filsfils, C., Ed., Dukes, D., Ed., Previdi, S., Leddy, J., Matsushima, S., and D. Voyer, "IPv6 Segment Routing Header (SRH)", RFC 8754, DOI 10.17487/RFC8754, March 2020, <<https://www.rfc-editor.org/info/rfc8754>>.
- [TS29281] 3GPP, "General Packet Radio System (GPRS) Tunnelling Protocol User Plane (GTPv1-U)", 2019, <http://www.3gpp.org/ftp//Specs/archive/29_series/29.281/29281-f60.zip>.

Authors' Addresses

Tetsuya Murakami (editor)
Arrcus, Inc
2077 Gateway Place, Suite 400
San Jose
USA

Email: tetsuya@arrcus.com

Satoru Matsushima
SoftBank
1-9-1 Higashi-Shinbashi, Munato-ku
Tokyo
Japan

Email: satoru.matsushima@g.softbank.co.jp

Kentaro Ebisawa
Toyota Motor Corporation
Tokyo
Japan

Email: ebisawa@toyota-tokyo.tech

Pablo Camarillo
Cisco Systems, Inc.
Spain

Email: pcamaril@cisco.com

Ravi Shekhar
Cisco Systems, Inc.
USA

Email: ravishek@cisco.com

DMM Working Group
Internet-Draft
Intended status: Standards Track
Expires: March 22, 2020

Z. Yan
G. Geng
CNNIC
J. Lee
Sangmyung University
H. Chan
Huawei Technologies
September 19, 2019

Mobility Capability Negotiation and Protocol Selection
draft-yan-dmm-man-05

Abstract

Based on different requirements, multiple mobility management protocols have been developed. Different protocols have different functional requirements on the network element or the host and then a scheme should be used in order to support the negotiation and selection of adopted mobility management protocol when a host accesses to a new network. In this draft, this issue is analyzed.

Status of This Memo

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

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

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

This Internet-Draft will expire on March 22, 2020.

Copyright Notice

Copyright (c) 2019 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 (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents

carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Motivations	4
3. Possible Cases	4
4. Principles and Possible Procedure	9
5. Extensions	9
6. Security Considerations	12
7. IANA Considerations	12
8. References	12
8.1. Normative References	12
8.2. Informative References	13
Authors' Addresses	13

1. Introduction

In order to clearly analyze the possible cases, the following category labels of the mobility management protocols are defined:

- o Mobile IPv6 (MIPv6) protocol: the mobility management scheme based on [RFC6275].
- o Proxy Mobile IPv6 (PMIPv6) protocol: the mobility management scheme based on [RFC5213].
- o MIPv6 suit protocols: based on MIPv6, there are multiple extension protocols have been standardized. These protocols can be classified into two types: protocols for the function extension and protocols for the performance enhancement. The protocols for the function extension are proposed to support some specific scenarios or functions, such as Dual-stack Mobile IPv6 (DSMIPv6) [RFC5555] for mobility of the dual-stack nodes, Multiple Care-of-address (MCoA) [RFC5648] for hosts with multiple access interfaces and Network Mobility (NEMO) [RFC3963] for mobility of sub-network. The other type is proposed to enhance the performance of the mobility management, such as Fast Mobile IPv6 (FMIPv6) [RFC5268] for fast handover, Hierarchical Mobile IPv6 (HMIPv6) [RFC5380] for hierarchical mobility optimization. In the MIPv6 suit protocols, location update is initiated by the host and the tunnel is also terminated at the host.
- o PMIPv6 suit protocols: in order to reduce the protocol cost and enhance the handover performance further, the network-based mobility management protocols were proposed and PMIPv6 was standardized as a basis. Based on PMIPv6, a series of its

- extensions were proposed, such as Dual-stack Proxy Mobile IPv6 (DS-PMIPv6) [RFC5844], and Distributed Mobility Management Proxy Mobile IPv6 (DMM-PMIPv6) [RFC7333]. Be different from the MIPv6 suit protocols, the location update in PMIPv6 suit protocols is triggered by the network entity and the tunnel is established between network entities. Then the host needs to do nothing about the signaling exchange during the movement, particularly, the mobility is transparent to the IP layer of the host.
- o Network-based protocols: generally, it means the mobility management protocols which do not require the involvement of the mobile node in order to accomplish mobility. It includes PMIPv6 suit protocols and other network-based solutions, such as GPRS Tunnelling Protocol (GTP) [TS.29274][TS.29281].
 - o Host-based protocols: generally, the mobility management protocols which require the involvement of the mobile node in order to accomplish mobility. It includes MIPv6 suit protocols and other host-based solutions, such as Host Identity Protocol (HIP) [RFC7401] and IKEv2 Mobility and Multihoming Protocol (MOBIKE) [RFC4555].

