

Internet Engineering Task Force  
Internet-Draft  
Intended status: Standards Track  
Expires: March 27, 2020

A. Aguado  
Nokia  
O. Gonzalez de Dios, Ed.  
V. Lopez  
Telefonica  
D. Voyer  
Bell Canada  
L. Munoz  
Vodafone  
September 24, 2019

Layer 3 VPN Network Model  
draft-aguado-opsawg-l3sm-l3nm-02

Abstract

RFC8299 defines a L3VPN Service YANG data Model (L3SM) that can be used for communication between customers and network operators. Such model is adequate for the customer to network operator conversation and plays the role of a Customer Service Model, according to the terminology defined in RFC8309.

There is a need for a YANG model to be used in the communication between the entity that interacts directly with the customer, the service orchestrator, (either fully automated or a human operator) and the entity in charge of network orchestration and control (aka network controller / orchestrator).

This document proposes a L3VPN Network Yang Model (L3NM) to facilitate communication between a service orchestrator and a network controller / orchestrator. The resulting model is called the L3VPN Network Model (L3NM) and provides a network-centric view of the L3VPN services.

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 27, 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. TERMINOLOGY . . . . .	3
1.2. Requirements Language . . . . .	3
2. REFERENCE ARCHITECTURE . . . . .	4
3. YANG MODEL EXPLANATION . . . . .	6
3.1. STRUCTURE OF THE MODEL . . . . .	7
3.2. SITE AND BEARERS . . . . .	7
3.3. BEARER AND ETHERNET ENCAPSULATION . . . . .	7
3.4. MULTI-DOMAIN RESOURCE MANAGEMENT . . . . .	7
3.5. REMOTE FAR-END CONFIGURATION . . . . .	8
3.6. PROVIDE EDGE IDENTIFICATION POINT . . . . .	8
4. DESING OF THE DATA MODEL . . . . .	9
5. YANG MODULE . . . . .	20
6. IANA CONSIDERATIONS . . . . .	93
7. SECURITY CONSIDERATIONS . . . . .	93
8. IMPLEMENTATION STATUS . . . . .	93
9. ACKNOWLEDGEMENTS . . . . .	94
10. CONTRIBUTORS . . . . .	94
11. References . . . . .	94
11.1. NORMATIVE REFERENCES . . . . .	94
11.2. INFORMATIVE REFERENCES . . . . .	94
Authors' Addresses . . . . .	95

## 1. INTRODUCTION

[RFC8299] defines a L3VPN Service YANG data Model (L3SM) model that can be used for communication between customers and network operators. Such model is focused on describing the customer view of the services, and provides an abstracted view of the customer's requested services. That approach limits the usage of the L3SM to the role of a Customer Service Model, according to the terminology defined in [RFC8309].

The YANG data model proposed in this document is called the L3VPN Network Model (L3NM). The L3NM model is aimed at providing a network-centric view of L3 VPN Services. The model can be used to facilitate communication between the service orchestrator, and the network controller / orchestrator. It enables further capabilities, such as resource management or to serve as a multi-domain orchestration interface, where transport resources must be synchronized. The YANG module has been built with a prune and extend approach, taking as a starting points the YANG model described in [RFC8299].

Hence, this document does not obsolete, but complements, the definitions in [RFC8299]. It aims to provide a different scope for the L3SM, but does not attempt to address all deployment cases especially those where the L3VPN connectivity is supported through the coordination of different VPNs in different underlying networks. More complex deployment scenarios involving the coordination of different VPN instances and different technologies to provide end-to-end VPN connectivity are addressed by a complementary YANG model defined in [I-D.evenwu-opsawg-yang-composed-vpn].

### 1.1. TERMINOLOGY

This document assumes that the reader is familiar with the contents of [RFC6241], [RFC7950], [RFC8299], [RFC8309], and [RFC8453] and uses terminology from those documents. Tree diagrams used in this document follow the notation defined in [RFC8340].

### 1.2. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

## 2. REFERENCE ARCHITECTURE

Figure 1 shows where the L3NM is used in a management stack. The figure is an expansion of the architecture presented in Section 5 of [RFC8299] and decomposes the box marked "orchestration" in that figure into three separate functional components called "Service Orchestration", "Network Orchestration", and "Domain Orchestration".

Note that some implementations may choose to construct a monolithic orchestration component, but this document assumes that there are many benefits for flexibility of implementation and deployment to separate the functional components, and that separation demands the existence of separate YANG models to be used between the components.

At the same time, terminology from [RFC8309] is introduced to show the distinction between the "Customer Service Model", the "Service Delivery Model", the "Network Configuration Model", and the "Device Configuration Model". In that context, the "Domain Orchestration" and "Config Manager" roles may be performed by "Controllers".

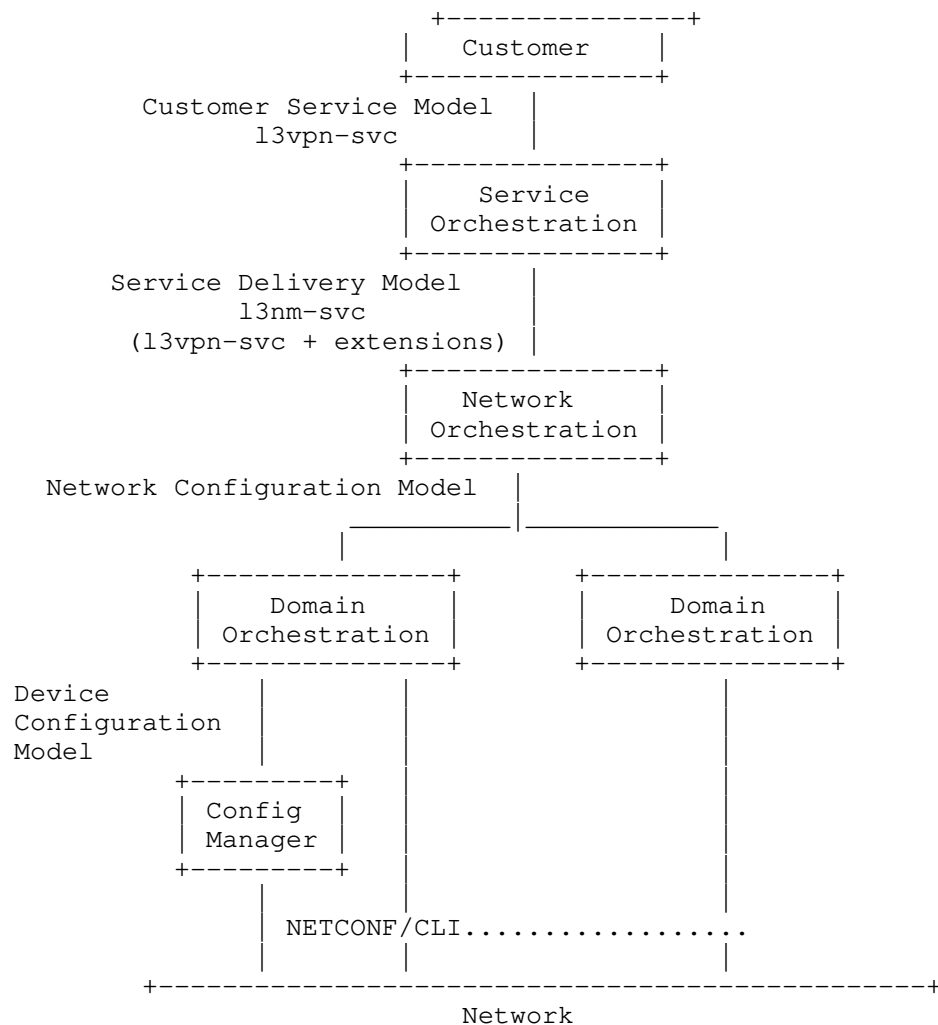


Figure 1: L3SM and L3NM

The L3SM and L3NM may also be set in the context of the ACTN architecture [RFC8453]. Figure 2 shows the Customer Network Controller (CNC), the Multi-Domain Service Coordinator (MDSC), and the Provisioning Network Controller (PNC). It also shows the interfaces between these functional units: the CNC-MDSC Interface (CMI), the MDSC-PNC Interface (MPI), and the Southbound Interface (SBI).

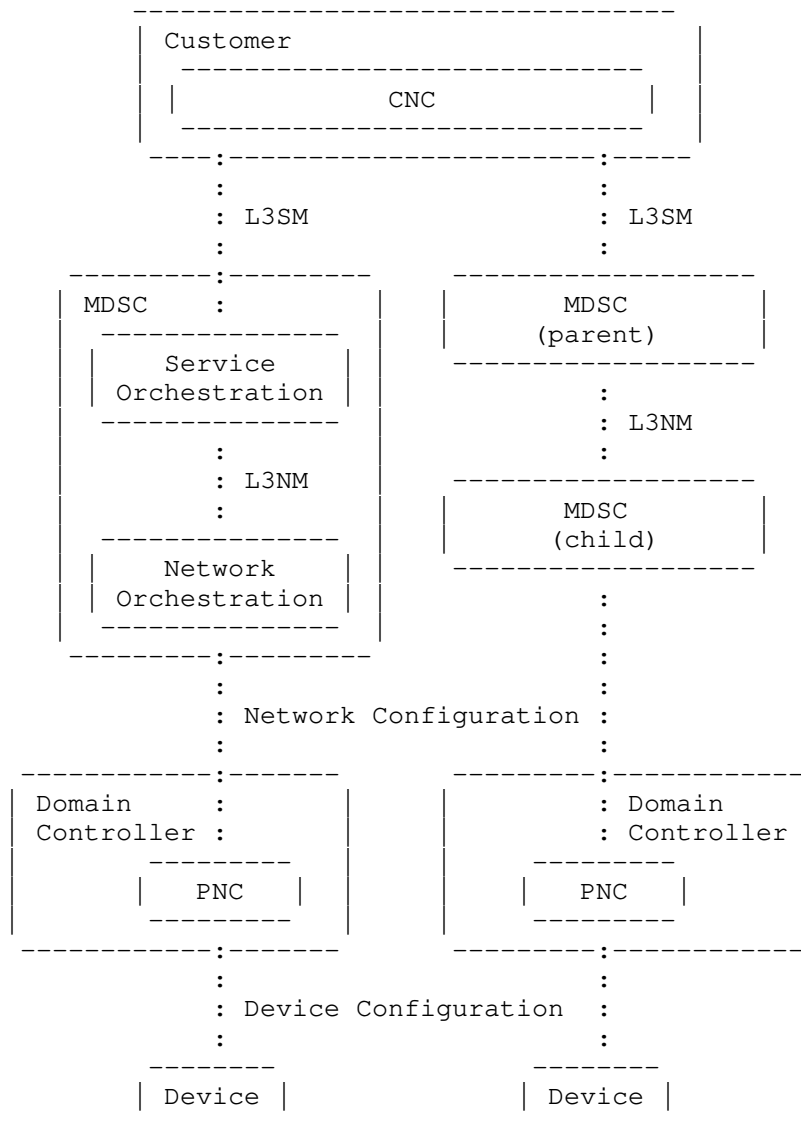


Figure 2: L3SM and L3NM in the Context of ACTN

### 3. YANG MODEL EXPLANATION

The scenarios covered in the L3NM model includes: the integration of Ethernet and encapsulation parameters, the extension for transport resources (e.g., Route targets and Route distinguishers) to be orchestrated from the management system, far-end configuration of PEs

not managed by the management system and the definition for PE identification.

### 3.1. STRUCTURE OF THE MODEL

The YANG module is divided into three main containers: "vpn-services", "sites" and "vpn-profiles".

### 3.2. SITE AND BEARERS

A site, as per [RFC8299], represents a connection of a customer office to one or more VPN services. As this YANG module, is the network view, each site is associated with a list of bearers. A bearer is the layer two connections with the site. In the module it is assumed that the bearer has been allocated by the Service Provider (e.g., by the service orchestrator). The bearer is associated to a network element and a port. Hence, a bearer is not just a bearer-reference, but also a true reference to a given port in the service provider network.

### 3.3. BEARER AND ETHERNET ENCAPSULATION

The definition of a L3VPN is commonly specified not only at the IP layer, but also requires to identify parameters at the Ethernet layer, such as encapsulation type (e.g., VLAN, QinQ, QinAny, VxLAN, etc.). This specification is not supported in [RFC8299], whilst it suggests that any extension on this direction shall be implemented via augmentation of the bearer container. The extension defined to cope with these parameters uses the connection container inside the site-network-access defined by the [RFC8466]. This container defines protocol parameters to enable connectivity at Layer 2. In the context of L3SM, the augmentation includes only mandatory parameters for the service configuration, which are mainly related to the interface encapsulation. Other definitions from L2SM connection container are left aside. For example, Link Aggregation (LAG) information is not required and it shall be configured prior to the service configuration, being the aggregated interface identified in the model as the bearer-reference, as discussed later in Section 3.4.

### 3.4. MULTI-DOMAIN RESOURCE MANAGEMENT

The implementation of L3VPN services which span across administratively separated domains (i.e., that are under the administration of different management systems or controllers) requires some network resources to be synchronized between systems. Particularly, there are two resources that must be orchestrated and manage to avoid asymmetric (non-functional) configuration, or the usage of unavailable resources. For example, RTs shall be

synchronized between PEs. When every PE is controlled by the same management system, RT allocation can be performed by the system. In cases where the service spans across multiple management systems, this task of allocating RTs has to be aligned across the domains, therefore, the service model must provide a way to specify RTs. In addition, RDs must also be synchronized to avoid collisions in RD allocation between separate systems. An incorrect allocation might lead to the same RD and IP prefixes being exported by different PE routers.

### 3.5. REMOTE FAR-END CONFIGURATION

Depending on the control plane implementation, different network scenarios might require additional information for the L3VPN service to be configured and active. For example, an L3VPN Option C service, if no reflection of IPv4 VPN routes is configured via ASBR or route reflector, may require additional configuration (e.g. a new BGP neighbor) to be coordinated between both management systems. This definition requires for every management system participant in the VPN to receive not just their own sites and site-network-accesses, but also to receive information about external ones, identified as an external site-network-access-type. In addition, this particular site-network-access is augmented to include the loopback address of the far-end (remote/external) PE router.

### 3.6. PROVIDE EDGE IDENTIFICATION POINT

[RFC8299] states that the "bearer-reference" parameter is used in cases where the customer has already ordered a network connection to the service provider (SP) apart from the IP VPN site and wants to reuse this connection. The string used is an internal reference from the SP and describes the already-available connection. Usually, a client interface (either a customer one or an interface used by the SP) is already in place and connected, although it has not being use previously. In some other cases (e.g., for stitching purposes), the termination of a VPN service is done over logical terminations within a PE router.

The bearer-reference must serve as a strict unequivocal parameters to identify the connection between a PE and a client (CE). This means that, despite the type is maintained as a string and there is no restriction in the way this data is formed, the bearer-reference must serve as the unique way to identify the PE router and the client interface. This, together with the encapsulation augments proposed in Section 3.2, serves as the way to identify the client interface and configure L2 specific parameters.



#### 4. DESIGN OF THE DATA MODEL

The augmentations defined in this document are organised per scenario, as defined in Section 3. The case described Section 3.4 does not need any further extension of the data model and only requires a more restricted definition on how the data model is used for PE router and client port identification, so no augmentation is implemented for this scenario.

The augmentations implemented are distributed as follows:

- o An extension including RT and RD definition for the L3VPN, following the YANG definitions from BESS-L3VPN. This extension was developed creating a container "ie-profiles" under the VPN Service. All the import-export information can be created and reused for several VPN-Nodes.
  - \* If the "ie-profile" is empty the domain controller should automatically assign RD and RTs. This is not valid for a multi-domain scenario
- o The second augmentation copes with the information from a remote PE not directly under management system supervision. This augmentation does not follow any previously defined model and includes the loopback IP address of the external router.
- o The third augmentation copes with a pseudowire termination under a VPN service. This termination requires the management of the Virtual Circuit Identifier under the VPN service.
- o Access-group-id has been added within the site network access in order to allow associations between interfaces that have similar behaviors. For example, identify two interfaces in dual homing distribution.
- o The last augmentation includes information below layer 3 that is required for the service. In particular, we include information related to clients interface encapsulation and aggregation.

The high-level model structure defined by this document is as shown below:

```
|----- EXAMPLE -----|  
  
module: ietf-l3vpn-ntw  
  +--rw l3vpn-ntw  
    +--rw vpn-profiles  
      | +--rw valid-provider-identifiers
```

```

    +--rw cloud-identifier* [id] {cloud-access}?
    |   +--rw id      string
    +--rw encryption-profile-identifier* [id]
    |   +--rw id      string
    +--rw qos-profile-identifier* [id]
    |   +--rw id      string
    +--rw bfd-profile-identifier* [id]
    |   +--rw id      string
    +--rw routing-profile-identifier* [id]
    |   +--rw id      string
+--rw vpn-services
+--rw vpn-service* [vpn-id]
+--rw vpn-id          svc-id
+--rw customer-name?  string
+--rw vpn-service-topology? identityref
+--rw description?    string
+--rw ie-profiles
+--rw ie-profile* [ie-profile-id]
+--rw ie-profile-id  string
+--rw rd?            rt-types:route-distinguisher
+--rw vpn-targets
+--rw vpn-target* [route-target]
+--rw route-target    rt-types:route-target
t
+--rw route-target-type rt-types:route-target
t-type
+--rw vpn-nodes
+--rw vpn-node* [vpn-node-id ne-id]
+--rw vpn-node-id  string
+--rw description? string
+--rw ne-id        string
+--rw router-id?   inet:ip-address
+--rw address-family? address-family
+--rw node-role?   identityref
+--rw rd?          rt-types:route-distinguisher
+--rw vpn-targets
+--rw vpn-target* [route-target]
+--rw route-target    rt-types:route-target
t
+--rw route-target-type rt-types:route-target
t-type
+--rw status
+--rw admin-enabled?  boolean
+--ro oper-status?    operational-type
+--rw maximum-routes
+--rw address-family* [af]
+--rw af              address-family
+--rw maximum-routes? uint32
+--rw node-ie-profile? -> /l3vpn-ntw/vpn-services/v
pn-service/ie-profiles/ie-profile/ie-profile-id
+--rw cloud-accesses {cloud-access}?
+--rw cloud-access* [cloud-identifier]
+--rw cloud-identifier -> /l3vpn-ntw/vpn-profil
es/valid-provider-identifiers/cloud-identifier/id

```

```

| | | +--rw (list-flavor)?
| | |   +---:(permit-any)
| | |     | ++-rw permit-any?          empty
| | |   +---:(deny-any-except)
| | |     | ++-rw permit-site*        -> /l3vpn-ntw/sites/site
/site-id
| | |   +---:(permit-any-except)
| | |     ++-rw deny-site*            -> /l3vpn-ntw/sites/site
/site-id
| | |   +--rw address-translation
| | |     +--rw nat44
| | |       +--rw enabled?              boolean
| | |       +--rw nat44-customer-address? inet:ipv4-addre
ss
| | | +--rw multicast {multicast}?
| | |   +--rw enabled?                    boolean
| | |   +--rw customer-tree-flavors
| | |     | ++-rw tree-flavor*      identityref
| | |   +--rw rp
| | |     +--rw rp-group-mappings
| | |       +--rw rp-group-mapping* [id]
| | |         +--rw id                uint16
| | |         +--rw provider-managed
| | |           +--rw enabled?          boolean
| | |           +--rw rp-redundancy?    boolean
| | |           +--rw optimal-traffic-delivery? boolean
| | |         +--rw rp-address          inet:ip-address
| | |         +--rw groups
| | |           +--rw group* [id]
| | |             +--rw id              uint16
| | |             +--rw (group-format)
| | |               +---:(singleaddress)
| | |                 | ++-rw group-address? inet:ip-addr
ess
| | |                   +---:(startend)
| | |                     +--rw group-start? inet:ip-addr
ess
| | |                       +--rw group-end?  inet:ip-addr
ess
| | | +--rw rp-discovery
| | |   +--rw rp-discovery-type?  identityref
| | |   +--rw bsr-candidates
| | |     +--rw bsr-candidate-address* inet:ip-address
+--rw carrierscarrier?          boolean {carrierscarrier}?
+--rw extranet-vpns {extranet-vpn}?
+--rw extranet-vpn* [vpn-id]
+--rw vpn-id                    svc-id
+--rw local-sites-role?        identityref
+--rw sites
+--rw site* [site-id]
+--rw site-id                  svc-id
+--rw description?             string
+--rw requested-site-start?     yang:date-and-time
+--rw requested-site-stop?      yang:date-and-time

```

```

+--rw locations
|   +--rw location* [location-id]
|   |   +--rw location-id      svc-id
|   |   +--rw address?         string
|   |   +--rw postal-code?     string
|   |   +--rw state?           string
|   |   +--rw city?            string
|   |   +--rw country-code?    string
+--rw devices
|   +--rw device* [device-id]
|   |   +--rw device-id        svc-id
|   |   +--rw location         -> ../../../../locations/location/lo
cation-id
|   |   +--rw management
|   |   |   +--rw address-family? address-family
|   |   |   +--rw address         inet:ip-address
+--rw site-diversity {site-diversity}?
|   +--rw groups
|   |   +--rw group* [group-id]
|   |   |   +--rw group-id      string
+--rw management
|   +--rw type      identityref
+--rw site-vpn-flavor?      identityref
+--rw maximum-routes
|   +--rw address-family* [af]
|   |   +--rw af          address-family
|   |   +--rw maximum-routes? uint32
+--rw security
|   +--rw authentication
|   +--rw encryption {encryption}?
|   |   +--rw enabled?    boolean
|   |   +--rw layer?      enumeration
+--rw encryption-profile
|   +--rw (profile)?
|   |   +--:(provider-profile)
|   |   |   +--rw profile-name?    -> /l3vpn-ntw/vpn-profil
es/valid-provider-identifiers/encryption-profile-identifier/id
|   |   +--:(customer-profile)
|   |   |   +--rw algorithm?        string
|   |   +--rw (key-type)?
|   |   |   +--:(psk)
|   |   |   +--rw preshared-key?    string
+--rw service
|   +--rw qos {qos}?
|   |   +--rw qos-classification-policy
|   |   |   +--rw rule* [id]
|   |   |   |   +--rw id                                string
|   |   |   |   +--rw (match-type)?
|   |   |   |   |   +--:(match-flow)
|   |   |   |   |   +--rw match-flow