Figure 1 illustrates the scopes of the above different category labels.

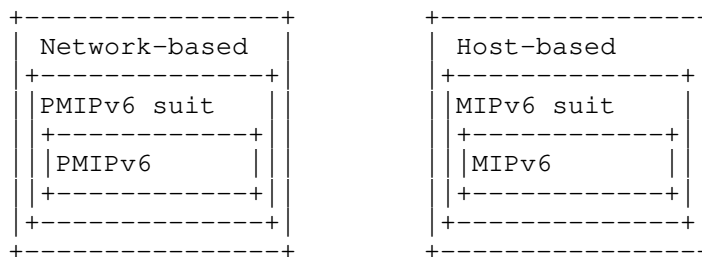


Figure 1: Scopes of different protocol category labels

In reality, the host-based protocols and network-based protocols will be co-existing and multiple protocol daemons will be configured on the network entities or host. That means a scheme is needed to support the negotiation and selection of mobility management protocol when the host accesses into a new access network initially or handover happens [Paper-CombiningMobilityStandards].

This document tries to present the principles for the protocol selection and analyze the possible scenarios which should be supported by the further solution.

2. Motivations

As illustrated above, these protocols may co-exist in reality and simultaneously be used in an access network or even the same entity. Due to their different requirements on the network entity or host, a scheme is needed to support the negotiation and selection of adopted mobility management protocol when the host accesses to a new network. Generally, two problems should be solved:

- o What principles should be followed for the protocol negotiation and selection?
- o What procedure should be adopted for the protocol negotiation and selection?

This scheme is needed because network entity and host may have different capabilities and preferences (may be decided by the capability and mobility pattern of the host). This scheme aims to guarantee that the optimum and most suitable protocol will be used.

3. Possible Cases

From both host and network aspects, their capacities of mobility management may have multiple cases as shown in Figure 2. We mainly analyze that host and network support single protocol, if multiple protocols are supported simultaneously by the host or network side, multiple cases exist at the same time but the logic is same as that in the case with single protocol supported. Specifically, the following cases should be considered.

1) Network supports network-based protocol, host supports network-based protocol

In this case, there are the following sub-cases:

a) Host supports PMIPv6 suit protocol, Network supports PMIPv6 suit protocol

- o if host supports PMIPv6 and network supports PMIPv6, PMIPv6 is selected.
- o if host supports PMIPv6 and network supports extended PMIPv6 protocol, extended PMIPv6 protocol is selected if no host involvement is needed, otherwise the plain PMIPv6 is selected (we assume that the extension protocols are backward-compatible with the related plain protocol).
- o if host supports extended PMIPv6 protocol and network supports PMIPv6, PMIPv6 is selected (we assume that the extension protocols are backward-compatible with the related plain protocol).

- o if host supports extended PMIPv6 protocol and network supports extended PMIPv6 protocol, the identical extension protocol is selected, otherwise, PMIPv6 is selected (we assume that the extension protocols are backward-compatible with the related plain protocol).

Network-based	PMIPv6 suit	PMIPv6	
			DS-PMIPv6
		PMIPv6 extensions	FPMIPv6
			DMM-PMIPv6
			...
Host-based	MIPv6 suit	MIPv6	
			DS-MIPv6
			FMIPv6
		MIPv6 extensions	HMIPv6
			NEMO
			DMM-MIPv6
			...
	Others	GTP	
		...	
		HIP	
		MOBIKE	
		...	

Figure 2: Possible capacities of host and network

- b) Host supports PMIPv6 suit protocol, Network supports other network-based protocol

- o if host supports PMIPv6 and network supports other network-based protocol, other network-based protocol is selected if no host involvement is needed, otherwise failure.
 - o if host supports extended PMIPv6 protocol and network supports other network-based protocol, other network-based protocol is selected if no host involvement is needed, otherwise failure.
- c) Host supports other network-based protocol, Network supports PMIPv6 suit protocol
- o if host supports other network-based protocol and network supports PMIPv6, PMIPv6 is selected.
 - o if host supports other network-based protocol and network supports extended PMIPv6 protocol, extended PMIPv6 protocol is selected if no host involvement is needed, otherwise failure.
- d) Host supports other network-based protocol, Network supports other network-based protocol
- o the identical protocol is selected, otherwise follow network capability if the protocols are different.
- 2) Network supports network-based protocol, host supports host-based protocol