```

						+-rw dscp?	inet:dscp
						+-rw dot1p?	uint8
						+-rw ipv4-src-prefix?	inet:ipv4-p
refix							
						+-rw ipv6-src-prefix?	inet:ipv6-p
refix							
						+-rw ipv4-dst-prefix?	inet:ipv4-p
refix							
						+-rw ipv6-dst-prefix?	inet:ipv6-p
refix							
						+-rw l4-src-port?	inet:port-n
umber							
						+-rw target-sites*	svc-id {tar
get-sites}?							
						+-rw l4-src-port-range	
						+-rw lower-port?	inet:port-numbe
r							
						+-rw upper-port?	inet:port-numbe
r							
						+-rw l4-dst-port?	inet:port-n
umber							
						+-rw l4-dst-port-range	
						+-rw lower-port?	inet:port-numbe
r							
						+-rw upper-port?	inet:port-numbe
r							
						+-rw protocol-field?	union
						+-: (match-application)	
						+-rw match-application?	identityref
						+-rw target-class-id?	string
						+-rw qos-profile	
						+-rw (qos-profile)?	
						+-: (standard)	
						+-rw profile?	-> /l3vpn-ntw/vpn-profile
s/valid-provider-identifiers/qos-profile-identifier/id						+-rw direction?	identityref
						+-: (custom)	
						+-rw classes {qos-custom}?	
						+-rw class* [class-id]	
						+-rw class-id	string
						+-rw direction?	identityref
						+-rw rate-limit?	decimal64
						+-rw latency	
						+-rw (flavor)?	
						+-: (lowest)	
						+-rw use-lowest-latency?	e
mpty							
						+-: (boundary)	
						+-rw latency-boundary?	u
int16							
						+-rw jitter	
						+-rw (flavor)?	
						+-: (lowest)	
						+-rw use-lowest-jitter?	em
pty							
						+-: (boundary)	
						+-rw latency-boundary?	ui
nt32							
						+-rw bandwidth	
						+-rw guaranteed-bw-percent	deci
mal64							
						+-rw end-to-end?	empty
y							

```
|  +--rw carrierscarrier {carrierscarrier}?  
|  |  +--rw signalling-type?  enumeration  
|  +--rw multicast {multicast}?
```

```

    +--rw multicast-site-type?          enumeration
    +--rw multicast-address-family
    |   +--rw ipv4?    boolean {ipv4}?
    |   +--rw ipv6?    boolean {ipv6}?
    +--rw protocol-type?                enumeration
+--rw traffic-protection {fast-reroute}?
|   +--rw enabled?    boolean
+--rw routing-protocols
|   +--rw routing-protocol* [type]
|   |   +--rw type          identityref
|   |   +--rw routing-profiles* [id]
|   |   |   +--rw id        -> /l3vpn-ntw/vpn-profiles/valid-pro
vider-identifiers/routing-profile-identifier/id
|   |   |   +--rw type?    ie-type
|   |   +--rw ospf {rtg-ospf}?
|   |   |   +--rw address-family*    address-family
|   |   |   +--rw area-address        yang:dotted-quad
|   |   |   +--rw metric?             uint16
|   |   |   +--rw mtu?                uint16
|   |   |   +--rw security
|   |   |   |   +--rw auth-key?    string
|   |   |   +--rw sham-links {rtg-ospf-sham-link}?
|   |   |   |   +--rw sham-link* [target-site]
|   |   |   |   |   +--rw target-site    svc-id
|   |   |   |   |   +--rw metric?      uint16
|   |   +--rw bgp {rtg-bgp}?
|   |   |   +--rw autonomous-system    uint32
|   |   |   +--rw address-family*      address-family
|   |   |   +--rw neighbor?            inet:ip-address
|   |   |   +--rw multihop?            uint8
|   |   |   +--rw security
|   |   |   |   +--rw auth-key?    string
|   |   +--rw static
|   |   |   +--rw cascaded-lan-prefixes
|   |   |   |   +--rw ipv4-lan-prefixes* [lan next-hop] {ipv4}?
|   |   |   |   |   +--rw lan          inet:ipv4-prefix
|   |   |   |   |   +--rw lan-tag?    string
|   |   |   |   |   +--rw next-hop    inet:ipv4-address
|   |   |   |   +--rw ipv6-lan-prefixes* [lan next-hop] {ipv6}?
|   |   |   |   |   +--rw lan          inet:ipv6-prefix
|   |   |   |   |   +--rw lan-tag?    string
|   |   |   |   |   +--rw next-hop    inet:ipv6-address
|   |   +--rw rip {rtg-rip}?
|   |   |   +--rw address-family*      address-family
|   |   +--rw vrrp {rtg-vrrp}?
|   |   |   +--rw address-family*      address-family
+--ro actual-site-start?                yang:date-and-time
+--ro actual-site-stop?                 yang:date-and-time
+--rw site-bearers

```

```

    +--rw bearer* [bearer-id]
    |   +--rw bearer-id      string
    |   +--rw BearerType?   identityref
    |   +--rw ne-id?        string
    |   +--rw port-id?      string
    |   +--rw lag-id?       string
+--rw site-network-accesses
  +--rw site-network-access* [site-network-access-id]
  |   +--rw site-network-access-id      svc-id
  |   +--rw description?                string
  |   +--rw status
  |   |   +--rw admin-enabled?   boolean
  |   |   +--ro oper-status?     operational-type
  |   +--rw site-network-access-type? identityref
  +--rw (location-flavor)
  |   +--:(location)
  |   |   +--rw location-reference?  -> ../../../../locatio
ns/location/location-id
  |   +--:(device)
  |   |   +--rw device-reference?    -> ../../../../devices
/device/device-id
  +--rw access-diversity {site-diversity}?
  |   +--rw groups
  |   |   +--rw group* [group-id]
  |   |   |   +--rw group-id      string
  |   +--rw constraints
  |   |   +--rw constraint* [constraint-type]
  |   |   |   +--rw constraint-type identityref
  |   |   |   +--rw target
  |   |   |   |   +--rw (target-flavor)?
  |   |   |   |   |   +--:(id)
  |   |   |   |   |   |   +--rw group* [group-id]
  |   |   |   |   |   |   |   +--rw group-id      string
  |   |   |   |   |   +--:(all-accesses)
  |   |   |   |   |   |   +--rw all-other-accesses? empty
  |   |   |   |   +--:(all-groups)
  |   |   |   |   |   +--rw all-other-groups?      empty
  +--rw bearer
  |   +--rw requested-type {requested-type}?
  |   |   +--rw requested-type?   string
  |   |   +--rw strict?           boolean
  |   +--rw always-on?            boolean {always-on}?
  |   +--rw bearer-reference?     string {bearer-reference
}?
  +--rw connection
  |   +--rw encapsulation-type?   identityref
  |   +--rw tagged-interface
  |   |   +--rw type?             identityref
  |   |   +--rw dot1q-vlan-tagged {dot1q}?
  |   |   |   +--rw tag-type?     identityref
  |   |   |   +--rw cvlan-id?     uint16

```



```

| | | +--rw priority-tagged
| | | | +---rw tag-type? identityref
+--rw qinq {qinq}?
| | | | +---rw tag-type? identityref
| | | | +---rw svlan-id uint16
| | | | +---rw cvlan-id uint16
+--rw qinany {qinany}?
| | | | +---rw tag-type? identityref
| | | | +---rw svlan-id uint16
+--rw vxlan {vxlan}?
| | | | +---rw vni-id uint32
| | | | +---rw peer-mode? identityref
| | | | +---rw peer-list* [peer-ip]
| | | | | +---rw peer-ip inet:ip-address
+--rw pseudowire
| | | | +---rw vcid? uint32
+--rw ip-connection
| | | +--rw ipv4 {ipv4}?
| | | | +---rw address-allocation-type? identityref
| | | | +--rw provider-dhcp
| | | | | +---rw provider-address? ine
t:ipv4-address
| | | | +---rw prefix-length? uin
t8
| | | | +--rw (address-assign)?
| | | | | +---:(number)
| | | | | | +---rw number-of-dynamic-address? uin
t16
| | | | | +---:(explicit)
| | | | | | +--rw customer-addresses
| | | | | | | +--rw address-group* [group-id]
| | | | | | | +---rw group-id string
| | | | | | +--rw start-address? inet:ipv4
-address
| | | | | | +---rw end-address? inet:ipv4
-address
| | | +--rw dhcp-relay
| | | | +---rw provider-address? inet:ipv4-add
ress
| | | | +---rw prefix-length? uint8
| | | | +--rw customer-dhcp-servers
| | | | | +---rw server-ip-address* inet:ipv4-addr
ess
| | | +--rw addresses
| | | | +---rw provider-address? inet:ipv4-address
| | | | +---rw customer-address? inet:ipv4-address
| | | | +---rw prefix-length? uint8
+--rw ipv6 {ipv6}?
| | | +--rw address-allocation-type? identityref
| | | +--rw provider-dhcp
| | | | +---rw provider-address? ine
t:ipv6-address
| | | | +---rw prefix-length? uin
t8
| | | | +--rw (address-assign)?
| | | | | +---:(number)
| | | | | | +---rw number-of-dynamic-address? uin
t16

```

```

+---:(explicit)
+--rw customer-addresses
+--rw address-group* [group-id]
+--rw group-id          string
+--rw start-address?    inet:ipv6
- address
+--rw end-address?      inet:ipv6
- address
+--rw dhcp-relay
+--rw provider-address?  inet:ipv6-add
ress
+--rw prefix-length?     uint8
+--rw customer-dhcp-servers
+--rw server-ip-address* inet:ipv6-addr
ess
+--rw addresses
+--rw provider-address?  inet:ipv6-address
+--rw customer-address?  inet:ipv6-address
+--rw prefix-length?     uint8
+--rw oam
+--rw bfd {bfd}?
+--rw enabled?           boolean
+--rw (holdtime)?
+--:(fixed)
+--rw fixed-value?       uint32
+--:(profile)
+--rw profile-name?      -> /l3vpn-ntw/vpn-
n-profiles/valid-provider-identifiers/bfd-profile-identifier/id
+--rw security
+--rw authentication
+--rw encryption {encryption}?
+--rw enabled?           boolean
+--rw layer?             enumeration
+--rw encryption-profile
+--rw (profile)?
+--:(provider-profile)
+--rw profile-name?      -> /l3vpn-ntw/vpn-
profiles/valid-provider-identifiers/encryption-profile-identifier/id
+--:(customer-profile)
+--rw algorithm?         string
+--rw (key-type)?
+--:(psk)
+--rw preshared-key?     string
+--rw service
+--rw svc-input-bandwidth  uint64
+--rw svc-output-bandwidth uint64
+--rw svc-mtu              uint16
+--rw qos {qos}?
+--rw qos-classification-policy
+--rw rule* [id]
+--rw id                  string
+--rw (match-type)?
+--:(match-flow)
+--rw match-flow

```

dscp						+--rw dscp?	inet:
						+--rw dot1p?	uint8
ipv4-prefix						+--rw ipv4-src-prefix?	inet:
ipv6-prefix						+--rw ipv6-src-prefix?	inet:
ipv4-prefix						+--rw ipv4-dst-prefix?	inet:
ipv6-prefix						+--rw ipv6-dst-prefix?	inet:
port-number						+--rw l4-src-port?	inet:
d {target-sites}?						+--rw target-sites*	svc-i
						+--rw l4-src-port-range	
-number						+--rw lower-port?	inet:port
-number						+--rw upper-port?	inet:port
port-number						+--rw l4-dst-port?	inet:
						+--rw l4-dst-port-range	
-number						+--rw lower-port?	inet:port
-number						+--rw upper-port?	inet:port
						+--rw protocol-field?	union
						+--:(match-application)	
ref						+--rw match-application?	identity
						+--rw target-class-id?	string
						+--rw qos-profile	
						+--rw (qos-profile)?	
						+--:(standard)	
						+--rw profile?	-> /l3vpn-ntw/vpn-p
rofiles/valid-provider-identifiers/qos-profile-identifier/id						+--rw direction?	identityref
						+--:(custom)	
						+--rw classes {qos-custom}?	
						+--rw class* [class-id]	
						+--rw class-id	string
						+--rw direction?	identityref
						+--rw rate-limit?	decimal64
						+--rw latency	
						+--rw (flavor)?	
						+--:(lowest)	
y? empty						+--rw use-lowest-latenc	
						+--:(boundary)	
uint16						+--rw latency-boundary?	
						+--rw jitter	
						+--rw (flavor)?	
						+--:(lowest)	
? empty						+--rw use-lowest-jitter	
						+--:(boundary)	
uint32						+--rw latency-boundary?	
						+--rw bandwidth	
decimal64						+--rw guaranteed-bw-percent	

```
empty          | |          +--rw end-to-end?
               | +--rw carrierscarrier {carrierscarrier}?
               | | +--rw signalling-type?  enumeration
               | +--rw multicast {multicast}?
```

```

|         +--rw multicast-site-type?          enumeration
|         +--rw multicast-address-family
|         |   +--rw ipv4?    boolean {ipv4}?
|         |   +--rw ipv6?    boolean {ipv6}?
|         +--rw protocol-type?                enumeration
+--rw routing-protocols
|   +--rw routing-protocol* [type]
|   +--rw type                identityref
|   +--rw routing-profiles* [id]
|   |   +--rw id              -> /l3vpn-ntw/vpn-profiles/val
id-provider-identifiers/routing-profile-identifier/id
|   |   +--rw type?          ie-type
+--rw ospf {rtg-ospf}?
|   +--rw address-family*    address-family
|   +--rw area-address        yang:dotted-quad
|   +--rw metric?            uint16
|   +--rw mtu?               uint16
|   +--rw security
|   |   +--rw auth-key?      string
+--rw sham-links {rtg-ospf-sham-link}?
|   +--rw sham-link* [target-site]
|   |   +--rw target-site    svc-id
|   |   +--rw metric?       uint16
+--rw bgp {rtg-bgp}?
|   +--rw autonomous-system  uint32
|   +--rw address-family*    address-family
|   +--rw neighbor?          inet:ip-address
|   +--rw multihop?          uint8
|   +--rw security
|   |   +--rw auth-key?      string
+--rw static
|   +--rw cascaded-lan-prefixes
|   |   +--rw ipv4-lan-prefixes* [lan next-hop] {
ipv4}?
|   |   |   +--rw lan          inet:ipv4-prefix
|   |   |   +--rw lan-tag?     string
|   |   |   +--rw next-hop     inet:ipv4-address
|   |   +--rw ipv6-lan-prefixes* [lan next-hop] {
ipv6}?
|   |   |   +--rw lan          inet:ipv6-prefix
|   |   |   +--rw lan-tag?     string
|   |   |   +--rw next-hop     inet:ipv6-address
|   +--rw rip {rtg-rip}?
|   |   +--rw address-family*  address-family
+--rw vrrp {rtg-vrrp}?
|   +--rw address-family*      address-family
+--rw availability
|   +--rw access-priority?     uint32
+--rw node-id?                -> /l3vpn-ntw/vpn-s
services/vpn-service/vpn-nodes/vpn-node/vpn-node-id
+--rw service-id?            -> /l3vpn-ntw/vpn-s
services/vpn-service/vpn-id
+--rw access-group-id?        yang:uuid

```

Figure 3

## 5. YANG MODULE

```
|----- EXAMPLE -----|
<CODE BEGINS>file "ietf-l3vpn-ntw@2019-09-13.YANG"
module ietf-l3vpn-ntw {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-l3vpn-ntw";
  prefix l3vpn-ntw;
  import ietf-inet-types {
    prefix inet;
  }
  import ietf-yang-types {
    prefix yang;
  }
  import ietf-netconf-acm {
    prefix nacm;
  }
  import ietf-routing-types {
    prefix rt-types;
  }
  organization
    "Individual draft";
  contact
    "Currently discussed in WG List: <mailto:opsawg@ietf.org>
    Editor: Oscar Gonzalez de Dios
      <mailto:oscar.gonzalezdedios@telefonica.com>";

  description
    "This YANG module defines a generic network-oriented model
    for the configuration of Layer 3 VPNs.
    Copyright (c) 2019 IETF Trust and the persons identified as
    authors of the code. All rights reserved.

    Redistribution and use in source and binary forms, with or
    without modification, is permitted pursuant to, and subject to
    the license terms contained in, the Simplified BSD License set
    forth in Section 4.c of the IETF Trust's Legal Provisions
    Relating to IETF Documents
    (https://trustee.ietf.org/license-info).

    This version of this YANG module is part of RFC XXXX
    (https://www.rfc-editor.org/info/rfcXXXX); see the RFC itself
    for full legal notices.

    The key words 'MUST', 'MUST NOT', 'REQUIRED', 'SHALL', 'SHALL
```

NOT', 'SHOULD', 'SHOULD NOT', 'RECOMMENDED', 'NOT RECOMMENDED', 'MAY', and 'OPTIONAL' in this document are to be interpreted as described in BCP 14 (RFC 2119) (RFC 8174) when, and only when, they appear in all capitals, as shown here.";

```
revision 2019-09-24 {
  description
    "Initial document. The document as a whole is based on L3SM
    module, defined in RFC 8299, modified to fit the requirements
    of the platforms at the network layer.";
  reference
    "RFC 8049.";
}
/* Features */
feature cloud-access {
  description
    "Allows the VPN to connect to a CSP.";
}
feature multicast {
  description
    "Enables multicast capabilities in a VPN.";
}
feature ipv4 {
  description
    "Enables IPv4 support in a VPN.";
}
feature ipv6 {
  description
    "Enables IPv6 support in a VPN.";
}
feature lan-tag {
  description
    "Enables LAN Tag support in a VPN Policy filter.";
}
feature carrierscarrier {
  description
    "Enables support of CsC.";
}
feature extranet-vpn {
  description
    "Enables support of extranet VPNs.";
}
feature site-diversity {
  description
    "Enables support of site diversity constraints.";
}
feature encryption {
  description
```

```
"Enables support of encryption.";
}
feature qos {
  description
    "Enables support of classes of services.";
}
feature qos-custom {
  description
    "Enables support of the custom QoS profile.";
}
feature rtg-bgp {
  description
    "Enables support of the BGP routing protocol.";
}
feature rtg-rip {
  description
    "Enables support of the RIP routing protocol.";
}
feature rtg-ospf {
  description
    "Enables support of the OSPF routing protocol.";
}
feature rtg-ospf-sham-link {
  description
    "Enables support of OSPF sham links.";
}
feature rtg-vrrp {
  description
    "Enables support of the VRRP routing protocol.";
}
feature fast-reroute {
  description
    "Enables support of Fast Reroute.";
}
feature bfd {
  description
    "Enables support of BFD.";
}
feature always-on {
  description
    "Enables support of the 'always-on' access constraint.";
}
feature requested-type {
  description
    "Enables support of the 'requested-type' access constraint.";
}
feature bearer-reference {
  description
```



```
    "Enables support of the 'bearer-reference' access constraint.";
}
feature target-sites {
    description
    "Enables support of the 'target-sites' match flow parameter.";
}
feature input-bw {
    description
    "Enables support of the 'input-bw' limit.";
}
feature dot1q {
    description
    "Enables support of the 'dot1q' encapsulation.";
}
feature qinq {
    description
    "Enables support of the 'qinq' encapsulation.";
}
feature qinany {
    description
    "Enables support of the 'qinany' encapsulation.";
}
feature vxlan {
    description
    "Enables support of the 'vxlan' encapsulation.";
}
/* Typedefs */
typedef svc-id {
    type string;
    description
    "Defines a type of service component identifier.";
}
typedef template-id {
    type string;
    description
    "Defines a type of service template identifier.";
}
typedef address-family {
    type enumeration {
        enum ipv4 {
            description
            "IPv4 address family.";
        }
        enum ipv6 {
            description
            "IPv6 address family.";
        }
    }
}
```

```
description
  "Defines a type for the address family.";
}

typedef ie-type {
  type enumeration {
    enum "import" {
      value 0;
      description "Import routing profile.";
    }
    enum "export" {
      value 1;
      description "Export routing profile";
    }
    enum "both" {
      value 2;
      description "Import/Export routing profile";
    }
  }
  description
  "Defines Import-Export routing profiles.
  Those are able to be reused between vpn-nodes";
}

typedef operational-type {
  type enumeration {
    enum "up" {
      value 0;
      description "Operational status UP.";
    }
    enum "down" {
      value 1;
      description "Operational status DOWN";
    }
    enum "unknown" {
      value 2;
      description "Operational status UNKNOWN";
    }
  }
  description
  "This is a read-only attribute used to determine the
  status of a particular element";
}

/* Identities */
identity site-network-access-type {
  description
  "Base identity for site-network-access type.";
}
```

```
}
identity point-to-point {
  base site-network-access-type;
  description
    "Identity for point-to-point connection.";
}
/* Extension */
identity pseudowire {
  base site-network-access-type;
  description
    "Identity for pseudowire connection.";
}
/* End of Extension */
identity multipoint {
  base site-network-access-type;
  description
    "Identity for multipoint connection.
    Example: Ethernet broadcast segment.";
}
identity placement-diversity {
  description
    "Base identity for site placement constraints.";
}
identity bearer-diverse {
  base placement-diversity;
  description
    "Identity for bearer diversity.
    The bearers should not use common elements.";
}
identity pe-diverse {
  base placement-diversity;
  description
    "Identity for PE diversity.";
}
identity pop-diverse {
  base placement-diversity;
  description
    "Identity for POP diversity.";
}
identity linecard-diverse {
  base placement-diversity;
  description
    "Identity for linecard diversity.";
}
identity same-pe {
  base placement-diversity;
  description
    "Identity for having sites connected on the same PE.";
```

```
}
identity same-bearer {
  base placement-diversity;
  description
    "Identity for having sites connected using the same bearer.";
}
identity customer-application {
  description
    "Base identity for customer application.";
}
identity web {
  base customer-application;
  description
    "Identity for Web application (e.g., HTTP, HTTPS).";
}
identity mail {
  base customer-application;
  description
    "Identity for mail application.";
}
identity file-transfer {
  base customer-application;
  description
    "Identity for file transfer application (e.g., FTP, SFTP).";
}
identity database {
  base customer-application;
  description
    "Identity for database application.";
}
identity social {
  base customer-application;
  description
    "Identity for social-network application.";
}
identity games {
  base customer-application;
  description
    "Identity for gaming application.";
}
identity p2p {
  base customer-application;
  description
    "Identity for peer-to-peer application.";
}
identity network-management {
  base customer-application;
  description
```

```
"Identity for management application
(e.g., Telnet, syslog, SNMP).";
}
identity voice {
  base customer-application;
  description
  "Identity for voice application.";
}
identity video {
  base customer-application;
  description
  "Identity for video conference application.";
}
identity embb {
  base customer-application;
  description
  "Identity for an enhanced Mobile Broadband (eMBB)
  application. Note that an eMBB application demands
  network performance with a wide variety of
  characteristics, such as data rate, latency,
  loss rate, reliability, and many other parameters.";
}
identity urllc {
  base customer-application;
  description
  "Identity for an Ultra-Reliable and Low Latency
  Communications (URLLC) application. Note that a
  URLLC application demands network performance
  with a wide variety of characteristics, such as latency,
  reliability, and many other parameters.";
}
identity mmhc {
  base customer-application;
  description
  "Identity for a massive Machine Type
  Communications (mMTC) application. Note that an
  mMTC application demands network performance
  with a wide variety of characteristics, such as data
  rate, latency, loss rate, reliability, and many
  other parameters.";
}
identity site-vpn-flavor {
  description
  "Base identity for the site VPN service flavor.";
}
identity site-vpn-flavor-single {
  base site-vpn-flavor;
  description
```

```
"Base identity for the site VPN service flavor.
Used when the site belongs to only one VPN.";
}
identity site-vpn-flavor-multi {
  base site-vpn-flavor;
  description
    "Base identity for the site VPN service flavor.
    Used when a logical connection of a site
    belongs to multiple VPNs.";
}
identity site-vpn-flavor-sub {
  base site-vpn-flavor;
  description
    "Base identity for the site VPN service flavor.
    Used when a site has multiple logical connections.
    Each connection may belong to different multiple VPNs.";
}
identity site-vpn-flavor-nni {
  base site-vpn-flavor;
  description
    "Base identity for the site VPN service flavor.
    Used to describe an NNI option A connection.";
}
identity management {
  description
    "Base identity for site management scheme.";
}
identity co-managed {
  base management;
  description
    "Base identity for co-managed site.";
}
identity customer-managed {
  base management;
  description
    "Base identity for customer-managed site.";
}
identity provider-managed {
  base management;
  description
    "Base identity for provider-managed site.";
}
identity address-allocation-type {
  description
    "Base identity for address-allocation-type for PE-CE link.";
}
identity provider-dhcp {
  base address-allocation-type;
```

```
    description
    "Provider network provides DHCP service to customer.";
}
identity provider-dhcp-relay {
    base address-allocation-type;
    description
    "Provider network provides DHCP relay service to customer.";
}
identity provider-dhcp-slaac {
    base address-allocation-type;
    description
    "Provider network provides DHCP service to customer,
    as well as SLAAC.";
}
identity static-address {
    base address-allocation-type;
    description
    "Provider-to-customer addressing is static.";
}
identity slaac {
    base address-allocation-type;
    description
    "Use IPv6 SLAAC.";
}
identity site-role {
    description
    "Base identity for site type.";
}
identity any-to-any-role {
    base site-role;
    description
    "Site in an any-to-any IP VPN.";
}
identity spoke-role {
    base site-role;
    description
    "Spoke site in a Hub-and-Spoke IP VPN.";
}
identity hub-role {
    base site-role;
    description
    "Hub site in a Hub-and-Spoke IP VPN.";
}
identity vpn-topology {
    description
    "Base identity for VPN topology.";
}
identity any-to-any {
```

```
base vpn-topology;
description
  "Identity for any-to-any VPN topology.";
}
identity hub-spoke {
  base vpn-topology;
  description
    "Identity for Hub-and-Spoke VPN topology.";
}
identity hub-spoke-disjoint {
  base vpn-topology;
  description
    "Identity for Hub-and-Spoke VPN topology
    where Hubs cannot communicate with each other.";
}
identity multicast-tree-type {
  description
    "Base identity for multicast tree type.";
}
identity ssm-tree-type {
  base multicast-tree-type;
  description
    "Identity for SSM tree type.";
}
identity asm-tree-type {
  base multicast-tree-type;
  description
    "Identity for ASM tree type.";
}
identity bidir-tree-type {
  base multicast-tree-type;
  description
    "Identity for bidirectional tree type.";
}
identity multicast-rp-discovery-type {
  description
    "Base identity for RP discovery type.";
}
identity auto-rp {
  base multicast-rp-discovery-type;
  description
    "Base identity for Auto-RP discovery type.";
}
identity static-rp {
  base multicast-rp-discovery-type;
  description
    "Base identity for static type.";
}
```



```
identity bsr-rp {
  base multicast-rp-discovery-type;
  description
    "Base identity for BSR discovery type.";
}
identity routing-protocol-type {
  description
    "Base identity for routing protocol type.";
}
identity ospf {
  base routing-protocol-type;
  description
    "Identity for OSPF protocol type.";
}
identity bgp {
  base routing-protocol-type;
  description
    "Identity for BGP protocol type.";
}
identity static {
  base routing-protocol-type;
  description
    "Identity for static routing protocol type.";
}
identity rip {
  base routing-protocol-type;
  description
    "Identity for RIP protocol type.";
}
identity vrrp {
  base routing-protocol-type;
  description
    "Identity for VRRP protocol type.
    This is to be used when LANs are directly connected
    to PE routers.";
}
identity direct {
  base routing-protocol-type;
  description
    "Identity for direct protocol type.";
}
identity protocol-type {
  description
    "Base identity for protocol field type.";
}
identity tcp {
  base protocol-type;
  description
```

```
"TCP protocol type.";
}
identity udp {
  base protocol-type;
  description
    "UDP protocol type.";
}

identity icmp {
  base protocol-type;
  description
    "ICMP protocol type.";
}
identity icmp6 {
  base protocol-type;
  description
    "ICMPv6 protocol type.";
}
identity gre {
  base protocol-type;
  description
    "GRE protocol type.";
}
identity ipip {
  base protocol-type;
  description
    "IP-in-IP protocol type.";
}
identity hop-by-hop {
  base protocol-type;
  description
    "Hop-by-Hop IPv6 header type.";
}
identity routing {
  base protocol-type;
  description
    "Routing IPv6 header type.";
}
identity esp {
  base protocol-type;
  description
    "ESP header type.";
}
identity ah {
  base protocol-type;
  description
    "AH header type.";
}
```

```
identity vpn-policy-filter-type {
  description
    "Base identity for VPN Policy filter type.";
}
identity ipv4 {
  base vpn-policy-filter-type;
  description
    "Identity for IPv4 Prefix filter type.";
}
identity ipv6 {
  base vpn-policy-filter-type;
  description
    "Identity for IPv6 Prefix filter type.";
}
identity lan {
  base vpn-policy-filter-type;
  description
    "Identity for LAN Tag filter type.";
}

identity qos-profile-direction {
  description
    "Base identity for QoS profile direction.";
}

identity site-to-wan {
  base qos-profile-direction;
  description
    "Identity for Site-to-WAN direction.";
}
identity wan-to-site {
  base qos-profile-direction;
  description
    "Identity for WAN-to-Site direction.";
}
identity both {
  base qos-profile-direction;
  description
    "Identity for both WAN-to-Site direction
    and Site-to-WAN direction.";
}

/* Extended Identities */

identity encapsulation-type {
  description
    "Identity for the encapsulation type.";
}
```

```
identity untagged-int {
    base encapsulation-type;
    description
        "Identity for Ethernet type.";
}

identity tagged-int {
    base encapsulation-type;
    description
        "Identity for the VLAN type.";
}

identity eth-inf-type {
    description
        "Identity of the Ethernet interface type.";
}

identity tagged {
    base eth-inf-type;
    description
        "Identity of the tagged interface type.";
}

identity untagged {
    base eth-inf-type;
    description
        "Identity of the untagged interface type.";
}

identity lag {
    base eth-inf-type;
    description
        "Identity of the LAG interface type.";
}

identity bearer-inf-type {
    description
        "Identity for the bearer interface type.";
}

identity port-id {
    base bearer-inf-type;
    description
        "Identity for the priority-tagged interface.";
}

identity lag-id {
    base bearer-inf-type;
    description
```

```
    "Identity for the priority-tagged interface.";
}

identity tagged-inf-type {
    description
        "Identity for the tagged interface type.";
}

identity priority-tagged {
    base tagged-inf-type;
    description
        "Identity for the priority-tagged interface.";
}

identity qinq {
    base tagged-inf-type;
    description
        "Identity for the QinQ tagged interface.";
}

identity dot1q {
    base tagged-inf-type;
    description
        "Identity for the dot1Q VLAN tagged interface.";
}

identity qinany {
    base tagged-inf-type;
    description
        "Identity for the QinAny tagged interface.";
}

identity vxlan {
    base tagged-inf-type;
    description
        "Identity for the VXLAN tagged interface.";
}

identity tag-type {
    description
        "Base identity from which all tag types are derived.";
}

identity c-vlan {
    base tag-type;
    description
        "A CVLAN tag, normally using the 0x8100 Ethertype.";
}
```

```
identity s-vlan {
    base tag-type;
    description
        "An SVLAN tag.";
}

identity c-s-vlan {
    base tag-type;
    description
        "Using both a CVLAN tag and an SVLAN tag.";
}

identity vxlan-peer-mode {
    description
        "Base identity for the VXLAN peer mode.";
}

identity static-mode {
    base vxlan-peer-mode;
    description
        "Identity for VXLAN access in the static mode.";
}

identity bgp-mode {
    base vxlan-peer-mode;
    description
        "Identity for VXLAN access by BGP EVPN learning.";
}

identity bw-direction {
    description
        "Identity for the bandwidth direction.";
}

identity input-bw {
    base bw-direction;
    description
        "Identity for the input bandwidth.";
}

identity output-bw {
    base bw-direction;
    description
        "Identity for the output bandwidth.";
}

identity bw-type {
    description
```

```
    "Identity of the bandwidth type.";
}

identity bw-per-cos {
    base bw-type;
    description
        "Bandwidth is per CoS.";
}

identity bw-per-port {
    base bw-type;
    description
        "Bandwidth is per site network access.";
}

identity bw-per-site {
    base bw-type;
    description
        "Bandwidth is per site.  It is applicable to
        all the site network accesses within the site.";
}

identity bw-per-svc {
    base bw-type;
    description
        "Bandwidth is per VPN service.";
}

/* Groupings */
grouping vpn-service-cloud-access {
    container cloud-accesses {
        if-feature cloud-access;
        list cloud-access {
            key cloud-identifier;
            leaf cloud-identifier {
                type leafref {
                    path "/l3vpn-ntw/vpn-profiles/" +
                        "valid-provider-identifiers/cloud-identifier/id";
                }
            }
            description
                "Identification of cloud service.
                Local administration meaning.";
        }
        choice list-flavor {
            case permit-any {
                leaf permit-any {
                    type empty;
                    description
```

```
    "Allows all sites.";
  }
}
case deny-any-except {
  leaf-list permit-site {
    type leafref {
      path "/l3vpn-ntw/sites/site/site-id";
    }
    description
      "Site ID to be authorized.";
  }
}
case permit-any-except {
  leaf-list deny-site {
    type leafref {
      path "/l3vpn-ntw/sites/site/site-id";
    }
    description
      "Site ID to be denied.";
  }
}
description
  "Choice for cloud access policy.  By
  default, all sites in the IP VPN MUST
  be authorized to access the cloud.";
}
container address-translation {
  container nat44 {
    leaf enabled {
      type boolean;
      default false;
      description
        "Controls whether or not Network address
        translation from IPv4 to IPv4 (NAT44)
        [RFC3022] is required.";
    }
  }
  leaf nat44-customer-address {
    type inet:ipv4-address;
    description
      "Address to be used for network address
      translation from IPv4 to IPv4.  This is
      to be used if the customer is providing
      the IPv4 address.  If the customer address
      is not set, the model assumes that the
      provider will allocate the address.";
  }
  description
    "IPv4-to-IPv4 translation.";
```



```
    }
    description
    "Container for NAT.";
  }
  description
  "Cloud access configuration.";
}
description
"Container for cloud access configurations.";
}
description
"Grouping for VPN cloud definition.";
}
grouping multicast-rp-group-cfg {
  choice group-format {
    mandatory true;
    case singleaddress {
      leaf group-address {
        type inet:ip-address;
        description
        "A single multicast group address.";
      }
    }
    case startend {
      leaf group-start {
        type inet:ip-address;
        description
        "The first multicast group address in
        the multicast group address range.";
      }
      leaf group-end {
        type inet:ip-address;
        description
        "The last multicast group address in
        the multicast group address range.";
      }
    }
  }
  description
  "Choice for multicast group format.";
}
description
"This grouping defines multicast group or
multicast groups for RP-to-group mapping.";
}
grouping vpn-service-multicast {
  container multicast {
    if-feature multicast;
    leaf enabled {
```

```
    type boolean;
    default false;
    description
    "Enables multicast.";
}
container customer-tree-flavors {
  leaf-list tree-flavor {
    type identityref {
      base multicast-tree-type;
    }
    description
    "Type of tree to be used.";
  }
  description
  "Type of trees used by customer.";
}
container rp {
  container rp-group-mappings {
    list rp-group-mapping {
      key id;
      leaf id {
        type uint16;
        description
        "Unique identifier for the mapping.";
      }
    }
    container provider-managed {
      leaf enabled {
        type boolean;
        default false;
        description
        "Set to true if the Rendezvous Point (RP)
        must be a provider-managed node. Set to false
        if it is a customer-managed node.";
      }
      leaf rp-redundancy {
        type boolean;
        default false;
        description
        "If true, a redundancy mechanism for the RP
        is required.";
      }
    }
    leaf optimal-traffic-delivery {
      type boolean;
      default false;
      description
      "If true, the SP must ensure that
      traffic uses an optimal path. An SP may use
      Anycast RP or RP-tree-to-SPT switchover";
    }
  }
}
```

```
    architectures.";
  }
  description
    "Parameters for a provider-managed RP.";
}
leaf rp-address {
  when "../provider-managed/enabled = 'false'" {
    description
      "Relevant when the RP is not provider-managed.";
  }
  type inet:ip-address;
  mandatory true;
  description
    "Defines the address of the RP.
    Used if the RP is customer-managed.";
}
container groups {
  list group {
    key id;
    leaf id {
      type uint16;
      description
        "Identifier for the group.";
    }
    uses multicast-rp-group-cfg;
    description
      "List of multicast groups.";
  }
  description
    "Multicast groups associated with the RP.";
}
description
  "List of RP-to-group mappings.";
}
description
  "RP-to-group mappings parameters.";
}
container rp-discovery {
  leaf rp-discovery-type {
    type identityref {
      base multicast-rp-discovery-type;
    }
    default static-rp;
    description
      "Type of RP discovery used.";
  }
}
container bsr-candidates {
  when "derived-from-or-self(../rp-discovery-type, "+
```

```
    "'l3vpn-ntw:bsr-rp') " {
      description
        "Only applicable if discovery type
        is BSR-RP.";
    }
    leaf-list bsr-candidate-address {
      type inet:ip-address;
      description
        "Address of BSR candidate.";
    }
    description
      "Container for List of Customer
      BSR candidate's addresses.";
  }
  description
    "RP discovery parameters.";
}
description
  "RP parameters.";
}
description
  "Multicast global parameters for the VPN service.";
}
description
  "Grouping for multicast VPN definition.";
}
grouping vpn-service-mpls {
  leaf carrierscarrier {
    if-feature carrierscarrier;
    type boolean;
    default false;
    description
      "The VPN is using CsC, and so MPLS is required.";
  }
  description
    "Grouping for MPLS CsC definition.";
}
grouping customer-location-info {
  container locations {
    list location {
      key location-id;
      leaf location-id {
        type svc-id;
        description
          "Identifier for a particular location.";
      }
      leaf address {
        type string;
      }
    }
  }
}
```

```
    description
    "Address (number and street) of the site.";
}
leaf postal-code {
    type string;
    description
    "Postal code of the site.";
}
leaf state {
    type string;
    description
    "State of the site.  This leaf can also be
    used to describe a region for a country that
    does not have states.";
}
leaf city {
    type string;
    description
    "City of the site.";
}
leaf country-code {
    type string {
        pattern '[A-Z]{2}';
    }
    description
    "Country of the site.
    Expressed as ISO ALPHA-2 code.";
}
description
"Location of the site.";
}
description
"List of locations for the site.";
}
description
"This grouping defines customer location parameters.";
}
grouping site-group {
    container groups {
        list group {
            key group-id;
            leaf group-id {
                type string;
                description
                "Group-id the site belongs to.";
            }
            description
            "List of group-ids.";
        }
    }
}
```

```
    }
    description
    "Groups the site or site-network-access belongs to.";
  }
  description
  "Grouping definition to assign
  group-ids to site or site-network-access.";
}
grouping site-diversity {
  container site-diversity {
    if-feature site-diversity;
    uses site-group;
    description
    "Diversity constraint type. All
    site-network-accesses will inherit
    the group values defined here.";
  }
  description
  "This grouping defines site
  diversity parameters.";
}
grouping access-diversity {
  container access-diversity {
    if-feature site-diversity;
    uses site-group;
    container constraints {
      list constraint {
        key constraint-type;
        leaf constraint-type {
          type identityref {
            base placement-diversity;
          }
        }
        description
        "Diversity constraint type.";
      }
    }
    container target {
      choice target-flavor {
        default id;
        case id {
          list group {
            key group-id;
            leaf group-id {
              type string;
              description
              "The constraint will be applied against
              this particular group-id for this site
              network access level.";
            }
          }
        }
      }
    }
  }
}
```

```
        description
        "List of group-ids associated with one specific
        constraint for this site network access level.";
    }
}
case all-accesses {