In this case, there are the following sub-cases:

- a) Host supports PMIPv6 suit protocol, Network supports MIPv6 suit protocol
- o if host supports PMIPv6 and network supports MIPv6, failure.
 - o if host supports PMIPv6 and network supports extended MIPv6 protocol, failure.
 - o if host supports extended PMIPv6 protocol and network supports MIPv6, failure.
 - o if host supports extended PMIPv6 protocol and network supports extended MIPv6 protocol, failure.
- b) Host supports PMIPv6 suit protocol, Network supports other host-based protocol
- o if host supports PMIPv6 and network supports other host-based protocol, failure.
 - o if host supports extended PMIPv6 protocol and network supports other host-based protocol, failure.
- c) Host supports other network-based protocol, Network supports MIPv6 suit protocol

- o if host supports other network-based protocol and network supports MIPv6, failure.
 - o if host supports other network-based protocol and network supports extended MIPv6 protocol, failure.
- d) Host supports other network-based protocol, Network supports other host-based protocol
- o failure.
- 3) Network supports host-based protocol, host supports network-based protocol

In this case, there are the following sub-cases:

- a) Host supports MIPv6 suit protocol, Network supports PMIPv6 suit protocol
- o if host supports MIPv6 and network supports PMIPv6, PMIPv6 is selected in default and MIPv6 is selected if host prefers it.
 - o if host supports MIPv6 and network supports extended PMIPv6 protocol, extended PMIPv6 is selected in default, then PMIPv6 is selected with the lower priority and MIPv6 is selected if host prefers it.
 - o if host supports extended MIPv6 protocol and network supports PMIPv6, PMIPv6 is selected in default, then extended MIPv6 protocol is selected if host prefers it and network also supports, otherwise MIPv6 is selected with the lowest priority.
 - o if host supports extended MIPv6 protocol and network supports extended PMIPv6 protocol, extended PMIPv6 protocol is selected in default, then PMIPv6 is selected, then extended MIPv6 protocol is selected if host prefers and network also supports, otherwise MIPv6 is selected with the lowest priority.
- b) Host supports MIPv6 suit protocol, Network supports other network-based protocol
- o if host supports MIPv6 and network supports other network-based protocol, other network-based protocol is selected if no host involvement is needed, otherwise failure.
 - o if host supports extended MIPv6 protocol and network supports other network-based protocol, other network-based protocol is selected if no host involvement is needed, otherwise failure.
- c) Host supports other host-based protocol, Network supports PMIPv6 suit protocol

- o if host supports other host-based protocol and network supports PMIPv6, PMIPv6 is selected in default, otherwise failure.
 - o if host supports other host-based protocol and network supports extended PMIPv6 protocol, extended PMIPv6 protocol is selected if no host involvement is needed, otherwise failure.
- d) Host supports other host-based protocol, Network supports other network-based protocol
- o other network-based protocol is selected if no host involvement is needed, otherwise failure.
- 4) Network supports host-based protocol, host supports host-based protocol

In this case, there are the following sub-cases:

- a) Host supports MIPv6 suit protocol, Network supports MIPv6 suit protocol
- o if host supports MIPv6 and network supports MIPv6, MIPv6 is selected.
 - o if host supports MIPv6 and network supports extended MIPv6 protocol, MIPv6 is selected.
 - o if host supports extended MIPv6 protocol and network supports MIPv6, MIPv6 is selected.
 - o if host supports extended MIPv6 protocol and network supports extended MIPv6 protocol, the identical protocol is selected, otherwise MIPv6 is selected.
- b) Host supports MIPv6 suit protocol, Network supports other host-based protocol
- o if host supports MIPv6 and network supports other host-based protocol, failure.
 - o if host supports extended MIPv6 protocol and network supports other host-based protocol, failure.
- c) Host supports other host-based protocol, Network supports MIPv6 suit protocol
- o if host supports other host-based protocol and network supports MIPv6, failure.
 - o if host supports other host-based protocol and network supports extended MIPv6 protocol, failure.
- d) Host supports other host-based protocol, Network supports other host-based protocol

- o the identical other host-based protocol is selected, otherwise failure.
- 5) Network supports host-based protocol and network-based protocol, host supports host-based protocol and network-based protocol
- o follow the network based protocol in default if the host can support, otherwise select the protocol both network and host can support if host prefers.