    leaf all-other-accesses {
        type empty;
        description
        "The constraint will be applied against
        all other site network accesses of this site.";
    }
}
case all-groups {
    leaf all-other-groups {
        type empty;
        description
        "The constraint will be applied against
        all other groups managed by the customer.";
    }
}
description
"Choice for the target flavor definition.";
}
description
"The constraint will be applied against a
Specific target, and the target can be a list
of group-ids, all other site network accesses of
this site, or all other groups managed by the
customer.";
}
description
"List of constraints.";
}
description
"Placement constraints for this site network access.";
}
description
"Diversity parameters.";
}
description
"This grouping defines access diversity parameters.";
}
grouping operational-requirements {
    leaf requested-site-start {
        type yang:date-and-time;
        description
        "Optional leaf indicating requested date and
```

```
        time when the service at a particular site is
        expected to start.";
    }

    leaf requested-site-stop {
        type yang:date-and-time;
        description
            "Optional leaf indicating requested date and
            time when the service at a particular site is
            expected to stop.";
    }
    description
        "This grouping defines some operational
        parameters.";
}
grouping operational-requirements-ops {
    leaf actual-site-start {
        type yang:date-and-time;
        config false;
        description
            "Optional leaf indicating actual date and
            time when the service at a particular site
            actually started.";
    }
    leaf actual-site-stop {
        type yang:date-and-time;
        config false;
        description
            "Optional leaf indicating actual date and
            time when the service at a particular site
            actually stopped.";
    }
    description
        "This grouping defines some operational
        parameters.";
}
grouping flow-definition {
    container match-flow {
        leaf dscp {
            type inet:dscp;
            description
                "DSCP value.";
        }
        leaf dot1p {
            type uint8 {
                range "0..7";
            }
            description
```



```
    "802.1p matching.";
  }
  leaf ipv4-src-prefix {
    type inet:ipv4-prefix;
    description
      "Match on IPv4 src address.";
  }
  leaf ipv6-src-prefix {
    type inet:ipv6-prefix;
    description
      "Match on IPv6 src address.";
  }
  leaf ipv4-dst-prefix {
    type inet:ipv4-prefix;
    description
      "Match on IPv4 dst address.";
  }
  leaf ipv6-dst-prefix {
    type inet:ipv6-prefix;
    description
      "Match on IPv6 dst address.";
  }
  leaf l4-src-port {
    type inet:port-number;
    must "current() < ../l4-src-port-range/lower-port or "+
      "current() > ../l4-src-port-range/upper-port" {
      description
        "If l4-src-port and l4-src-port-range/lower-port and
        upper-port are set at the same time, l4-src-port
        should not overlap with l4-src-port-range.";
    }
    description
      "Match on Layer 4 src port.";
  }
  leaf-list target-sites {
    if-feature target-sites;
    type svc-id;
    description
      "Identify a site as traffic destination.";
  }
  container l4-src-port-range {
    leaf lower-port {
      type inet:port-number;
      description
        "Lower boundary for port.";
    }
    leaf upper-port {
      type inet:port-number;
```

```
    must ". >= ../lower-port" {
      description
        "Upper boundary for port.  If it
         exists, the upper boundary must be
         higher than the lower boundary.";
    }
    description
      "Upper boundary for port.";
  }
  description
    "Match on Layer 4 src port range.  When
     only the lower-port is present, it represents
     a single port.  When both the lower-port and
     upper-port are specified, it implies
     a range inclusive of both values.";
}
leaf l4-dst-port {
  type inet:port-number;
  must "current() < ../l4-dst-port-range/lower-port or "+
       "current() > ../l4-dst-port-range/upper-port" {
    description
      "If l4-dst-port and l4-dst-port-range/lower-port
       and upper-port are set at the same time,
       l4-dst-port should not overlap with
       l4-src-port-range.";
  }
  description
    "Match on Layer 4 dst port.";
}
container l4-dst-port-range {
  leaf lower-port {
    type inet:port-number;
    description
      "Lower boundary for port.";
  }
  leaf upper-port {
    type inet:port-number;
    must ". >= ../lower-port" {
      description
        "Upper boundary must be
         higher than lower boundary.";
    }
    description
      "Upper boundary for port.  If it exists,
       upper boundary must be higher than lower
       boundary.";
  }
  description
```

```
    "Match on Layer 4 dst port range.  When only
    lower-port is present, it represents a single
    port.  When both lower-port and upper-port are
    specified, it implies a range inclusive of both
    values.";
  }
  leaf protocol-field {
    type union {
      type uint8;
      type identityref {
        base protocol-type;
      }
    }
    description
      "Match on IPv4 protocol or IPv6 Next Header field.";
  }
  description
    "Describes flow-matching criteria.";
}
description
  "Flow definition based on criteria.";
}
grouping site-service-basic {
  leaf svc-input-bandwidth {
    type uint64;
    units bps;
    mandatory true;
    description
      "From the customer site's perspective, the service
      input bandwidth of the connection or download
      bandwidth from the SP to the site.";
  }
  leaf svc-output-bandwidth {
    type uint64;
    units bps;
    mandatory true;
    description
      "From the customer site's perspective, the service
      output bandwidth of the connection or upload
      bandwidth from the site to the SP.";
  }
}
leaf svc-mtu {
  type uint16;
  units bytes;
  mandatory true;
  description
    "MTU at service level.  If the service is IP,
    it refers to the IP MTU.  If CsC is enabled,
```

```
    the requested 'svc-mtu' leaf will refer to the
    MPLS MTU and not to the IP MTU.";
}
description
"Defines basic service parameters for a site.";
}
grouping site-protection {
  container traffic-protection {
    if-feature fast-reroute;
    leaf enabled {
      type boolean;
      default false;
      description
        "Enables traffic protection of access link.";
    }
    description
      "Fast Reroute service parameters for the site.";
  }
  description
    "Defines protection service parameters for a site.";
}
grouping site-service-mpls {
  container carrierscarrier {
    if-feature carrierscarrier;
    leaf signalling-type {
      type enumeration {
        enum ldp {
          description
            "Use LDP as the signalling protocol
            between the PE and the CE. In this case,
            an IGP routing protocol must also be activated.";
        }
        enum bgp {
          description
            "Use BGP (as per RFC 8277) as the signalling protocol
            between the PE and the CE.
            In this case, BGP must also be configured as
            the routing protocol.";
        }
      }
    }
    default bgp;
    description
      "MPLS signalling type.";
  }
  description
    "This container is used when the customer provides
    MPLS-based services. This is only used in the case
    of CsC (i.e., a customer builds an MPLS service using
```

```
    an IP VPN to carry its traffic).";
  }
  description
    "Defines MPLS service parameters for a site.";
}
grouping site-service-qos-profile {
  container qos {
    if-feature qos;
    container qos-classification-policy {
      list rule {
        key id;
        ordered-by user;
        leaf id {
          type string;
          description
            "A description identifying the
             qos-classification-policy rule.";
        }
        choice match-type {
          default match-flow;
          case match-flow {
            uses flow-definition;
          }
          case match-application {
            leaf match-application {
              type identityref {
                base customer-application;
              }
              description
                "Defines the application to match.";
            }
          }
        }
        description
          "Choice for classification.";
      }
      leaf target-class-id {
        type string;
        description
          "Identification of the class of service.
           This identifier is internal to the administration.";
      }
      description
        "List of marking rules.";
    }
    description
      "Configuration of the traffic classification policy.";
  }
  container qos-profile {
```

```
choice qos-profile {
  description
  "Choice for QoS profile.
  Can be standard profile or customized profile.";
  case standard {
    description
    "Standard QoS profile.";
    leaf profile {
      type leafref {
        path "/l3vpn-ntw/vpn-profiles/valid-provider-identifiers"+
          "/qos-profile-identifier/id";
      }
      description
      "QoS profile to be used.";
    }
    leaf direction {
      type identityref {
        base qos-profile-direction;
        default both;
        description
        "The direction to which the QoS profile
        is applied.";
      }
    }
  }
  case custom {
    description
    "Customized QoS profile.";
    container classes {
      if-feature qos-custom;
      list class {
        key class-id;
        leaf class-id {
          type string;
          description
          "Identification of the class of service.
          This identifier is internal to the
          administration.";
        }
        leaf direction {
          type identityref {
            base qos-profile-direction;
          }
          default both;
          description
          "The direction to which the QoS profile
          is applied.";
        }
        leaf rate-limit {
```

```
        type decimal64 {
            fraction-digits 5;
            range "0..100";
        }

        units percent;
        description
            "To be used if the class must be rate-limited.
            Expressed as percentage of the service
            bandwidth.";
    }

    container latency {
        choice flavor {
            case lowest {
                leaf use-lowest-latency {
                    type empty;
                    description
                        "The traffic class should use the path with the
                        lowest latency.";
                }
            }
            case boundary {
                leaf latency-boundary {
                    type uint16;
                    units msec;
                    default 400;
                    description
                        "The traffic class should use a path with a
                        defined maximum latency.";
                }
            }
        }
        description
            "Latency constraint on the traffic class.";
    }
    description
        "Latency constraint on the traffic class.";
}

container jitter {
    choice flavor {
        case lowest {
            leaf use-lowest-jitter {
                type empty;
                description
                    "The traffic class should use the path with the
                    lowest jitter.";
            }
        }
        case boundary {
```

```
    leaf latency-boundary {
      type uint32;
      units usec;
      default 40000;
      description
        "The traffic class should use a path with a
        defined maximum jitter.";
    }
  }
  description
    "Jitter constraint on the traffic class.";
}
description
  "Jitter constraint on the traffic class.";
}
container bandwidth {
  leaf guaranteed-bw-percent {
    type decimal64 {
      fraction-digits 5;
      range "0..100";
    }
    units percent;
    mandatory true;
    description
      "To be used to define the guaranteed bandwidth
      as a percentage of the available service bandwidth.";
  }
  leaf end-to-end {
    type empty;
    description
      "Used if the bandwidth reservation
      must be done on the MPLS network too.";
  }
  description
    "Bandwidth constraint on the traffic class.";
}
description
  "List of classes of services.";
}
description
  "Container for list of classes of services.";
}
}
description
  "QoS profile configuration.";
}
description
```



```
    "QoS configuration.";
  }
  description
    "This grouping defines QoS parameters for a site.";
}
grouping site-security-authentication {
  container authentication {
    description
      "Authentication parameters.";
  }
  description
    "This grouping defines authentication parameters for a site.";
}
grouping site-security-encryption {
  container encryption {
    if-feature encryption;
    leaf enabled {
      type boolean;
      default false;
      description
        "If true, traffic encryption on the connection is required.";
    }
    leaf layer {
      when "../enabled = 'true'" {
        description
          "Require a value for layer when enabled is true.";
      }
      type enumeration {
        enum layer2 {
          description
            "Encryption will occur at Layer 2.";
        }
        enum layer3 {
          description
            "Encryption will occur at Layer 3.
            For example, IPsec may be used when
            a customer requests Layer 3 encryption.";
        }
      }
    }
    description
      "Layer on which encryption is applied.";
  }
  description
    "";
}
container encryption-profile {
  choice profile {
    case provider-profile {
```

```
    leaf profile-name {
      type leafref {
        path "/l3vpn-ntw/vpn-profiles/valid-provider-identifiers"+
          "/encryption-profile-identifier/id";
      }
      description
        "Name of the SP profile to be applied.";
    }
  }
  case customer-profile {
    leaf algorithm {
      type string;
      description
        "Encryption algorithm to be used.";
    }
  }
  description
    "";
}
choice key-type {
  default psk;
  case psk {
    leaf preshared-key {
      type string;
      description
        "Pre-Shared Key (PSK) coming from the customer.";
    }
  }
  description
    "Choice of encryption profile.
    The encryption profile can be the provider profile
    or customer profile.";
}
description
  "This grouping defines encryption parameters for a site.";
}
description
  "";
}
grouping site-attachment-bearer {
  container bearer {
    container requested-type {
      if-feature requested-type;
      leaf requested-type {
        type string;
        description
          "Type of requested bearer: Ethernet, DSL,
          Wireless, etc. Operator specific.";
      }
    }
  }
}
```

```
    }
    leaf strict {
      type boolean;
      default false;
      description
        "Defines whether requested-type is a preference
        or a strict requirement.";
    }
    description
      "Container for requested-type.";
  }
  leaf always-on {
    if-feature always-on;
    type boolean;
    default true;
    description
      "Request for an always-on access type.
      For example, this could mean no dial access type.";
  }

  leaf bearer-reference {
    if-feature bearer-reference;
    type string;
    description
      "This is an internal reference for the SP.";
  }
  description
    "Bearer-specific parameters.
    To be augmented.";

  uses ethernet-params;

  uses pseudowire-params {
    when "/l3vpn-ntw/sites/site/site-network-accesses" +
        "/site-network-access/site-network-access-type ='pseudowire'"
    {
      description "pseudowire specific parameters";
    }
  }
}
description
  "Defines physical properties of a site attachment.";
}
grouping site-routing {
  container routing-protocols {
    list routing-protocol {
      key type;
    }
  }
}
```

```
leaf type {
  type identityref {
    base routing-protocol-type;
  }
  description
  "Type of routing protocol.";
}

list routing-profiles {
  key "id";

  leaf id {
    type leafref {
      path "/l3vpn-ntw/vpn-profiles/valid-provider-identifiers"+
        "/routing-profile-identifier/id";
    }
    description
    "Routing profile to be used.";
  }

  leaf type {
    type ie-type;
    description
    "Import, export or both.";
  }
}

description
"Import or Export profile reference";
}

container ospf {
  when "derived-from-or-self(.. /type, 'l3vpn-ntw:ospf') " {
    description
    "Only applies when protocol is OSPF.";
  }
  if-feature rtg-ospf;
  leaf-list address-family {
    type address-family;
    min-elements "1";
    description
    "If OSPF is used on this site, this node
    contains a configured value. This node
    contains at least one address family
    to be activated.";
  }
  leaf area-address {
    type yang:dotted-quad;
    mandatory true;
  }
}
```

```
        description
        "Area address.";
    }
    leaf metric {
        type uint16;
        default 1;
        description
        "Metric of the PE-CE link. It is used
        in the routing state calculation and
        path selection.";
    }
}

/* Extension */

leaf mtu {
    type uint16;
    description "Maximum transmission unit for a given
    OSPF link.";
}

uses security-params;

/* End of Extension */

container sham-links {
    if-feature rtg-ospf-sham-link;
    list sham-link {
        key target-site;
        leaf target-site {
            type svc-id;
            description
            "Target site for the sham link connection.
            The site is referred to by its ID.";
        }
        leaf metric {
            type uint16;
            default 1;
            description
            "Metric of the sham link. It is used in
            the routing state calculation and path
            selection. The default value is set
            to 1.";
        }
        description
        "Creates a sham link with another site.";
    }
}
description
```

```
    "List of sham links.";
  }
  description
  "OSPF-specific configuration.";
}
container bgp {
  when "derived-from-or-self(..../type, 'l3vpn-ntw:bgp')" {
    description
    "Only applies when protocol is BGP.";
  }
  if-feature rtg-bgp;
  leaf autonomous-system {
    type uint32;
    mandatory true;
    description
    "Customer AS number in case the customer
    requests BGP routing.";
  }
  leaf-list address-family {
    type address-family;
    min-elements "1";
    description
    "If BGP is used on this site, this node
    contains a configured value. This node
    contains at least one address family
    to be activated.";
  }
  /* Extension */
  leaf neighbor {
    type inet:ip-address;
    description
    "IP address of the BGP neighbor.";
  }

  leaf multihop {
    type uint8;
    description
    "Describes the number of hops allowed between the
    given BGP neighbor and the PE router.";
  }

  uses security-params;

  description
  "BGP-specific configuration.";
}
container static {
  when "derived-from-or-self(..../type, 'l3vpn-ntw:static')" {
```

```
    description
    "Only applies when protocol is static.
    BGP activation requires the SP to know
    the address of the customer peer.  When
    BGP is enabled, the 'static-address'
    allocation type for the IP connection
    MUST be used.";
}
container cascaded-lan-prefixes {
  list ipv4-lan-prefixes {
    if-feature ipv4;
    key "lan next-hop";
    leaf lan {
      type inet:ipv4-prefix;
      description
      "LAN prefixes.";
    }
    leaf lan-tag {
      type string;
      description
      "Internal tag to be used in VPN policies.";
    }
    leaf next-hop {
      type inet:ipv4-address;
      description
      "Next-hop address to use on the customer side.";
    }
  }
  description
  "List of LAN prefixes for the site.";
}
list ipv6-lan-prefixes {
  if-feature ipv6;
  key "lan next-hop";
  leaf lan {
    type inet:ipv6-prefix;
    description
    "LAN prefixes.";
  }
  leaf lan-tag {
    type string;
    description
    "Internal tag to be used in VPN policies.";
  }
  leaf next-hop {
    type inet:ipv6-address;
    description
    "Next-hop address to use on the customer side.";
  }
}
```

```
    description
    "List of LAN prefixes for the site.";
  }
  description
  "LAN prefixes from the customer.";
}
description
"Configuration specific to static routing.";
}
container rip {
  when "derived-from-or-self(..../type, 'l3vpn-ntw:rip')" {
    description
    "Only applies when the protocol is RIP. For IPv4,
    the model assumes that RIP version 2 is used.";
  }
  if-feature rtg-rip;
  leaf-list address-family {
    type address-family;
    min-elements "1";
    description
    "If RIP is used on this site, this node
    contains a configured value. This node
    contains at least one address family
    to be activated.";
  }
  description
  "Configuration specific to RIP routing.";
}
container vrrp {
  when "derived-from-or-self(..../type, 'l3vpn-ntw:vrrp')" {
    description
    "Only applies when protocol is VRRP.";
  }
  if-feature rtg-vrrp;
  leaf-list address-family {
    type address-family;
    min-elements "1";
    description
    "If VRRP is used on this site, this node
    contains a configured value. This node contains
    at least one address family to be activated.";
  }
  description
  "Configuration specific to VRRP routing.";
}
description
"List of routing protocols used on
the site. This list can be augmented.";
```



```
    }
    description
    "Defines routing protocols.";
  }
  description
  "Grouping for routing protocols.";
}
grouping site-attachment-ip-connection {

  container ip-connection {
    container ipv4 {
      if-feature ipv4;
      leaf address-allocation-type {
        type identityref {
          base address-allocation-type;
        }
        must "not (derived-from-or-self(current(), 'l3vpn-ntw:slaac') or "+
            "derived-from-or-self(current(), '"+
            "'l3vpn-ntw:provider-dhcp-slaac'))" {
          error-message "SLAAC is only applicable to IPv6";
        }
        description
        "Defines how addresses are allocated.
        If there is no value for the address
        allocation type, then IPv4 is not enabled.";
      }
    }
    container provider-dhcp {
      when "derived-from-or-self(..../address-allocation-type, '"+
          "'l3vpn-ntw:provider-dhcp')" {
        description
        "Only applies when addresses are allocated by DHCP.";
      }
    }
    leaf provider-address {
      type inet:ipv4-address;
      description
      "Address of provider side.  If provider-address is not
      specified, then prefix length should not be specified
      either.  It also implies provider-dhcp allocation is
      not enabled.  If provider-address is specified, then
      the prefix length may or may not be specified.";
    }
    leaf prefix-length {
      type uint8 {
        range "0..32";
      }
      must "(../provider-address)" {
        error-message
        "If the prefix length is specified, provider-address
```

```
        must also be specified.";
        description
            "If the prefix length is specified, provider-address
            must also be specified.";
    }
    description
        "Subnet prefix length expressed in bits.
        If not specified, or specified as zero,
        this means the customer leaves the actual
        prefix length value to the provider.";
    }
    choice address-assign {
        default number;
        case number {
            leaf number-of-dynamic-address {
                type uint16;
                default 1;
                description
                    "Describes the number of IP addresses
                    the customer requires.";
            }
        }
        case explicit {
            container customer-addresses {
                list address-group {
                    key "group-id";
                    leaf group-id {
                        type string;
                        description
                            "Group-id for the address range from
                            start-address to end-address.";
                    }
                }
                leaf start-address {
                    type inet:ipv4-address;
                    description
                        "First address.";
                }
                leaf end-address {
                    type inet:ipv4-address;
                    description
                        "Last address.";
                }
            }
            description
                "Describes IP addresses allocated by DHCP.
                When only start-address or only end-address
                is present, it represents a single address.
                When both start-address and end-address are
                specified, it implies a range inclusive of both
```

```

        addresses.  If no address is specified, it implies
        customer addresses group is not supported.";
    }
    description
    "Container for customer addresses is allocated by DHCP.";
}
}
    description
    "Choice for the way to assign addresses.";
}
    description
    "DHCP allocated addresses related parameters.";
}
container dhcp-relay {
    when "derived-from-or-self(..address-allocation-type, "+
    "'l3vpn-ntw:provider-dhcp-relay')" {
        description
        "Only applies when provider is required to implement
        DHCP relay function.";
    }
}
leaf provider-address {
    type inet:ipv4-address;
    description
    "Address of provider side.  If provider-address is not
    specified, then prefix length should not be specified
    either.  It also implies provider-dhcp allocation is
    not enabled.  If provider-address is specified, then
    prefix length may or may not be specified.";
}
leaf prefix-length {
    type uint8 {
        range "0..32";
    }
}
must "(../provider-address)" {
    error-message
    "If prefix length is specified, provider-address
    must also be specified.";
    description
    "If prefix length is specified, provider-address
    must also be specified.";
}
}
    description
    "Subnet prefix length expressed in bits.  If not
    specified, or specified as zero, this means the
    customer leaves the actual prefix length value
    to the provider.";
}
container customer-dhcp-servers {

```

```
leaf-list server-ip-address {
  type inet:ipv4-address;
  description
    "IP address of customer DHCP server.";
}
description
  "Container for list of customer DHCP servers.";
}
description
  "DHCP relay provided by operator.";
}
container addresses {
  when "derived-from-or-self(..../address-allocation-type, '"+
    "'l3vpn-ntw:static-address')" {
    description
      "Only applies when protocol allocation type is static.";
  }
  leaf provider-address {
    type inet:ipv4-address;
    description
      "IPv4 Address List of the provider side.
      When the protocol allocation type is static,
      the provider address must be configured.";
  }
  leaf customer-address {
    type inet:ipv4-address;
    description
      "IPv4 Address of customer side.";
  }
  leaf prefix-length {
    type uint8 {
      range "0..32";
    }
    description
      "Subnet prefix length expressed in bits.
      It is applied to both provider-address
      and customer-address.";
  }
  description
    "Describes IPv4 addresses used.";
}
description
  "IPv4-specific parameters.";
}
container ipv6 {
  if-feature ipv6;
  leaf address-allocation-type {
    type identityref {
```

```
    base address-allocation-type;
  }
  description
    "Defines how addresses are allocated.
    If there is no value for the address
    allocation type, then IPv6 is
    not enabled.";
}

container provider-dhcp {
  when "derived-from-or-self(..address-allocation-type, "+"
    "'l3vpn-ntw:provider-dhcp'") "+"
    "or derived-from-or-self(..address-allocation-type, "+"
    "'l3vpn-ntw:provider-dhcp-slaac'")" {
    description
      "Only applies when addresses are allocated by DHCP.";
  }
  leaf provider-address {
    type inet:ipv6-address;
    description
      "Address of the provider side. If provider-address
      is not specified, then prefix length should not be
      specified either. It also implies provider-dhcp
      allocation is not enabled. If provider-address is
      specified, then prefix length may or may
      not be specified.";
  }
  leaf prefix-length {
    type uint8 {
      range "0..128";
    }
    must "(../provider-address)" {
      error-message
        "If prefix length is specified, provider-address
        must also be specified.";
      description
        "If prefix length is specified, provider-address
        must also be specified.";
    }
    description
      "Subnet prefix length expressed in bits. If not
      specified, or specified as zero, this means the
      customer leaves the actual prefix length value
      to the provider.";
  }
  choice address-assign {
    default number;
    case number {
```

```
    leaf number-of-dynamic-address {
      type uint16;
      default 1;
      description
        "Describes the number of IP addresses the customer
        requires.";
    }
  }
  case explicit {
    container customer-addresses {
      list address-group {
        key "group-id";
        leaf group-id {
          type string;
          description
            "Group-id for the address range from
            start-address to end-address.";
        }
        leaf start-address {
          type inet:ipv6-address;
          description
            "First address.";
        }
        leaf end-address {
          type inet:ipv6-address;
          description
            "Last address.";
        }
        description
          "Describes IP addresses allocated by DHCP. When only
          start-address or only end-address is present, it
          represents a single address. When both start-address
          and end-address are specified, it implies a range
          inclusive of both addresses. If no address is
          specified, it implies customer addresses group is
          not supported.";
      }
      description
        "Container for customer addresses allocated by DHCP.";
    }
  }
  description
    "Choice for the way to assign addresses.";
}
description
  "DHCP allocated addresses related parameters.";
}
container dhcp-relay {
```

```
when "derived-from-or-self(..address-allocation-type, "+
    "'l3vpn-ntw:provider-dhcp-relay')" {
  description
  "Only applies when the provider is required
  to implement DHCP relay function.";
}

leaf provider-address {
  type inet:ipv6-address;
  description
  "Address of the provider side.  If provider-address is
  not specified, then prefix length should not be
  specified either.  It also implies provider-dhcp
  allocation is not enabled.  If provider address
  is specified, then prefix length may or may
  not be specified.";
}

leaf prefix-length {
  type uint8 {
    range "0..128";
  }
  must "(../provider-address)" {
    error-message
    "If prefix length is specified, provider-address
    must also be specified.";
    description
    "If prefix length is specified, provider-address
    must also be specified.";
  }
  description
  "Subnet prefix length expressed in bits.  If not
  specified, or specified as zero, this means the
  customer leaves the actual prefix length value
  to the provider.";
}

container customer-dhcp-servers {
  leaf-list server-ip-address {
    type inet:ipv6-address;
    description
    "This node contains the IP address of
    the customer DHCP server.  If the DHCP relay
    function is implemented by the
    provider, this node contains the
    configured value.";
  }
  description
  "Container for list of customer DHCP servers.";
}

description
```

```
"DHCP relay provided by operator.";
}
container addresses {
  when "derived-from-or-self(..address-allocation-type, "+
    "'l3vpn-ntw:static-address')" {
    description
      "Only applies when protocol allocation type is static.";
  }
  leaf provider-address {
    type inet:ipv6-address;
    description
      "IPv6 Address of the provider side. When the protocol
      allocation type is static, the provider address
      must be configured.";
  }
  leaf customer-address {
    type inet:ipv6-address;
    description
      "The IPv6 Address of the customer side.";
  }
  leaf prefix-length {
    type uint8 {
      range "0..128";
    }
    description
      "Subnet prefix length expressed in bits.
      It is applied to both provider-address and
      customer-address.";
  }
  description
    "Describes IPv6 addresses used.";
  description
    "IPv6-specific parameters.";
}
container oam {
  container bfd {
    if-feature bfd;
    leaf enabled {
      type boolean;
      default false;
      description
        "If true, BFD activation is required.";
    }
  }
  choice holdtime {
    default fixed;
    case fixed {
      leaf fixed-value {
```



```
    type uint32;
    units msec;
    description
      "Expected BFD holdtime expressed in msec. The customer
      may impose some fixed values for the holdtime period
      if the provider allows the customer use this function.
      If the provider doesn't allow the customer to use this
      function, the fixed-value will not be set.";
  }
}
case profile {
  leaf profile-name {
    type leafref {
      path "/l3vpn-ntw/vpn-profiles/valid-provider-identifiers/"+
        "bfd-profile-identifier/id";
    }
    description
      "Well-known SP profile name. The provider can propose
      some profiles to the customer, depending on the service
      level the customer wants to achieve. Profile names
      must be communicated to the customer.";
  }
  description
    "Well-known SP profile.";
}
description
  "Choice for holdtime flavor.";
}
description
  "Container for BFD.";
}
description
  "Defines the Operations, Administration, and Maintenance (OAM)
  mechanisms used on the connection. BFD is set as a fault
  detection mechanism, but the 'oam' container can easily
  be augmented by other mechanisms";
}
description
  "Defines connection parameters.";
}
description
  "This grouping defines IP connection parameters.";
}
grouping site-service-multicast {
  container multicast {
    if-feature multicast;
    leaf multicast-site-type {
      type enumeration {
```

```
enum receiver-only {
  description
    "The site only has receivers.";
}
enum source-only {
  description
    "The site only has sources.";
}
enum source-receiver {
  description
    "The site has both sources and receivers.";
}
}
default source-receiver;
description
  "Type of multicast site.";
}
container multicast-address-family {
  leaf ipv4 {
    if-feature ipv4;
    type boolean;
    default false;
    description
      "Enables IPv4 multicast.";
  }
  leaf ipv6 {
    if-feature ipv6;
    type boolean;
    default false;
    description
      "Enables IPv6 multicast.";
  }
  description
    "Defines protocol to carry multicast.";
}
leaf protocol-type {
  type enumeration {
    enum host {
      description
        "Hosts are directly connected to the provider network.
        Host protocols such as IGMP or MLD are required.";
    }
    enum router {
      description
        "Hosts are behind a customer router.
        PIM will be implemented.";
    }
    enum both {
```

```
    description
    "Some hosts are behind a customer router, and
    some others are directly connected to the
    provider network. Both host and routing protocols
    must be used. Typically, IGMP and PIM will be
    implemented.";
  }
}
default "both";
description
"Multicast protocol type to be used with the customer site.";
}
description
"Multicast parameters for the site.";
}
description
"Multicast parameters for the site.";
}
grouping site-management {
  container management {
    leaf type {
      type identityref {
        base management;
      }
      mandatory true;
      description
      "Management type of the connection.";
    }
    description
    "Management configuration.";
  }
  description
  "Management parameters for the site.";
}
grouping site-devices {
  container devices {
    when "derived-from-or-self(..management/type, "+
    "'l3vpn-ntw:provider-managed') or "+
    "derived-from-or-self(..management/type, 'l3vpn-ntw:co-managed') " {
      description
      "Applicable only for provider-managed or
      co-managed device.";
    }
  }
  list device {
    key device-id;
    leaf device-id {
      type svc-id;
      description
```

```
    "Identifier for the device.";
  }
  leaf location {
    type leafref {
      path "../../../../../locations/" +
        "location/location-id";
    }
    mandatory true;
    description
      "Location of the device.";
  }
  container management {
    when "derived-from-or-self ../../../../management/type, "+
      "'l3vpn-ntw:co-managed'" {
      description
        "Applicable only for co-managed device.";
    }
    leaf address-family {
      type address-family;
      description
        "Address family used for management.";
    }
    leaf address {
      when "(../../address-family)" {
        description
          "If address-family is specified, then address should
           also be specified. If address-family is not specified,
           then address should also not be specified.";
      }
      type inet:ip-address;
      mandatory true;
      description
        "Management address.";
    }
    description
      "Management configuration. Applicable only for
       co-managed device.";
  }
  description
    "List of devices requested by customer.";
}
description
  "Device configuration.";
}
description
  "Grouping for device allocation.";
}
grouping site-vpn-flavor {
```

```
leaf site-vpn-flavor {
  type identityref {
    base site-vpn-flavor;
  }
  default site-vpn-flavor-single;
  description
    "Defines the way the VPN multiplexing is done, e.g., whether
    the site belongs to a single VPN site or a multiVPN; or, in the case
    of a multiVPN, whether the logical accesses of the sites belong
    to the same set of VPNs or each logical access maps to
    different VPNs.";
}
description
  "Grouping for site VPN flavor.";
}
grouping site-maximum-routes {
  container maximum-routes {
    list address-family {
      key af;
      leaf af {
        type address-family;
        description
          "Address family.";
      }
      leaf maximum-routes {
        type uint32;
        description
          "Maximum prefixes the VRF can accept
          for this address family.";
      }
      description
        "List of address families.";
    }
    description
      "Defines 'maximum-routes' for the VRF.";
  }
  description
    "Defines 'maximum-routes' for the site.";
}
grouping site-security {
  container security {
    uses site-security-authentication;
    uses site-security-encryption;
    description
      "Site-specific security parameters.";
  }
  description
    "Grouping for security parameters.";
```

```
}
grouping site-service {
  container service {
    uses site-service-qos-profile;
    uses site-service-mpls;
    uses site-service-multicast;
    description
      "Service parameters on the attachment.";
  }
  description
    "Grouping for service parameters.";
}
grouping site-network-access-service {
  container service {
    uses site-service-basic;
    /* Extension */
    /* uses svc-bandwidth-params; */
    /* EoExt */
    uses site-service-qos-profile;
    uses site-service-mpls;
    uses site-service-multicast;
    description
      "Service parameters on the attachment.";
  }
  description
    "Grouping for service parameters.";
}
grouping vpn-extranet {
  container extranet-vpns {
    if-feature extranet-vpn;
    list extranet-vpn {
      key vpn-id;
      leaf vpn-id {
        type svc-id;
        description
          "Identifies the target VPN the local VPN want to access.";
      }
      leaf local-sites-role {
        type identityref {
          base site-role;
        }
        default any-to-any-role;
        description
          "This describes the role of the
          local sites in the target VPN topology.  In the any-to-any VPN
          service topology, the local sites must have the same role, which
          will be 'any-to-any-role'.  In the Hub-and-Spoke VPN service
          topology or the Hub-and-Spoke disjoint VPN service topology,
```

```
    the local sites must have a Hub role or a Spoke role.";
  }
  description
    "List of extranet VPNs or target VPNs the local VPN is
    attached to.";
  }
  description
    "Container for extranet VPN configuration.";
  }
  description
    "Grouping for extranet VPN configuration.
    This provides an easy way to interconnect
    all sites from two VPNs.";
  }
  grouping site-attachment-availability {
    container availability {
      leaf access-priority {
        type uint32;
        default 100;
        description
          "Defines the priority for the access.
          The higher the access-priority value,
          the higher the preference of the
          access will be.";
      }
      description
        "Availability parameters (used for multihoming).";
    }
    description
      "Defines availability parameters for a site.";
  }
  grouping vpn-profile-cfg {
    container valid-provider-identifiers {
      list cloud-identifier {
        if-feature cloud-access;
        key id;
        leaf id {
          type string;
          description
            "Identification of cloud service.
            Local administration meaning.";
        }
        description
          "List for Cloud Identifiers.";
      }
      list encryption-profile-identifier {
        key id;
        leaf id {
```

```
    type string;
    description
      "Identification of the SP encryption profile
      to be used.  Local administration meaning.";
  }
  description
    "List for encryption profile identifiers.";
}
list qos-profile-identifier {
  key id;
  leaf id {
    type string;
    description
      "Identification of the QoS Profile to be used.
      Local administration meaning.";
  }
  description
    "List for QoS Profile Identifiers.";
}
list bfd-profile-identifier {
  key id;
  leaf id {
    type string;
    description
      "Identification of the SP BFD Profile to be used.
      Local administration meaning.";
  }
  description
    "List for BFD Profile identifiers.";
}

list routing-profile-identifier {
  key id;
  leaf id {
    type string;
    description
      "Identification of the routing Profile to be used
      by the routing-protocols within sites and site-
      network-accesses. Local administration meaning.";
  }
  description
    "List for Routing Profile Identifiers.";
}

nacm:default-deny-write;
description
  "Container for Valid Provider Identifies.";
}
```



```
    description
      "Grouping for VPN Profile configuration.";
  }
  grouping vpn-svc-cfg {
    leaf vpn-id {
      type svc-id;
      description
        "VPN identifier. Local administration meaning.";
    }
    leaf customer-name {
      type string;
      description
        "Name of the customer that actually uses the VPN service.
        In the case that any intermediary (e.g., Tier-2 provider
        or partner) sells the VPN service to their end user
        on behalf of the original service provider (e.g., Tier-1
        provider), the original service provider may require the
        customer name to provide smooth activation/commissioning
        and operation for the service.";
    }
    leaf vpn-service-topology {
      type identityref {
        base vpn-topology;
      }
      default any-to-any;
      description
        "VPN service topology.";
    }

    leaf description {
      type string;
      description
        "Textual description of a VPN service.";
    }
  }

  uses ie-profiles-params;
  uses vpn-nodes-params;
  uses vpn-service-cloud-access;
  uses vpn-service-multicast;
  uses vpn-service-mpls;
  uses vpn-extranet;
  description
    "Grouping for VPN service configuration.";
}
grouping site-top-level-cfg {
  uses operational-requirements;
  uses customer-location-info;
  uses site-devices;
```

```
uses site-diversity;
uses site-management;
uses site-vpn-flavor;
uses site-maximum-routes;
uses site-security;
uses site-service;
uses site-protection;
uses site-routing;
description
  "Grouping for site top-level configuration.";
}
grouping site-network-access-top-level-cfg {

  /* Extension */

  uses status-params;

  /* End of Extension */

  leaf site-network-access-type {
    type identityref {
      base site-network-access-type;
    }
    default point-to-point;
    description
      "Describes the type of connection, e.g.,
      point-to-point or multipoint.";
  }
  choice location-flavor {
    case location {
      when "derived-from-or-self ../../management/type, "+
        "'l3vpn-ntw:customer-managed'" {
        description
          "Applicable only for customer-managed device.";
      }
      leaf location-reference {
        type leafref {
          path "../../locations/location/location-id";
        }
        description
          "Location of the site-network-access.";
      }
    }
  }
  case device {
    when "derived-from-or-self ../../management/type, "+
      "'l3vpn-ntw:provider-managed' or "+
      "derived-from-or-self ../../management/type, "+
      "'l3vpn-ntw:co-managed'" {
```

```
    description
      "Applicable only for provider-managed or co-managed device.";
  }
  leaf device-reference {
    type leafref {
      path "../..../devices/device/device-id";
    }
    description
      "Identifier of CE to use.";
  }
}
mandatory true;
description
  "Choice of how to describe the site's location.";
}
uses access-diversity;
uses site-attachment-bearer;
uses site-attachment-ip-connection;
uses site-security;
uses site-network-access-service;
uses site-routing;
uses site-attachment-availability;
description
  "Grouping for site network access top-level configuration.";
}

/* Extensions */

/* Bearers in a site */
grouping site-bearer-params {

  container site-bearers {
    list bearer {
      key "bearer-id";

      leaf bearer-id {
        type string;
        description "";
      }

      leaf BearerType {
        type identityref {
          base bearer-inf-type;
        }
        description
          "Request for an Bearer access type."
      }
    }
  }
}
```

```
        Choose between port or lag connection type.";
    }

    leaf ne-id {
        type string;
        description
            "NE-id reference.";
    }

    leaf port-id {
        type string;
        description
            "Port-id in format slot/ card /port.";
    }

    leaf lag-id {
        type string;
        description
            "lag-id in format id.";
    }
    description
        "Parameters used to identify each bearer";
    }
    description
        "Grouping to reuse the site bearer assignment";
    }
    description
        "Grouping to reuse the site bearer assignment";
    }

/* UNUSED */
grouping svc-bandwidth-params {
    container svc-bandwidth {
        if-feature "input-bw";
        list bandwidth {
            key "direction type";
            leaf direction {
                type identityref {
                    base bw-direction;
                }
                description
                    "Indicates the bandwidth direction. It can be
                     the bandwidth download direction from the SP to
                     the site or the bandwidth upload direction from
                     the site to the SP.";
            }
            leaf type {
                type identityref {
```

```
        base bw-type;
    }
    description
        "Bandwidth type. By default, the bandwidth type
        is set to 'bw-per-cos'.";
}
leaf cos-id {
    when "derived-from-or-self(..type, "
        + "'l3vpn-ntw:bw-per-cos')" {
        description
            "Relevant when the bandwidth type is set to
            'bw-per-cos'.";
    }
    type uint8;
    description
        "Identifier of the CoS, indicated by DSCP or a
        CE-VLAN CoS (802.1p) value in the service frame.
        If the bandwidth type is set to 'bw-per-cos',
        the CoS ID MUST also be specified.";
}
leaf vpn-id {
    when "derived-from-or-self(..type, "
        + "'l3vpn-ntw:bw-per-svc')" {
        description
            "Relevant when the bandwidth type is
            set as bandwidth per VPN service.";
    }
    type svc-id;
    description
        "Identifies the target VPN. If the bandwidth
        type is set as bandwidth per VPN service, the
        vpn-id MUST be specified.";
}
leaf cir {
    type uint64;
    units "bps";
    mandatory true;
    description
        "Committed Information Rate. The maximum number
        of bits that a port can receive or send over
        an interface in one second.";
}
leaf cbs {
    type uint64;
    units "bps";
    mandatory true;
    description
        "Committed Burst Size (CBS). Controls the bursty
```

```
        nature of the traffic. Traffic that does not
        use the configured Committed Information Rate
        (CIR) accumulates credits until the credits
        reach the configured CBS.";
    }
    leaf eir {
        type uint64;
        units "bps";
        description
            "Excess Information Rate (EIR), i.e., excess frame
            delivery allowed that is not subject to an SLA.
            The traffic rate can be limited by the EIR.";
    }
    leaf ebs {
        type uint64;
        units "bps";
        description
            "Excess Burst Size (EBS). The bandwidth available
            for burst traffic from the EBS is subject to the
            amount of bandwidth that is accumulated during
            periods when traffic allocated by the EIR
            policy is not used.";
    }
    leaf pir {
        type uint64;
        units "bps";
        description
            "Peak Information Rate, i.e., maximum frame
            delivery allowed. It is equal to or less
            than the sum of the CIR and the EIR.";
    }
    leaf pbs {
        type uint64;
        units "bps";
        description
            "Peak Burst Size. It is measured in bytes per
            second.";
    }
    description
        "List of bandwidth values (e.g., per CoS,
        per vpn-id).";
}
description
    "From the customer site's perspective, the service
    input/output bandwidth of the connection or
    download/upload bandwidth from the SP/site
    to the site/SP.";
```

```
        description
            " ";
    }

    grouping status-params {
        container status {
            leaf admin-enabled {
                type boolean;
                description
                    "Administrative Status UP/DOWN";
            }
            leaf oper-status {
                type operational-type;
                config false;
                description
                    "Operations status";
            }
            description "";
        }
        description
            "Grouping used to join operational and administrative status
            is re used in the Site Network Access and in the VPN-Node";
    }

    /* Parameters related to vpn-nodes (VRF config.) */
    grouping vpn-nodes-params {
        container vpn-nodes {
            description "";

            list vpn-node {
                key "vpn-node-id ne-id";

                leaf vpn-node-id {
                    type string;
                    description "";
                }

                leaf description {
                    type string;
                    description
                        "Textual description of a VPN node.";
                }

                leaf ne-id {
                    type string;
                    description "";
                }
            }
        }
    }
}
```

```
    }

    leaf router-id {
      type inet:ip-address;
      description
        "router-id information can be ipv4/6 addresses";
    }

    leaf address-family {
      type address-family;
      description
        "Address family used for router-id information.";
    }

    leaf node-role {
      type identityref {
        base site-role;
      }
      default any-to-any-role;
      description
        "Role of the vpn-node in the IP VPN.";
    }
    uses rt-rd;
    uses status-params;

    /* Here we use the name given to the existing structure in sites */
    uses site-maximum-routes;

    leaf node-ie-profile {
      type leafref {
        path "/l3vpn-ntw/vpn-services/"+
          "vpn-service/ie-profiles/ie-profile/ie-profile-id";
      }
      description "";
    }
    description "";
  }
  description "Grouping to define VRF-specific configuration.";
}

/* Parameters related to import and export profiles (RTs RDs.) */
grouping ie-profiles-params {
  container ie-profiles {
    list ie-profile {
      key "ie-profile-id";
      leaf ie-profile-id {
        type string;
      }
    }
  }
}
```



```
        description
            "";
    }
    uses rt-rd;
    description
    "";
    }
    description
    "";
    }
    description
    "Grouping to specify rules for route import and export";
}

grouping pseudowire-params {
    container pseudowire {
        /*leaf far-end {*/
        /*  description "IP of the remote peer of the pseudowire.";*/
        /*  type inet:ip-address;*/
        /*}*/
        leaf vcid {
            type uint32;
            description
            "PW or VC identifier.";
        }
        description
        "Pseudowire termination parameters";
    }
    description
    "Grouping pseudowire termination parameters";
}

grouping security-params {
    container security {
        leaf auth-key {
            type string;
            description
            "MD5 authentication password for the connection towards the
            customer edge.";
        }
        description
        "Container for aggregating any security parameter for routing
        sessions between a PE and a CE.";
    }
    description
    "Grouping to define security parameters";
}
```

```
grouping ethernet-params {
  container connection {
    leaf encapsulation-type {
      type identityref {
        base encapsulation-type;
      }
      default "untagged-int";
      description
        "Encapsulation type. By default, the
        encapsulation type is set to 'untagged'.";
    }
    container tagged-interface {
      leaf type {
        type identityref {
          base tagged-inf-type;
        }
        default "priority-tagged";
        description
          "Tagged interface type. By default,
          the type of the tagged interface is
          'priority-tagged'.";
      }
      container dot1q-vlan-tagged {
        when "derived-from-or-self(..../type, "
          + "'l3vpn-ntw:dot1q') " {
          description
            "Only applies when the type of the tagged
            interface is 'dot1q'.";
        }
        if-feature "dot1q";
        leaf tag-type {
          type identityref {
            base tag-type;
          }
          default "c-vlan";
          description
            "Tag type. By default, the tag type is
            'c-vlan'.";
        }
        leaf cvlan-id {
          type uint16;
          description
            "VLAN identifier.";
        }
        description
          "Tagged interface.";
      }
    }
    container priority-tagged {
```

```
when "derived-from-or-self(..type, "
  + "'l3vpn-ntw:priority-tagged')" {
  description
    "Only applies when the type of the tagged
    interface is 'priority-tagged'.";
}
leaf tag-type {
  type identityref {
    base tag-type;
  }
  default "c-vlan";
  description
    "Tag type. By default, the tag type is
    'c-vlan'.";
}
description
  "Priority tagged.";
}
container qinq {
  when "derived-from-or-self(..type, "
    + "'l3vpn-ntw:qinq')" {
    description
      "Only applies when the type of the tagged
      interface is 'qinq'.";
  }
  if-feature "qinq";
  leaf tag-type {
    type identityref {
      base tag-type;
    }
    default "c-s-vlan";
    description
      "Tag type. By default, the tag type is
      'c-s-vlan'.";
  }
  leaf svlan-id {
    type uint16;
    mandatory true;
    description
      "SVLAN identifier.";
  }
  leaf cvlan-id {
    type uint16;
    mandatory true;
    description
      "CVLAN identifier.";
  }
  description
```

```
    "QinQ.";
}
container qinany {
    when "derived-from-or-self(..type, "
        + "'l3vpn-ntw:qinany')" {
        description
            "Only applies when the type of the tagged
            interface is 'qinany'.";
    }
    if-feature "qinany";
    leaf tag-type {
        type identityref {
            base tag-type;
        }
        default "s-vlan";
        description
            "Tag type. By default, the tag type is
            's-vlan'.";
    }
    leaf svlan-id {
        type uint16;
        mandatory true;
        description
            "Service VLAN ID.";
    }
    description
        "Container for QinAny.";
}
container vxlan {
    when "derived-from-or-self(..type, "
        + "'l3vpn-ntw:vxlan')" {
        description
            "Only applies when the type of the tagged
            interface is 'vxlan'.";
    }
    if-feature "vxlan";
    leaf vni-id {
        type uint32;
        mandatory true;
        description
            "VXLAN Network Identifier (VNI).";
    }
    leaf peer-mode {
        type identityref {
            base vxlan-peer-mode;
        }
        default "static-mode";
        description
```

```
        "Specifies the VXLAN access mode. By default,
        the peer mode is set to 'static-mode'.";
    }
    list peer-list {
        key "peer-ip";
        leaf peer-ip {
            type inet:ip-address;
            description
                "Peer IP.";
        }
        description
            "List of peer IP addresses.";
    }
    description
        "QinQ.";
}
description
    "Container for tagged interfaces.";
}
description
    "Encapsulation types";
}
description
    "Grouping to define encapsulation types";
}

grouping rt-rd {
    leaf rd {
        type rt-types:route-distinguisher;
        description
            "";
    }
    container vpn-targets {
        description
            "Set of route-targets to match for import and export routes
            to/from VRF";
        uses rt-types:vpn-route-targets;
    }
    description
        "";
}

/* Main blocks */
container l3vpn-ntw {
    container vpn-profiles {
        uses vpn-profile-cfg;
        description
            "Container for VPN Profiles.";
    }
}
```

```
}
container vpn-services {
  list vpn-service {
    key vpn-id;
    uses vpn-svc-cfg;
    description
      "List of VPN services.";
  }
  description
    "Top-level container for the VPN services.";
}
container sites {
  list site {
    key site-id;
    leaf site-id {
      type svc-id;
      description
        "Identifier of the site.";
    }
    leaf description {
      type string;
      description
        "Textual description of a site.";
    }
  }
  uses site-top-level-cfg;
  uses operational-requirements-ops;
  uses site-bearer-params;
  container site-network-accesses {
    list site-network-access {
      key site-network-access-id;
      leaf site-network-access-id {
        type svc-id;
        description
          "Identifier for the access.";
      }
      leaf description {
        type string;
        description
          "Textual description of a VPN service.";
      }
    }
    uses site-network-access-top-level-cfg;
    leaf node-id {
      type leafref{
        path "/l3vpn-ntw/vpn-services/vpn-service/vpn-nodes/vpn-node/vpn-node-
id";
      }
      description
        "Reference the VPN node id";
    }
  }
}
```

```
leaf service-id {
  type leafref{
    path "/l3vpn-ntw/vpn-services/vpn-service/vpn-id";
  }
  description
    "Reference the VPN node id";
}
leaf access-group-id {
  type yang:uuid;
  description
    "Reference the Access Goup ID.
    It is used to group and identify SNA with common behavior
    such as dual-homming";
}
description
  "List of accesses for a site.";
}
description
  "List of accesses for a site.";
}
description
  "List of sites.";
}
description
  "Container for sites.";
}
description
  "Main container for L3VPN service configuration.";
}
}
```