4. Principles and Possible Procedure

Two different schemes may be used for the protocol negotiation and selection: host-initiated and network-initiated. Within the MIPv6/PMIPv6 protocols, the priority of the function-extension protocols should be higher than the performance-enhancement protocols. Generally, the following principles should be followed:

- o In default: Network based scheme if it can be supported
- o Priority 1: Follow network capability
- o Priority 2: Follow host preference
- o Priority 3: Support the functional extensions
- o Priority 4: Support the performance enhancements

And the general procedure for the protocol selection should be:

- o During initiation, network-based protocol may be used as a default mobility management protocol once the network supports it.
- o If the host prefers host-based protocols, a negotiation is executed to handover from network-based protocol to host-based protocol.
- o After initial attachment, a profile will be generated in the management store to record the selected or preferred protocol of this host.
- o When the handover happens, the network will check the selected or preferred protocol during the authentication process. But the network also needs to notify the host if the selected protocol cannot be supported herein.

5. Extensions

In order to fulfill the above principles, some extensions should be supported, for example:

- 1) Extended negotiation messages

The protocol negotiation may be included in the MN_ATTACH Function [MN-AR.IF] and the implementation may be based on a new signaling

message or extended messages (e.g., ICMPv6, Diameter, and RADIUS). Besides these, some other protocols may also be used in some specified scenarios, such as extended IEEE 802.21 primitives.

As a possible solution, a new option under ICMPv6 is proposed in this draft in order to support the protocol negotiation when the mobile terminal initially accesses the network or hands over to a different network. In the RA and Router Solicitation (RS) message headers, a one-bit flag (C) is used to illustrate that mobility capability negotiation is needed and a Mobility Capability (MC) option is included in the message body. The format of MC option is shown in Figure 3.

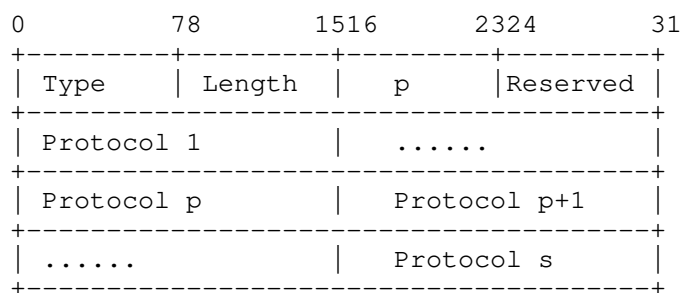


Figure 3: The format of Mobility Capability option

"Type" indicates that this option is of the type Mobility Capability. "p" is the number of preferred protocols. "Protocol 1" to "Protocol s" is a list of s supported protocols, which can be selected. Out of the s supported protocols, the first p protocols are ones preferred by the network and the terminal, listed in the order of preference, whereas no preference is indicated in the remaining protocols from p+1 to s. How to code the protocol types with the 16-bit space is implementation-dependent.

"Length" is the number of octets in this option excluding option type and option length, and it can be seen that $\text{Length} = 2 \times (s+1)$.

"Reserved" is for future use.

It is noted that when $p = 0$, preference is not indicated in the entire list of supported protocols, and when $p = s$, preference is indicated in all the supported protocols.

Based on this extension, when the mobile terminal receives the RA message with the "C" flag set to 1 and "p" in MC option is at least one, it means that the access network has selected the first supported protocol (protocol 1) as the default protocol for the

terminal. Based on the previous principles, the terminal should follow this selected protocol if it is able to. If the terminal is not capable to use the first supported protocol, it will use the second supported protocol (protocol 2) if it is able to. If it is still not capable of using the second supported protocol, it will try the third one and so on.

If the terminal is not capable of using any of the p supported protocols, it will try to use any of the remaining protocols. When the mobile terminal receives the RA message with the "C" flag set to 1 and "Preferred protocol" in MC option is null, it means that the access network has not selected the default protocol for the terminal but illustrated to the terminal about the supported protocols from the network side. Then based on the previous principles, the terminal should select one protocol and notify it to the network with the MC option in RS message. The network should acknowledge it with a new re-formatted RA message. In this new RA message, the protocol selected by terminal is included in the "Preferred protocol" of MC option. After the choice has been made, the terminal may inform the network of the choice by sending a message with MC option in which $p = s = 1$, and the protocol field is the selected protocol.