Figure 4

## 6. IANA CONSIDERATIONS

This memo includes no request to IANA.

## 7. SECURITY CONSIDERATIONS

All the security considerations of [RFC8299] apply to this document. Subsequent versions will provide additional security considerations.

## 8. IMPLEMENTATION STATUS

This section will be used to track the status of the implementations of the model. It is aimed at being removed if the document becomes RFC.

## 9. ACKNOWLEDGEMENTS

Thanks to Adrian Farrel and Miguel Cros for the suggestions on the document. Thanks to Stephane Litowski and Philip Eardlay for the review. Lots of thanks for the discussions on opsawg mailing list and at IETF meeting. Some of the comments have already been incorporated and the other part of the comments will be addressed in the next versions.

This work was supported in part by the European Commission funded H2020-ICT-2016-2 METRO-HAUL project (G.A. 761727).

## 10. CONTRIBUTORS

Daniel King  
Old Dog Consulting  
Email: daniel@olddog.co.uk

Samier Barguil  
Telefonica  
Email: samier.barguilgiraldo.ext@telefonica.com

Luay Jalil  
Verizon  
Email: luay.jalil@verizon.com

Qin Wu  
Huawei  
Email: bill.wu@huawei.com>

## 11. References

### 11.1. 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>>.

### 11.2. INFORMATIVE REFERENCES

[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>>.



- [RFC7950] Bjorklund, M., Ed., "The YANG 1.1 Data Modeling Language", RFC 7950, DOI 10.17487/RFC7950, August 2016, <<https://www.rfc-editor.org/info/rfc7950>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.
- [RFC8299] Wu, Q., Ed., Litkowski, S., Tomotaki, L., and K. Ogaki, "YANG Data Model for L3VPN Service Delivery", RFC 8299, DOI 10.17487/RFC8299, January 2018, <<https://www.rfc-editor.org/info/rfc8299>>.
- [RFC8309] Wu, Q., Liu, W., and A. Farrel, "Service Models Explained", RFC 8309, DOI 10.17487/RFC8309, January 2018, <<https://www.rfc-editor.org/info/rfc8309>>.
- [RFC8340] Bjorklund, M. and L. Berger, Ed., "YANG Tree Diagrams", BCP 215, RFC 8340, DOI 10.17487/RFC8340, March 2018, <<https://www.rfc-editor.org/info/rfc8340>>.
- [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>>.
- [RFC8466] Wen, B., Fioccola, G., Ed., Xie, C., and L. Jalil, "A YANG Data Model for Layer 2 Virtual Private Network (L2VPN) Service Delivery", RFC 8466, DOI 10.17487/RFC8466, October 2018, <<https://www.rfc-editor.org/info/rfc8466>>.

## Authors' Addresses

Alejandro Aguado  
Nokia  
Madrid  
ES

Email: [alejandro.aguado\\_martin@nokia.com](mailto:alejandro.aguado_martin@nokia.com)

Oscar Gonzalez de Dios (editor)  
Telefonica  
Madrid  
ES

Email: [oscar.gonzalezdedios@telefonica.com](mailto:oscar.gonzalezdedios@telefonica.com)

Victor Lopez  
Telefonica  
Madrid  
ES

Email: victor.lopezalvarez@telefonica.com

Daniel Voyer  
Bell Canada  
CA

Email: daniel.voyer@bell.ca

Luis Angel Munoz  
Vodafone  
ES

Email: luis-angel.munoz@vodafone.com

Network Working Group  
Internet-Draft  
Intended status: Informational  
Expires: January 10, 2020

J. Arkko  
Ericsson  
July 09, 2019

Changes in the Internet Threat Model  
draft-arkko-arch-internet-threat-model-01

Abstract

Communications security has been at the center of many security improvements in the Internet. The goal has been to ensure that communications are protected against outside observers and attackers.

This memo suggests that the existing threat model, while important and still valid, is no longer alone sufficient to cater for the pressing security issues in the Internet. For instance, it is also necessary to protect systems against endpoints that are compromised, malicious, or whose interests simply do not align with the interests of the users. While such protection is difficult, there are some measures that can be taken.

It is particularly important to ensure that as we continue to develop Internet technology, non-communications security related threats are properly understood. While the consideration of these issues is relatively new in the IETF, this memo provides some initial ideas about potential broader threat models to consider when designing protocols for the Internet or when trying to defend against pervasive monitoring. Further down the road, updated threat models could result in changes in RFC 3552 (guidelines for writing security considerations) and RFC 7258 (pervasive monitoring), to include proper consideration of non-communications security threats. It may also be necessary to have dedicated guidance on how systems design and architecture affects security.

Status of This Memo

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

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

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

time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on January 10, 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 (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

#### Table of Contents

1. Introduction . . . . .	2
2. Improvements in Communications Security . . . . .	5
3. Issues in Security Beyond Communications Security . . . . .	5
4. Impacts . . . . .	8
4.1. The Role of End-to-end . . . . .	8
4.2. Trusted networks . . . . .	10
4.2.1. Even closed networks can have compromised nodes . . . . .	11
4.3. Balancing Threats . . . . .	12
5. Guidelines . . . . .	12
6. Potential Changes in IETF Analysis of Protocols . . . . .	14
6.1. Changes in RFC 3552 . . . . .	14
6.2. Changes in RFC 7258 . . . . .	15
6.3. System and Architecture Aspects . . . . .	15
7. Other Work . . . . .	15
8. Conclusions . . . . .	15
9. Acknowledgements . . . . .	16
10. Informative References . . . . .	16
Author's Address . . . . .	18

#### 1. Introduction

Communications security has been at the center of many security improvements in the Internet. The goal has been to ensure that communications are protected against outside observers and attackers. At the IETF, this approach has been formalized in BCP 72 [RFC3552], which defined the Internet threat model in 2003.

The purpose of a threat model is to outline what threats exist in order to assist the protocol designer. But RFC 3552 also ruled some threats to be in scope and of primary interest, and some threats out of scope [RFC3552]:

The Internet environment has a fairly well understood threat model. In general, we assume that the end-systems engaging in a protocol exchange have not themselves been compromised. Protecting against an attack when one of the end-systems has been compromised is extraordinarily difficult. It is, however, possible to design protocols which minimize the extent of the damage done under these circumstances.

By contrast, we assume that the attacker has nearly complete control of the communications channel over which the end-systems communicate. This means that the attacker can read any PDU (Protocol Data Unit) on the network and undetectably remove, change, or inject forged packets onto the wire.

However, the communications-security -only threat model is becoming outdated. This is due to three factors:

- o Advances in protecting most of our communications with strong cryptographic means. This has resulted in much improved communications security, but also highlights the need for addressing other, remaining issues. This is not to say that communications security is not important, it still is: improvements are still needed. Not all communications have been protected, and even out of the already protected communications, not all of their aspects have been fully protected. Fortunately, there are ongoing projects working on improvements.
- o Adversaries have increased their pressure against other avenues of attack, from compromising devices to legal coercion of centralized endpoints in conversations.
- o New adversaries and risks have arisen, e.g., due to creation of large centralized information sources.

In short, attacks are migrating towards the currently easier targets, which no longer necessarily include direct attacks on traffic flows. In addition, trading information about users and ability to influence them has become a common practice for many Internet services, often without consent of the users.

This memo suggests that the existing threat model, while important and still valid, is no longer alone sufficient to cater for the pressing security issues in the Internet. For instance, while it

continues to be very important to protect Internet communications against outsiders, it is also necessary to protect systems against endpoints that are compromised, malicious, or whose interests simply do not align with the interests of the users.

Of course, there are many trade-offs in the Internet on who one chooses to interact with and why or how. It is not the role of this memo to dictate those choices. But it is important that we understand the implications of different practices. It is also important that when it comes to basic Internet infrastructure, our chosen technologies lead to minimal exposure with respect to the non-communications threats.

It is particularly important to ensure that non-communications security related threats are properly understood for any new Internet technology. While the consideration of these issues is relatively new in the IETF, this memo provides some initial ideas about potential broader threat models to consider when designing protocols for the Internet or when trying to defend against pervasive monitoring. Further down the road, updated threat models could result in changes in BCP 72 [RFC3552] (guidelines for writing security considerations) and BCP 188 [RFC7258] (pervasive monitoring), to include proper consideration of non-communications security threats.

It may also be necessary to have dedicated guidance on how systems design and architecture affects security. The sole consideration of communications security aspects in designing Internet protocols may lead to accidental or increased impact of security issues elsewhere. For instance, allowing a participant to unnecessarily collect or receive information may lead to a similar effect as described in [RFC8546] for protocols: over time, unnecessary information will get used with all the associated downsides, regardless of what deployment expectations there were during protocol design.

The rest of this memo is organized as follows. Section 2 and Section 3 outline the situation with respect to communications security and beyond it. Section 4.1 discusses how the author believes the Internet threat model should evolve, and what types of threats should be seen as critical ones and in-scope. Section 5 will also discuss high-level guidance to addressing these threats.

Section 6 outlines the author's suggested future changes to RFC 3552 and RFC 7258 and the need for guidance on the impacts of system design and architecture on security. Comments are solicited on these and other aspects of this document. The best place for discussion is on the arch-discuss list (<https://www.ietf.org/mailman/listinfo/Architecture-discuss>). This memo acts also as an input for the IAB

retreat discussion on threat models, and it is a submission for the IAB DEDR workshop (<https://www.iab.org/activities/workshops/dedr-workshop/>).

Finally, Section 7 highlights other discussions in this problem space and Section 8 draws some conclusions for next steps.

## 2. Improvements in Communications Security

The fraction of Internet traffic that is cryptographically protected has grown tremendously in the last few years. Several factors have contributed to this change, from Snowden revelations to business reasons and to better available technology such as HTTP/2 [RFC7540], TLS 1.3 [RFC8446], QUIC [I-D.ietf-quic-transport].

In many networks, the majority of traffic has flipped from being cleartext to being encrypted. Reaching the level of (almost) all traffic being encrypted is no longer something unthinkable but rather a likely outcome in a few years.

At the same time, technology developments and policy choices have driven the scope of cryptographic protection from protecting only the pure payload to protecting much of the rest as well, including far more header and meta-data information than was protected before. For instance, efforts are ongoing in the IETF to assist encrypting transport headers [I-D.ietf-quic-transport], server domain name information in TLS [I-D.ietf-tls-esni], and domain name queries [RFC8484].

There has also been improvements to ensure that the security protocols that are in use actually have suitable credentials and that those credentials have not been compromised, see, for instance, Let's Encrypt [RFC8555], HSTS [RFC6797], HPKP [RFC7469], and Expect-CT [I-D.ietf-httpbis-expect-ct].

This is not to say that all problems in communications security have been resolved – far from it. But the situation is definitely different from what it was a few years ago. Remaining issues will be and are worked on; the fight between defense and attack will also continue. Communications security will stay at the top of the agenda in any Internet technology development.

## 3. Issues in Security Beyond Communications Security

There are, however, significant issues beyond communications security in the Internet. To begin with, it is not necessarily clear that one can trust all the endpoints.

Of course, the endpoints were never trusted, but the pressures against endpoints issues seem to be mounting. For instance, the users may not be in as much control over their own devices as they used to be due to manufacturer-controlled operating system installations and locked device ecosystems. And within those ecosystems, even the applications that are available tend to have features that users by themselves would most likely not desire to have, such as excessive rights to media, location, and peripherals. There are also designated efforts by various authorities to hack end-user devices as a means of intercepting data about the user.

The situation is different, but not necessarily better on the side of servers. The pattern of communications in today's Internet is almost always via a third party that has at least as much information than the other parties have. For instance, these third parties are typically endpoints for any transport layer security connections, and able to see any communications or other messaging in cleartext. There are some exceptions, of course, e.g., messaging applications with end-to-end protection.

With the growth of trading users' information by many of these third parties, it becomes necessary to take precautions against endpoints that are compromised, malicious, or whose interests simply do not align with the interests of the users.

Specifically, the following issues need attention:

- o Security of users' devices and the ability of the user to control their own equipment.
- o Leaks and attacks related to data at rest.
- o Coercion of some endpoints to reveal information to authorities or surveillance organizations, sometimes even in an extra-territorial fashion.
- o Application design patterns that result in cleartext information passing through a third party or the application owner.
- o Involvement of entities that have no direct need for involvement for the sake of providing the service that the user is after.
- o Network and application architectures that result in a lot of information collected in a (logically) central location.
- o Leverage and control points outside the hands of the users or end-user device owners.



For instance, while e-mail transport security [RFC7817] has become much more widely distributed in recent years, progress in securing e-mail messages between users has been much slower. This has led to a situation where e-mail content is considered a critical resource by mail providers who use it for machine learning, advertisement targeting, and other purposes.

The Domain Name System (DNS) shows signs of ageing but due to the legacy of deployed systems, has changed very slowly. Newer technology [RFC8484] developed at the IETF enables DNS queries to be performed confidentially, but its deployment is happening mostly in browsers that use global DNS resolver services, such as Cloudflare's 1.1.1.1 or Google's 8.8.8.8. This results in faster evolution and better security for end users.

However, if one steps back and considers the overall security effects of these developments, the resulting effects can be different. While the security of the actual protocol exchanges improves with the introduction of this new technology, at the same time this implies a move from using a worldwide distributed set of DNS resolvers into more centralised global resolvers. While these resolvers are very well maintained (and a great service), they are potential high-value targets for pervasive monitoring and Denial-of-Service (DoS) attacks. In 2016, for example, DoS attacks were launched against Dyn, one of the largest DNS providers, leading to some outages. It is difficult to imagine that DNS resolvers wouldn't be a target in many future attacks or pervasive monitoring projects.

Unfortunately, there is little that even large service providers can do to refuse authority-sanctioned pervasive monitoring. As a result it seems that the only reasonable course of defense is to ensure that no such information or control point exists.

There are other examples about the perils of centralised solutions in Internet infrastructure. The DNS example involved an interesting combination of information flows (who is asking for what domain names) as well as a potential ability to exert control (what domains will actually resolve to an address). Routing systems are primarily about control. While there are intra-domain centralized routing solutions (such as PCE [RFC4655]), a control within a single administrative domain is usually not the kind of centralization that we would be worried about. Global centralization would be much more concerning. Fortunately, global Internet routing is performed among peers. However, controls could be introduced even in this global, distributed system. To secure some of the control exchanges, the Resource Public Key Infrastructure (RPKI) system ([RFC6480]) allows selected Certification Authorities (CAs) to help drive decisions about which participants in the routing infrastructure can

make what claims. If this system were globally centralized, it would be a concern, but again, fortunately, current designs involve at least regional distribution.

In general, many recent attacks relate more to information than communications. For instance, personal information leaks typically happen via information stored on a compromised server rather than capturing communications. There is little hope that such attacks can be prevented entirely. Again, the best course of action seems to be avoid the disclosure of information in the first place, or at least to not perform that in a manner that makes it possible that others can readily use the information.

#### 4. Impacts

##### 4.1. The Role of End-to-end

[RFC1958] notes that "end-to-end functions can best be realised by end-to-end protocols":

The basic argument is that, as a first principle, certain required end-to-end functions can only be performed correctly by the end-systems themselves. A specific case is that any network, however carefully designed, will be subject to failures of transmission at some statistically determined rate. The best way to cope with this is to accept it, and give responsibility for the integrity of communication to the end systems. Another specific case is end-to-end security.

The "end-to-end argument" was originally described by Saltzer et al [Saltzer]. They said:

The function in question can completely and correctly be implemented only with the knowledge and help of the application standing at the endpoints of the communication system. Therefore, providing that questioned function as a feature of the communication system itself is not possible.

These functional arguments align with other, practical arguments about the evolution of the Internet under the end-to-end model. The endpoints evolve quickly, often with simply having one party change the necessary software on both ends. Whereas waiting for network upgrades would involve potentially a large number of parties from application owners to multiple network operators.

The end-to-end model supports permissionless innovation where new innovation can flourish in the Internet without excessive wait for other parties to act.

But the details matter. What is considered an endpoint? What characteristics of Internet are we trying to optimize? This memo makes the argument that, for security purposes, there is a significant distinction between actual endpoints from a user's interaction perspective (e.g., another user) and from a system perspective (e.g., a third party relaying a message).

This memo proposes to focus on the distinction between "real ends" and other endpoints to guide the development of protocols. A conversation between one "real end" to another "real end" has necessarily different security needs than a conversation between, say, one of the "real ends" and a component in a larger system. The end-to-end argument is used primarily for the design of one protocol. The security of the system, however, depends on the entire system and potentially multiple storage, compute, and communication protocol aspects. All have to work properly together to obtain security.

For instance, a transport connection between two components of a system is not an end-to-end connection even if it encompasses all the protocol layers up to the application layer. It is not end-to-end, if the information or control function it carries actually extends beyond those components. For instance, just because an e-mail server can read the contents of an e-mail message does not make it a legitimate recipient of the e-mail.

This memo also proposes to focus on the "need to know" aspect in systems. Information should not be disclosed, stored, or routed in cleartext through parties that do not absolutely need to have that information.

The proposed argument about real ends is as follows:

Application functions are best realised by the entities directly serving the users, and when more than one entity is involved, by end-to-end protocols. The role and authority of any additional entities necessary to carry out a function should match their part of the function. No information or control roles should be provided to these additional entities unless it is required by the function they provide.

For instance, a particular piece of information may be necessary for the other real endpoint, such as message contents for another user. The same piece of information may not be necessary for any additional parties, unless the information had to do with, say, routing information for the message to reach the other user. When information is only needed by the actual other endpoint, it should be protected and be only relayed to the actual other endpoint. Protocol

design should ensure that the additional parties do not have access to the information.

Note that it may well be that the easiest design approach is to send all information to a third party and have majority of actual functionality reside in that third party. But this is a case of a clear tradeoff between ease of change by evolving that third party vs. providing reasonable security against misuse of information.

Note that the above "real ends" argument is not limited to communication systems. Even an application that does not communicate with anyone else than its user may be implemented on top of a distributed system where some information about the user is exposed to untrusted parties.

The implications of the system security also extend beyond information and control aspects. For instance, poorly design component protocols can become DoS vectors which are then used to attack other parts of the system. Availability is an important aspect to consider in the analysis along other aspects.

#### 4.2. Trusted networks

Some systems are thought of as being deployed only in a closed setting, where all the relevant nodes are under direct control of the network administrators. Technologies developed for such networks tend to be optimized, at least initially, for these environments, and may lack security features necessary for different types of deployments.

It is well known that many such systems evolve over time, grow, and get used and connected in new ways. For instance, the collaboration and mergers between organizations, and new services for customers may change the system or its environment. A system that used to be truly within an administrative domain may suddenly need to cross network boundaries or even run over the Internet. As a result, it is also well known that it is good to ensure that underlying technologies used in such systems can cope with that evolution, for instance, by having the necessary security capabilities to operate in different environments.

In general, the outside vs. inside security model is outdated for most situations, due to the complex and evolving networks and the need to support mixture of devices from different sources (e.g., BYOD networks). Network virtualization also implies that previously clear notions of local area networks and physical proximity may create an entirely different reality from what appears from a simple notion of a local network.

#### 4.2.1. Even closed networks can have compromised nodes

This memo argues that the situation is even more dire than what was explained above. It is impossible to ensure that all components in a network are actually trusted. Even in a closed network with carefully managed components there may be compromised components, and this should be factored into the design of the system and protocols used in the system.

For instance, during the Snowden revelations it was reported that internal communication flows of large content providers were compromised in an effort to acquire information from large number of end users. This shows the need to protect not just communications targeted to go over the Internet, but in many cases also internal and control communications.

Furthermore, there is a danger of compromised nodes, so communications security alone will be insufficient to protect against this. The defences against this include limiting information within networks to the parties that have a need to know, as well as limiting control capabilities. This is necessary even when all the nodes are under the control of the same network manager; the network manager needs to assume that some nodes and communications will be compromised, and build a system to mitigate or minimise attacks even under that assumption.

Even airgapped networks can have these issues, as evidenced, for instance, by the Stuxnet worm. The Internet is not the only form of connectivity, as most systems include, for instance, USB ports that proved to be the achilles heel of the targets in the Stuxnet case. More commonly, every system runs large amount of software, and it is often not practical or even possible to black the software to prevent compromised code even in a high-security setting, let alone in commercial or private networks. Installation media, physical ports, both open source and proprietary programs, firmware, or even innocent-looking components on a circuit board can be suspect. In addition, complex underlying computing platforms, such as modern CPUs with underlying security and management tools are prone for problems.

In general, this means that one cannot entirely trust even a closed system where you picked all the components yourself. Analysis for the security of many interesting real-world systems now commonly needs to include cross-component attacks, e.g., the use of car radios and other externally communicating devices as part of attacks launched against the control components such as breaks in a car [Savage].

#### 4.3. Balancing Threats

Note that not all information needs to be protected, and not all threats can be protected against. But it is important that the main threats are understood and protected against.

Sometimes there are higher-level mechanisms that provide safeguards for failures. For instance, it is very difficult in general to protect against denial-of-service against compromised nodes on a communications path. However, it may be possible to detect that a service has failed.

Another example is from packet-carrying networks. Payload traffic that has been properly protected with encryption does not provide much value to an attacker. As a result, it does not always make sense, for instance, to encrypt every packet transmission in a packet-carrying system where the traffic is already encrypted at other layers. But it almost always makes sense to protect control communications and to understand the impacts of compromised nodes, particularly control nodes.

#### 5. Guidelines

As [RFC3935] says:

We embrace technical concepts such as decentralized control, edge-user empowerment and sharing of resources, because those concepts resonate with the core values of the IETF community.

To be more specific, this memo suggests the following guidelines for protocol designers:

1. Consider first principles in protecting information and systems, rather than following a specific pattern such as protecting information in a particular way or at a particular protocol layer. It is necessary to understand what components can be compromised, where interests may or may not be aligned, and what parties have a legitimate role in being a party to a specific information or a control task.
2. Minimize information passed to others: Information passed to another party in a protocol exchange should be minimized to guard against the potential compromise of that party.
3. Perform end-to-end protection via other parties: Information passed via another party who does not intrinsically need the information to perform its function should be protected end-to-end to its intended recipient. This guideline is general, and

holds equally for sending TCP/IP packets, TLS connections, or application-layer interactions. As [I-D.iab-wire-image] notes, it is a useful design rule to avoid "accidental invariance" (the deployment of on-path devices that over-time start to make assumptions about protocols). However, it is also a necessary security design rule to avoid "accidental disclosure" where information originally thought to be benign and untapped over-time becomes a significant information leak. This guideline can also be applied for different aspects of security, e.g., confidentiality and integrity protection, depending on what the specific need for information is in the other parties.

4. Minimize passing of control functions to others: Any passing of control functions to other parties should be minimized to guard against the potential misuse of those control functions. This applies to both technical (e.g., nodes that assign resources) and process control functions (e.g., the ability to allocate number or develop extensions). Control functions can also become a matter of contest and power struggle, even in cases where their function as such is minimal, as we saw with the IANA transition debates.
5. Avoid centralized resources: While centralized components, resources, and function provide usually a useful function, there are grave issues associated with them. Protocol and network design should balance the benefits of centralized resources or control points against the threats arising from them. The general guideline is to avoid such centralized resources when possible. And if it is not possible, find a way to allow the centralized resources to be selectable, depending on context and user settings.
6. Have explicit agreements: When users and their devices provide information to network entities, it would be beneficial to have an opportunity for the users to state their requirements regarding the use of the information provided in this way. While the actual use of such requirements and the willingness of network entities to agree to them remains to be seen, at the moment even the technical means of doing this are limited. For instance, it would be beneficial to be able to embed usage requirements within popular data formats.
7. Treat parties that your equipment connects to with suspicion, even if the communications are encrypted. The other endpoint may misuse any information or control opportunity in the communication. Similarly, even parties within your own system need to be treated with suspicion, as some nodes may become compromised.

8. Do not take any of this as blanket reason to provide no information to anyone, encrypt everything to everyone, or other extreme measures. However, the designers of a system need to be aware of the different threats facing their system, and deal with the most serious ones (of which there are typically many). Similarly, users should be aware of the choices made in a particular design, and avoid designs or products that protect against some threats but are wide open to other serious issues.

## 6. Potential Changes in IETF Analysis of Protocols

### 6.1. Changes in RFC 3552

This memo suggests that changes maybe necessary in RFC 3552. One initial, draft proposal for such changes would be this:

OLD:

In general, we assume that the end-systems engaging in a protocol exchange have not themselves been compromised. Protecting against an attack when one of the end-systems has been compromised is extraordinarily difficult. It is, however, possible to design protocols which minimize the extent of the damage done under these circumstances.

NEW:

In general, we assume that the end-system engaging in a protocol exchange has not itself been compromised. Protecting against an attack of a protocol implementation itself is extraordinarily difficult. It is, however, possible to design protocols which minimize the extent of the damage done when the other parties in a protocol become compromised or do not act in the best interests the end-system implementing a protocol.

In addition, the following new section could be added to discuss the capabilities required to mount an attack:

NEW:

#### 3.x. Other endpoint compromise

In this attack, the other endpoints in the protocol become compromised. As a result, they can, for instance, misuse any information that the end-system implementing a protocol has sent to the compromised endpoint.



## 6.2. Changes in RFC 7258

This memo also suggests that additional guidelines may be necessary in RFC 7258. An initial, draft suggestion for starting point of those changes could be adding the following paragraph after the 2nd paragraph in Section 2:

NEW:

PM attacks include those cases where information collected by a legitimate protocol participant is misused for PM purposes. The attacks also include those cases where a protocol or network architecture results in centralized data storage or control functions relating to many users, raising the risk of said misuse.

## 6.3. System and Architecture Aspects

This definitely needs more attention from Internet technology developers and standards organizations. Here is one possible

The design of any Internet technology should start from an understanding of the participants in a system, their roles, and the extent to which they should have access to information and ability to control other participants.

## 7. Other Work

See, for instance, [I-D.farrell-etm].

## 8. Conclusions

More work is needed in this area. To start with, Internet technology developers need to be better aware of the issues beyond communications security, and consider them in design. At the IETF it would be beneficial to include some of these considerations in the usual systematic security analysis of technologies under development.

In particular, when the IETF develops infrastructure technology for the Internet (such as routing or naming systems), considering the impacts of data generated by those technologies is important. Minimising data collection from users, minimising the parties who get exposed to user data, and protecting data that is relayed or stored in systems should be a priority.

A key focus area at the IETF has been the security of transport protocols, and how transport layer security can be best used to provide the right security for various applications. However, more work is needed in equivalently broadly deployed tools for minimising

or obfuscating information provided by users to other entities, and the use of end-to-end security through entities that are involved in the protocol exchange but who do not need to know everything that is being passed through them.

Comments on the issues discussed in this memo are gladly taken either privately or on the architecture-discuss mailing list.

## 9. Acknowledgements

The author would like to thank John Mattsson, Mirja Kuehlewind, Alissa Cooper, Stephen Farrell, Eric Rescorla, Simone Ferlin, Kathleen Moriarty, Brian Trammell, Mark Nottingham, Christian Huitema, Karl Norrman, Ted Hardie, Mohit Sethi, Phillip Hallam-Baker, Goran Eriksson and the IAB for interesting discussions in this problem space. The author would also like to thank all members of the 2019 Design Expectations vs. Deployment Reality (DEDR) IAB workshop held in Kirkkonummi, Finland.

## 10. Informative References

[I-D.farrell-etm]

Farrell, S., "We're gonna need a bigger threat model", draft-farrell-etm-03 (work in progress), July 2019.

[I-D.iab-wire-image]

Trammell, B. and M. Kuehlewind, "The Wire Image of a Network Protocol", draft-iab-wire-image-01 (work in progress), November 2018.

[I-D.ietf-httpbis-expect-ct]

estark@google.com, e., "Expect-CT Extension for HTTP", draft-ietf-httpbis-expect-ct-08 (work in progress), December 2018.

[I-D.ietf-quic-transport]

Iyengar, J. and M. Thomson, "QUIC: A UDP-Based Multiplexed and Secure Transport", draft-ietf-quic-transport-20 (work in progress), April 2019.

[I-D.ietf-tls-esni]

Rescorla, E., Oku, K., Sullivan, N., and C. Wood, "Encrypted Server Name Indication for TLS 1.3", draft-ietf-tls-esni-03 (work in progress), March 2019.

[I-D.nottingham-for-the-users]

Nottingham, M., "The Internet is for End Users", draft-nottingham-for-the-users-08 (work in progress), June 2019.

- [RFC1958] Carpenter, B., Ed., "Architectural Principles of the Internet", RFC 1958, DOI 10.17487/RFC1958, June 1996, <<https://www.rfc-editor.org/info/rfc1958>>.
- [RFC3552] Rescorla, E. and B. Korver, "Guidelines for Writing RFC Text on Security Considerations", BCP 72, RFC 3552, DOI 10.17487/RFC3552, July 2003, <<https://www.rfc-editor.org/info/rfc3552>>.
- [RFC3935] Alvestrand, H., "A Mission Statement for the IETF", BCP 95, RFC 3935, DOI 10.17487/RFC3935, October 2004, <<https://www.rfc-editor.org/info/rfc3935>>.
- [RFC4655] Farrel, A., Vasseur, J., and J. Ash, "A Path Computation Element (PCE)-Based Architecture", RFC 4655, DOI 10.17487/RFC4655, August 2006, <<https://www.rfc-editor.org/info/rfc4655>>.
- [RFC6480] Lepinski, M. and S. Kent, "An Infrastructure to Support Secure Internet Routing", RFC 6480, DOI 10.17487/RFC6480, February 2012, <<https://www.rfc-editor.org/info/rfc6480>>.
- [RFC6797] Hodges, J., Jackson, C., and A. Barth, "HTTP Strict Transport Security (HSTS)", RFC 6797, DOI 10.17487/RFC6797, November 2012, <<https://www.rfc-editor.org/info/rfc6797>>.
- [RFC6973] Cooper, A., Tschofenig, H., Aboba, B., Peterson, J., Morris, J., Hansen, M., and R. Smith, "Privacy Considerations for Internet Protocols", RFC 6973, DOI 10.17487/RFC6973, July 2013, <<https://www.rfc-editor.org/info/rfc6973>>.
- [RFC7258] Farrell, S. and H. Tschofenig, "Pervasive Monitoring Is an Attack", BCP 188, RFC 7258, DOI 10.17487/RFC7258, May 2014, <<https://www.rfc-editor.org/info/rfc7258>>.
- [RFC7469] Evans, C., Palmer, C., and R. Sleevi, "Public Key Pinning Extension for HTTP", RFC 7469, DOI 10.17487/RFC7469, April 2015, <<https://www.rfc-editor.org/info/rfc7469>>.
- [RFC7540] Belshe, M., Peon, R., and M. Thomson, Ed., "Hypertext Transfer Protocol Version 2 (HTTP/2)", RFC 7540, DOI 10.17487/RFC7540, May 2015, <<https://www.rfc-editor.org/info/rfc7540>>.

- [RFC7817] Melnikov, A., "Updated Transport Layer Security (TLS) Server Identity Check Procedure for Email-Related Protocols", RFC 7817, DOI 10.17487/RFC7817, March 2016, <<https://www.rfc-editor.org/info/rfc7817>>.
- [RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <<https://www.rfc-editor.org/info/rfc8446>>.
- [RFC8484] Hoffman, P. and P. McManus, "DNS Queries over HTTPS (DoH)", RFC 8484, DOI 10.17487/RFC8484, October 2018, <<https://www.rfc-editor.org/info/rfc8484>>.
- [RFC8546] Trammell, B. and M. Kuehlewind, "The Wire Image of a Network Protocol", RFC 8546, DOI 10.17487/RFC8546, April 2019, <<https://www.rfc-editor.org/info/rfc8546>>.
- [RFC8555] Barnes, R., Hoffman-Andrews, J., McCarney, D., and J. Kasten, "Automatic Certificate Management Environment (ACME)", RFC 8555, DOI 10.17487/RFC8555, March 2019, <<https://www.rfc-editor.org/info/rfc8555>>.
- [Saltzer] Saltzer, J., Reed, D., and D. Clark, "End-To-End Arguments in System Design", ACM TOCS, Vol 2, Number 4, pp 277-288 , November 1984.
- [Savage] Savage, S., "Modern Automotive Vulnerabilities: Causes, Disclosures, and Outcomes", USENIX , 2016.

## Author's Address

Jari Arkko  
Ericsson

Email: [jari.arkko@piuha.net](mailto:jari.arkko@piuha.net)

Network Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: March 23, 2020

A. Choudhary  
M. Jethanandani  
Cisco Systems  
N. Strahle  
E. Aries  
Juniper Networks  
I. Chen  
Jabil  
Sep 20, 2019

YANG Model for QoS  
draft-asechoud-rtgwg-qos-model-11

Abstract

This document describes a YANG model for Quality of Service (QoS) configuration and operational parameters.

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 23, 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
1.1. Tree Diagrams . . . . .	3
2. Terminology . . . . .	3
3. QoS Model Design . . . . .	3
4. DiffServ Model Design . . . . .	4
5. Modules Tree Structure . . . . .	4
6. Modules . . . . .	13
6.1. IETF-QOS-CLASSIFIER . . . . .	14
6.2. IETF-QOS-POLICY . . . . .	17
6.3. IETF-QOS-ACTION . . . . .	20
6.4. IETF-QOS-TARGET . . . . .	38
6.5. IETF-DIFFSERV . . . . .	40
6.6. IETF-QUEUE-POLICY . . . . .	50
6.7. IETF-SCHEDULER-POLICY . . . . .	53
7. IANA Considerations . . . . .	56
8. Security Considerations . . . . .	57
9. Acknowledgement . . . . .	57
10. References . . . . .	57
10.1. Normative References . . . . .	57
10.2. Informative References . . . . .	58
Appendix A. Company A, Company B and Company C examples . . . . .	58
A.1. Example of Company A Diffserv Model . . . . .	58
A.2. Example of Company B Diffserv Model . . . . .	68
A.3. Example of Company C Diffserv Model . . . . .	82
Authors' Addresses . . . . .	88

## 1. Introduction

This document defines a base YANG [RFC6020] [RFC7950] data module for Quality of Service (QoS) configuration parameters. Differentiated Services (DiffServ) module is an augmentation of the base QoS model. Remote Procedure Calls (RPC) or notification definition is not part of this document. QoS base modules define a basic building blocks to define a classifier, policy, action and target. The base modules have been augmented to include packet match fields and action parameters to define the DiffServ module. Queues and schedulers are stitched as part of diffserv policy itself or separate modules are defined for creating Queue policy and Scheduling policy. The DiffServ model is based on DiffServ architecture, and various references have been made to available standard architecture documents.

DiffServ is a preferred approach for network service providers to offer services to different customers based on their network Quality-of-Service (QoS) objectives. The traffic streams are differentiated based on DiffServ Code Points (DSCP) carried in the IP header of each packet. The DSCP markings are applied by upstream node or by the edge router on entry to the DiffServ network.

Editorial Note: (To be removed by RFC Editor)

This draft contains several placeholder values that need to be replaced with finalized values at the time of publication. Please apply the following replacements: o "XXXX" --> the assigned RFC value for this draft both in this draft and in the YANG models under the revision statement. o The "revision" date in model, in the format XXXX-XX-XX, needs to be updated with the date the draft gets approved.

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

### 1.1. Tree Diagrams

Tree diagrams used in this document follow the notation defined in [RFC8340 [RFC8340]]

## 2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

## 3. QoS Model Design

A classifier consists of packets which may be grouped when a logical set of rules are applied on different packet header fields. The grouping may be based on different values or range of values of same packet header field, presence or absence of some values or range of values of a packet field or a combination thereof. The QoS classifier is defined in the ietf-qos-classifier module.

A classifier entry contains one or more packet conditioning functions. A packet conditioning function is typically based on direction of traffic and may drop, mark or delay network packets. A set of classifier entries with corresponding conditioning functions when arranged in order of priority represents a QoS policy. A QoS

policy may contain one or more classifier entries. These are defined in ietf-qos-policy module.

Actions are configured in line with respect to the policy module. These include marking, dropping or shaping. Actions are defined in the ietf-qos-action module.

A meter qualifies if the traffic arrival rate is based on agreed upon rate and variability. A meter is modeled based on commonly used algorithms in industry, Single Rate Tri Color Marking (srTCM) [RFC2697] meter, Two Rate Tri Color Marking (trTCM) [RFC2698] meter, and Single Rate Two Color Marking meter. Different vendors can extend it with other types of meters as well.

#### 4. DiffServ Model Design

DiffServ architecture [RFC3289] and [RFC2475] describe the architecture as a simple model where traffic entering a network is classified and possibly conditioned at the boundary of the network and assigned a different Behavior Aggregate (BA). Each BA is identified by a specific value of DSCP, and is used to select a Per Hop Behavior (PHB).

The packet classification policy identifies the subset of traffic which may receive a DiffServ by being conditioned or mapped. Packet classifiers select packets within a stream based on the content of some portion of the packet header. There are two types of classifiers, the BA classifier, and the Multi-Field (MF) classifier which selects packets based on a value which is combination of one or more header fields. In the ietf-diffserv module, this is realized by augmenting the QoS classification module.

Traffic conditioning includes metering, shaping and/or marking. A meter is used to measure the traffic against a given traffic profile. The traffic profile specifies the temporal property of the traffic. A packet that arrives is first determined to be in or out of the profile, which will result in the action of marked, dropped or shaped. This is realized in vendor specific modules based on the parameters defined in action module. The metering parameters are augmented to the QoS policy module when metering is defined inline, and to the metering template when metering profile is referred in policy module.

#### 5. Modules Tree Structure

This document defines seven YANG modules - four QoS base modules, a scheduler policy module, a queuing policy module and one DiffServ module.



ietf-qos-classifier consists of classifier entries identified by a classifier entry name. Each entry MAY contain a list of filter entries. When no filter entry is present in a classifier entry, it matches all traffic.

```

module: ietf-qos-classifier
  +--rw classifiers
    +--rw classifier-entry* [classifier-entry-name]
      +--rw classifier-entry-name      string
      +--rw classifier-entry-descr?    string
      +--rw classifier-entry-filter-operation? identityref
      +--rw filter-entry* [filter-type filter-logical-not]
        +--rw filter-type              identityref
        +--rw filter-logical-not       boolean

```

An ietf-qos-policy module contains list of policy objects identified by a policy name and policy type which MUST be provided. With different values of policy types, each vendor MAY define their own construct of policy for different QoS functionalities. Each vendor MAY augment classifier entry in a policy definition with a set of actions.

```

module: ietf-qos-policy
  +--rw policies
    +--rw policy-entry* [policy-name policy-type]
      +--rw policy-name      string
      +--rw policy-type      identityref
      +--rw policy-descr?    string
      +--rw classifier-entry* [classifier-entry-name]
        +--rw classifier-entry-name      string
        +--rw classifier-entry-inline?    boolean
        +--rw classifier-entry-filter-oper? identityref
        +--rw filter-entry* [filter-type filter-logical-not]
          {policy-inline-classifier-config}?
          +--rw filter-type      identityref
          +--rw filter-logical-not boolean
        +--rw classifier-action-entry-cfg* [action-type]
          +--rw action-type      identityref
          +--rw (action-cfg-params)?