When the network receives the RS message with the "C" flag set to 1 and "p" in MC option is zero, it means that the terminal only lists its supported mobility management protocols but does not have any preference. The network will then select one based on the principles and notify the selected protocol to the terminal with a RA message containing the MC option in which $p=1$ and "protocol 1" is the selected protocol.

When the network receives the RS message with the "C" flag set to 1 and "p" in MC option is at least one, it means that the terminal tries to negotiate the mobility management protocol and has included the preferred protocols listed in the order of preference. The network will try one by one to select from protocol 1 to protocol p until it has found one it supports. If the network is not capable of supporting all the p protocols, it will try the remaining $s-p$ protocols.

2) Extended management store

When the host accesses to the network, an authentication should be executed before the mobility management service is provided. In order to support the mobility management protocol selection, a new information should be recorded by the network after the successful authentication during the initial attachment. The newly introduced information shows the selected mobility management protocol and should be updated when the used protocol changes.

6. Security Considerations

Generally, this function will not incur additional security issues. The detailed influence should be analyzed in the future.

7. IANA Considerations

A new ICMP option or authentication option or other signaling message may be used with a new code number.

8. References

8.1. Normative References

- [MN-AR.IF] Laganier, J., Narayanan, S., and P. McCann, "Interface between a Proxy MIPv6 Mobility Access Gateway and a Mobile Node", draft-ietf-netlmm-mn-ar-if-03, February 2008.
- [RFC3963] Devarapalli, V., Wakikawa, R., Petrescu, A., and P. Thubert, "Network Mobility (NEMO) Basic Support Protocol", RFC 3963, DOI 10.17487/RFC3963, January 2005, <<https://www.rfc-editor.org/info/rfc3963>>.
- [RFC4555] Eronen, P., "IKEv2 Mobility and Multihoming Protocol (MOBIKE)", RFC 4555, DOI 10.17487/RFC4555, June 2006, <<https://www.rfc-editor.org/info/rfc4555>>.
- [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>>.
- [RFC5268] Koodli, R., Ed., "Mobile IPv6 Fast Handovers", RFC 5268, DOI 10.17487/RFC5268, June 2008, <<https://www.rfc-editor.org/info/rfc5268>>.
- [RFC5380] Soliman, H., Castelluccia, C., ElMalki, K., and L. Bellier, "Hierarchical Mobile IPv6 (HMIPv6) Mobility Management", RFC 5380, DOI 10.17487/RFC5380, October 2008, <<https://www.rfc-editor.org/info/rfc5380>>.
- [RFC5555] Soliman, H., Ed., "Mobile IPv6 Support for Dual Stack Hosts and Routers", RFC 5555, DOI 10.17487/RFC5555, June 2009, <<https://www.rfc-editor.org/info/rfc5555>>.

- [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, <<https://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, <<https://www.rfc-editor.org/info/rfc5844>>.
- [RFC6275] Perkins, C., Ed., Johnson, D., and J. Arkko, "Mobility Support in IPv6", RFC 6275, DOI 10.17487/RFC6275, July 2011, <<https://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, <<https://www.rfc-editor.org/info/rfc7333>>.
- [RFC7401] Moskowitz, R., Ed., Heer, T., Jokela, P., and T. Henderson, "Host Identity Protocol Version 2 (HIPv2)", RFC 7401, DOI 10.17487/RFC7401, April 2015, <<https://www.rfc-editor.org/info/rfc7401>>.
- [TS.29274] "3GPP Evolved Packet System (EPS); Evolved General Packet Radio Service (GPRS) Tunnelling Protocol for Control plane (GTPv2-C); Stage 3", 3GPP TS 29.274 8.10.0, June 2011.
- [TS.29281] "General Packet Radio System (GPRS) Tunnelling Protocol User Plane (GTPv1-U)", 3GPP TS 29.281 10.3.0, September 2011.

8.2. Informative References

- [Paper-CombiningMobilityStandards]
Oliva, A., Soto, I., Calderon, M., Bernardos, C., and M. Sanchez, "The costs and benefits of combining different IP mobility standards", Computer Standards and Interfaces, February 2013.

Authors' Addresses

Zhiwei Yan
CNNIC
No.4 South 4th Street, Zhongguancun
Beijing 100190
China

Email: yan@cnnic.cn

Guanggang Geng
CNNIC
No.4 South 4th Street, Zhongguancun
Beijing 100190
China

Email: ggg@cnnic.cn

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

Email: jonghyouk@smu.ac.kr

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

Email: h.a.chan@ieee.org