```

ietf-qos-action module contains grouping of set of QoS actions. These include metering, marking, dropping and shaping. Marking sets DiffServ codepoint value in the classified packet. Color-aware and Color-blind meters are augmented by vendor specific modules based on the parameters defined in action module.

```

module: ietf-qos-action
  +--rw meter-template
    +--rw meter-entry* [meter-name] {meter-template-support}?
      +--rw meter-name          string
      +--rw (meter-type)?
        +--:(one-rate-two-color-meter-type)
          +--rw one-rate-two-color-meter
            +--rw committed-rate-value?    uint64
            +--rw committed-rate-unit?     identityref
            +--rw committed-burst-value?   uint64
            +--rw committed-burst-unit?    identityref
            +--rw conform-action
              | +--rw conform-2color-meter-action-params*
              | | [conform-2color-meter-action-type]
              | | +--rw conform-2color-meter-action-type
              | | | identityref
              | | +--rw (conform-2color-meter-action-val)?
            +--rw exceed-action
              | +--rw exceed-2color-meter-action-params*
              | | [exceed-2color-meter-action-type]
              | | +--rw exceed-2color-meter-action-type
              | | | identityref
              | | +--rw (exceed-2color-meter-action-val)?
        +--:(one-rate-tri-color-meter-type)
          +--rw one-rate-tri-color-meter
            +--rw committed-rate-value?    uint64
            +--rw committed-rate-unit?     identityref
            +--rw committed-burst-value?   uint64
            +--rw committed-burst-unit?    identityref
            +--rw excess-burst-value?      uint64
            +--rw excess-burst-unit?       identityref
            +--rw conform-action
              | +--rw conform-3color-meter-action-params*
              | | [conform-3color-meter-action-type]
              | | +--rw conform-3color-meter-action-type
              | | | identityref
              | | +--rw (conform-3color-meter-action-val)?
            +--rw exceed-action
              | +--rw exceed-3color-meter-action-params*
              | | [exceed-3color-meter-action-type]
              | | +--rw exceed-3color-meter-action-type
              | | | identityref
              | | +--rw (exceed-3color-meter-action-val)?
            +--rw violate-action
              | +--rw violate-3color-meter-action-params*
              | | [violate-3color-meter-action-type]
              | | +--rw violate-3color-meter-action-type
              | | | identityref

```

```

|           +---rw (violate-3color-meter-action-val)?
+---:(two-rate-tri-color-meter-type)
  +---rw two-rate-tri-color-meter
    +---rw committed-rate-value?      uint64
    +---rw committed-rate-unit?       identityref
    +---rw committed-burst-value?     uint64
    +---rw committed-burst-unit?     identityref
    +---rw peak-rate-value?           uint64
    +---rw peak-rate-unit?            identityref
    +---rw peak-burst-value?          uint64
    +---rw peak-burst-unit?           identityref
    +---rw conform-action
    |   +---rw conform-3color-meter-action-params*
    |       [conform-3color-meter-action-type]
    |   +---rw conform-3color-meter-action-type
    |       identityref
    |   +---rw (conform-3color-meter-action-val)?
    +---rw exceed-action
    |   +---rw exceed-3color-meter-action-params*
    |       [exceed-3color-meter-action-type]
    |   +---rw exceed-3color-meter-action-type
    |       identityref
    |   +---rw (exceed-3color-meter-action-val)?
    +---rw violate-action
    |   +---rw violate-3color-meter-action-params*
    |       [violate-3color-meter-action-type]
    |   +---rw violate-3color-meter-action-type
    |       identityref
    |   +---rw (violate-3color-meter-action-val)?

```

ietf-qos-target module contains reference of qos-policy and augments ietf-interfaces [RFC8343] module. A single policy of a particular policy-type can be applied on an interface in each direction of traffic. Policy-type is of type identity and is populated in a vendor specific manner. This way it provides greater flexibility for each vendor to define different policy types each with its own capabilities and restrictions.

Classifier, metering and queuing counters are associated with a target.

```

module: ietf-qos-target
augment /if:interfaces/if:interface:
  +---rw qos-target-entry* [direction policy-type]
    +---rw direction      identityref
    +---rw policy-type     identityref
    +---rw policy-name     string

```

Diffserv module augments QoS classifier module. Many of the YANG types defined in [RFC6991] are represented as leafs in the classifier module.

Metering and marking actions are realized by augmenting the QoS policy-module. Any queuing, AQM and scheduling actions are part of vendor specific augmentation. Statistics are realized by augmenting the QoS target module.

```

module: ietf-diffserv
  augment /classifier:classifiers/classifier:classifier-entry +
    /classifier:filter-entry:
    +--rw (filter-param)?
      +--:(dscp)
        +--rw dscp-cfg* [dscp-min dscp-max]
          +--rw dscp-min      inet:dscp
          +--rw dscp-max      inet:dscp
      +--:(source-ipv4-address)
        +--rw source-ipv4-address-cfg* [source-ipv4-addr]
          +--rw source-ipv4-addr      inet:ipv4-prefix
      +--:(destination-ipv4-address)
        +--rw destination-ipv4-address-cfg* [destination-ipv4-addr]
          +--rw destination-ipv4-addr      inet:ipv4-prefix
      +--:(source-ipv6-address)
        +--rw source-ipv6-address-cfg* [source-ipv6-addr]
          +--rw source-ipv6-addr      inet:ipv6-prefix
      +--:(destination-ipv6-address)
        +--rw destination-ipv6-address-cfg* [destination-ipv6-addr]
          +--rw destination-ipv6-addr      inet:ipv6-prefix
      +--:(source-port)
        +--rw source-port-cfg* [source-port-min source-port-max]
          +--rw source-port-min      inet:port-number
          +--rw source-port-max      inet:port-number
      +--:(destination-port)
        +--rw destination-port-cfg*
          [destination-port-min destination-port-max]
          +--rw destination-port-min      inet:port-number
          +--rw destination-port-max      inet:port-number
      +--:(protocol)
        +--rw protocol-cfg* [protocol-min protocol-max]
          +--rw protocol-min      uint8
          +--rw protocol-max      uint8
      +--:(traffic-group)
        +--rw traffic-group-cfg
        +--rw traffic-group-name?      string
  augment /policy:policies/policy:policy-entry +
    /policy:classifier-entry/policy:filter-entry:
    +--rw (filter-params)?

```

```

+---:(dscp)
|   +---rw dscp-cfg* [dscp-min dscp-max]
|       +---rw dscp-min      inet:dscp
|       +---rw dscp-max      inet:dscp
+---:(source-ipv4-address)
|   +---rw source-ipv4-address-cfg* [source-ipv4-addr]
|       +---rw source-ipv4-addr      inet:ipv4-prefix
+---:(destination-ipv4-address)
|   +---rw destination-ipv4-address-cfg* [destination-ipv4-addr]
|       +---rw destination-ipv4-addr      inet:ipv4-prefix
+---:(source-ipv6-address)
|   +---rw source-ipv6-address-cfg* [source-ipv6-addr]
|       +---rw source-ipv6-addr      inet:ipv6-prefix
+---:(destination-ipv6-address)
|   +---rw destination-ipv6-address-cfg* [destination-ipv6-addr]
|       +---rw destination-ipv6-addr      inet:ipv6-prefix
+---:(source-port)
|   +---rw source-port-cfg* [source-port-min source-port-max]
|       +---rw source-port-min      inet:port-number
|       +---rw source-port-max      inet:port-number
+---:(destination-port)
|   +---rw destination-port-cfg*
|       [destination-port-min destination-port-max]
|       +---rw destination-port-min      inet:port-number
|       +---rw destination-port-max      inet:port-number
+---:(protocol)
|   +---rw protocol-cfg* [protocol-min protocol-max]
|       +---rw protocol-min      uint8
|       +---rw protocol-max      uint8
+---:(traffic-group)
|   +---rw traffic-group-cfg
|       +---rw traffic-group-name?      string
augment /policy:policies/policy:policy-entry +
/policy:classifier-entry +
/policy:classifier-action-entry-cfg +
/policy:action-cfg-params:
+---:(dscp-marking)
|   +---rw dscp-cfg
|       +---rw dscp?      inet:dscp
+---:(meter-inline) {action:meter-inline-feature}?
|   +---rw (meter-type)?
|       +---:(one-rate-two-color-meter-type)
|           +---rw one-rate-two-color-meter
|               +---rw committed-rate-value?      uint64
|               +---rw committed-rate-unit?      identityref
|               +---rw committed-burst-value?      uint64
|               +---rw committed-burst-unit?      identityref
|               +---rw conform-action

```

```

|         |         |   +---rw conform-2color-meter-action-params*
|         |         |       [conform-2color-meter-action-type]
|         |         |   +---rw conform-2color-meter-action-type
|         |         |       identityref
|         |         |   +---rw (conform-2color-meter-action-val)?
|         |         +---rw exceed-action
|         |         |   +---rw exceed-2color-meter-action-params*
|         |         |       [exceed-2color-meter-action-type]
|         |         |   +---rw exceed-2color-meter-action-type
|         |         |       identityref
|         |         |   +---rw (exceed-2color-meter-action-val)?
+---: (one-rate-tri-color-meter-type)
|         |         +---rw one-rate-tri-color-meter
|         |         |   +---rw committed-rate-value?         uint64
|         |         |   +---rw committed-rate-unit?         identityref
|         |         |   +---rw committed-burst-value?        uint64
|         |         |   +---rw committed-burst-unit?         identityref
|         |         |   +---rw excess-burst-value?           uint64
|         |         |   +---rw excess-burst-unit?            identityref
|         |         |   +---rw conform-action
|         |         |       +---rw conform-3color-meter-action-params*
|         |         |           [conform-3color-meter-action-type]
|         |         |       +---rw conform-3color-meter-action-type
|         |         |           identityref
|         |         |       +---rw (conform-3color-meter-action-val)?
+---rw exceed-action
|         |         |   +---rw exceed-3color-meter-action-params*
|         |         |       [exceed-3color-meter-action-type]
|         |         |   +---rw exceed-3color-meter-action-type
|         |         |       identityref
|         |         |   +---rw (exceed-3color-meter-action-val)?
+---rw violate-action
|         |         |   +---rw violate-3color-meter-action-params*
|         |         |       [violate-3color-meter-action-type]
|         |         |   +---rw violate-3color-meter-action-type
|         |         |       identityref
|         |         |   +---rw (violate-3color-meter-action-val)?
+---: (two-rate-tri-color-meter-type)
|         |         +---rw two-rate-tri-color-meter
|         |         |   +---rw committed-rate-value?         uint64
|         |         |   +---rw committed-rate-unit?         identityref
|         |         |   +---rw committed-burst-value?        uint64
|         |         |   +---rw committed-burst-unit?         identityref
|         |         |   +---rw peak-rate-value?              uint64
|         |         |   +---rw peak-rate-unit?               identityref
|         |         |   +---rw peak-burst-value?              uint64
|         |         |   +---rw peak-burst-unit?              identityref
|         |         |   +---rw conform-action

```

```

|         |   +---rw conform-3color-meter-action-params*
|         |           [conform-3color-meter-action-type]
|         |   +---rw conform-3color-meter-action-type
|         |           identityref
|         |   +---rw (conform-3color-meter-action-val)?
+---rw exceed-action
|         |   +---rw exceed-3color-meter-action-params*
|         |           [exceed-3color-meter-action-type]
|         |   +---rw exceed-3color-meter-action-type
|         |           identityref
|         |   +---rw (exceed-3color-meter-action-val)?
+---rw violate-action
|         |   +---rw violate-3color-meter-action-params*
|         |           [violate-3color-meter-action-type]
|         |   +---rw violate-3color-meter-action-type
|         |           identityref
|         |   +---rw (violate-3color-meter-action-val)?
+---:(meter-reference) {action:meter-reference-feature}?
|   +---rw meter-reference-cfg
|       +---rw meter-reference-name      string
|       +---rw meter-type                identityref
+---:(traffic-group-marking) {action:traffic-group-feature}?
|   +---rw traffic-group-cfg
|       +---rw traffic-group?    string
+---:(child-policy) {action:child-policy-feature}?
|   +---rw child-policy-cfg {child-policy-feature}?
|       +---rw policy-name?    string
+---:(count) {action:count-feature}?
|   +---rw count-cfg {count-feature}?
|       +---rw count-action?    empty
+---:(named-count) {action:named-counter-feature}?
|   +---rw named-counter-cfg {named-counter-feature}?
|       +---rw count-name-action?    string
+---:(queue-inline) {diffserv-queue-inline-support}?
|   +---rw queue-cfg
|       +---rw priority-cfg
|           |   +---rw priority-level?    uint8
+---rw min-rate-cfg
|       +---rw rate-value?    uint64
|       +---rw rate-unit?    identityref
+---rw max-rate-cfg
|       +---rw rate-value?    uint64
|       +---rw rate-unit?    identityref
|       +---rw burst-value?    uint64
|       +---rw burst-unit?    identityref
+---rw algorithmic-drop-cfg
|       +---rw (drop-algorithm)?
|           +---:(tail-drop)

```

```

|           +---rw tail-drop-cfg
|           +---rw tail-drop-alg?   empty
+---:(scheduler-inline) {diffserv-scheduler-inline-support}?
+---rw scheduler-cfg
|   +---rw min-rate-cfg
|   |   +---rw rate-value?   uint64
|   |   +---rw rate-unit?   identityref
|   +---rw max-rate-cfg
|   |   +---rw rate-value?   uint64
|   |   +---rw rate-unit?   identityref
|   |   +---rw burst-value?  uint64
|   |   +---rw burst-unit?  identityref
module: ietf-queue-policy
+---rw queue-template {queue-policy-support}?
+---rw name?          string
+---rw queue-cfg
|   +---rw priority-cfg
|   |   +---rw priority-level?  uint8
|   +---rw min-rate-cfg
|   |   +---rw rate-value?   uint64
|   |   +---rw rate-unit?   identityref
|   +---rw max-rate-cfg
|   |   +---rw rate-value?   uint64
|   |   +---rw rate-unit?   identityref
|   |   +---rw burst-value?  uint64
|   |   +---rw burst-unit?  identityref
+---rw algorithmic-drop-cfg
|   +---rw (drop-algorithm)?
|   |   +---:(tail-drop)
|   |   |   +---rw tail-drop-cfg
|   |   |   |   +---rw tail-drop-alg?   empty
augment /policy:policies/policy:policy-entry +
/policy:classifier-entry/policy:filter-entry:
+---rw (filter-params)? {queue-policy-support}?
+---:(traffic-group-name)
|   +---rw traffic-group-reference-cfg
|   |   +---rw traffic-group-name   string
augment /policy:policies/policy:policy-entry +
/policy:classifier-entry +
/policy:classifier-action-entry-cfg +
/policy:action-cfg-params:
+---:(queue-template-name)
|   {queue-template-support,queue-policy-support}?
|   |   +---rw queue-template-reference-cfg
|   |   |   +---rw queue-template-name   string
+---:(queue-inline)
|   {queue-inline-support,queue-policy-support}?

```



```

    +---rw queue-cfg
    |   +---rw priority-cfg
    |   |   +---rw priority-level?    uint8
    +---rw min-rate-cfg
    |   +---rw rate-value?            uint64
    |   +---rw rate-unit?             identityref
    +---rw max-rate-cfg
    |   +---rw rate-value?            uint64
    |   +---rw rate-unit?             identityref
    |   +---rw burst-value?           uint64
    |   +---rw burst-unit?            identityref
    +---rw algorithmic-drop-cfg
    |   +---rw (drop-algorithm)?
    |   |   +---:(tail-drop)
    |   |   |   +---rw tail-drop-cfg
    |   |   |   |   +---rw tail-drop-alg?    empty
    module: ietf-scheduler-policy
    augment /policy:policies/policy:policy-entry +
        /policy:classifier-entry/policy:filter-entry:
    +---rw (filter-params)?
    +---:(filter-match-all)
    +---rw match-all-cfg
    +---rw match-all-action?    empty
    augment /policy:policies/policy:policy-entry +
        /policy:classifier-entry +
        /policy:classifier-action-entry-cfg +
        /policy:action-cfg-params:
    +---:(scheduler)
    |   +---rw scheduler-cfg
    |   |   +---rw min-rate-cfg
    |   |   |   +---rw rate-value?            uint64
    |   |   |   +---rw rate-unit?             identityref
    |   +---rw max-rate-cfg
    |   |   +---rw rate-value?            uint64
    |   |   +---rw rate-unit?             identityref
    |   |   +---rw burst-value?           uint64
    |   |   +---rw burst-unit?            identityref
    +---:(queue-policy-name)
    |   +---rw queue-policy-name
    |   |   +---rw queue-policy    string

```

## 6. Modules

## 6.1. IETF-QOS-CLASSIFIER

```
<CODE BEGINS>file "ietf-qos-classifier@2019-03-13.yang"
module iETF-qos-classifier {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-qos-classifier";
  prefix classifier;

  organization
    "IETF RTG (Routing Area) Working Group";
  contact
    "WG Web:    <http://tools.ietf.org/wg/rtgwg/>
    WG List:    <mailto:rtgwg@ietf.org>
    WG Chair:   Chris Bowers
                <mailto:cbowers@juniper.net>
    WG Chair:   Jeff Tantsura
                <mailto:jefftant.ietf@gmail.com>
    Editor:     Aseem Choudhary
                <mailto:asechoud@cisco.com>
    Editor:     Mahesh Jethanandani
                <mailto:mjethanandani@gmail.com>
    Editor:     Norm Strahle
                <mailto:nstrahle@juniper.net>";
  description
    "This module contains a collection of YANG definitions for
    configuring qos specification implementations.
    Copyright (c) 2019 IETF Trust and the persons identified as
    authors of the code. All rights reserved.
    Redistribution and use in source and binary forms, with or
    without modification, is permitted pursuant to, and subject
    to the license terms contained in, the Simplified BSD License
    set forth in Section 4.c of the IETF Trust's Legal Provisions
    Relating to IETF Documents
    (http://trustee.ietf.org/license-info).
    This version of this YANG module is part of RFC XXXX; see
    the RFC itself for full legal notices.";

  revision 2019-03-13 {
    description
      "Latest revision of qos base classifier module";
    reference "RFC XXXX: YANG Model for QoS";
  }

  feature policy-inline-classifier-config {
    description
      " This feature allows classifier configuration
      directly under policy.";
  }
```

```
feature classifier-template-feature {
  description
    " This feature allows classifier as template configuration
      in a policy.";
}

feature match-any-filter-type-support {
  description
    " This feature allows classifier configuration
      directly under policy.";
}

identity filter-type {
  description
    "This is identity of base filter-type";
}

identity classifier-entry-filter-operation-type {
  description
    "Classifier entry filter logical operation";
}

identity match-all-filter {
  base classifier-entry-filter-operation-type;
  description
    "Classifier entry filter logical AND operation";
}

identity match-any-filter {
  base classifier-entry-filter-operation-type;
  if-feature "match-any-filter-type-support";
  description
    "Classifier entry filter logical OR operation";
}

grouping filters {
  description
    "Filters types in a Classifier entry";
  leaf filter-type {
    type identityref {
      base filter-type;
    }
    description
      "This leaf defines type of the filter";
  }
  leaf filter-logical-not {
    type boolean;
    description

```

```
        "
        This is logical-not operator for a filter. When true, it
        indicates filter looks for absence of a pattern defined
        by the filter
        ";
    }
}

grouping classifier-entry-generic-attr {
    description
        "
        Classifier generic attributes like name,
        description, operation type
        ";
    leaf classifier-entry-name {
        type string;
        description
            "classifier entry name";
    }
    leaf classifier-entry-descr {
        type string;
        description
            "classifier entry description statement";
    }
    leaf classifier-entry-filter-operation {
        type identityref {
            base classifier-entry-filter-operation-type;
        }
        default "match-all-filter";
        description
            "Filters are applicable as match-any or match-all filters";
    }
}

grouping classifier-entry-inline-attr {
    description
        "attributes of inline classifier in a policy";
    leaf classifier-entry-inline {
        type boolean;
        default "false";
        description
            "Indication of inline classifier entry";
    }
    leaf classifier-entry-filter-oper {
        type identityref {
            base classifier-entry-filter-operation-type;
        }
        default "match-all-filter";
    }
}
```

```

        description
            "Filters are applicable as match-any or match-all filters";
    }
    list filter-entry {
        if-feature "policy-inline-classifier-config";
        must " ../classifier-entry-inline = 'true' " {
            description
                "For inline filter configuration, inline attributemust
                be true";
        }
        key "filter-type filter-logical-not";
        uses filters;
        description
            "Filters configured inline in a policy";
    }
}

container classifiers {
    if-feature "classifier-template-feature";
    description
        "list of classifier entry";
    list classifier-entry {
        key "classifier-entry-name";
        description
            "each classifier entry contains a list of filters";
        uses classifier-entry-generic-attr;
        list filter-entry {
            key "filter-type filter-logical-not";
            uses filters;
            description
                "Filter entry configuration";
        }
    }
}
}
}
}
<CODE ENDS>

```

## 6.2. IETF-QOS-POLICY

```

<CODE BEGINS>file "ietf-qos-policy@2019-03-13.yang"
module iETF-qos-policy {
    yang-version 1.1;
    namespace "urn:ietf:params:xml:ns:yang:ietf-qos-policy";
    prefix policy;
    import iETF-qos-classifier {
        prefix classifier;
        reference "RFC XXXX: YANG Model for QoS";
    }
}

```

```
organization "IETF RTG (Routing Area) Working Group";
contact
  "WG Web:    <http://tools.ietf.org/wg/rtgwg/>
  WG List:    <mailto:rtgwg@ietf.org>
  WG Chair:   Chris Bowers
              <mailto:cbowers@juniper.net>
  WG Chair:   Jeff Tantsura
              <mailto:jefftant.ietf@gmail.com>
  Editor:     Aseem Choudhary
              <mailto:asechoud@cisco.com>
  Editor:     Mahesh Jethanandani
              <mailto:mjethanandani@gmail.com>
  Editor:     Norm Strahle
              <mailto:nstrahle@juniper.net>";
description
  "This module contains a collection of YANG definitions for
  configuring qos specification implementations.
  Copyright (c) 2019 IETF Trust and the persons identified as
  authors of the code. All rights reserved.
  Redistribution and use in source and binary forms, with or
  without modification, is permitted pursuant to, and subject
  to the license terms contained in, the Simplified BSD License
  set forth in Section 4.c of the IETF Trust's Legal Provisions
  Relating to IETF Documents
  (http://trustee.ietf.org/license-info).
  This version of this YANG module is part of RFC XXXX; see
  the RFC itself for full legal notices.";
revision 2019-03-13 {
  description
    "Latest revision of qos policy";
  reference "RFC XXXX: YANG Model for QoS";
}
identity policy-type {
  description
    "This base identity type defines policy-types";
}
grouping policy-generic-attr {
  description
    "Policy Attributes";
  leaf policy-name {
    type string;
    description
      "policy name";
  }
  leaf policy-type {
    type identityref {
      base policy-type;
    }
  }
}
```

```
        description
            "policy type";
    }
    leaf policy-descr {
        type string;
        description
            "policy description";
    }
}
identity action-type {
    description
        "This base identity type defines action-types";
}
grouping classifier-action-entry-cfg {
    description
        "List of Configuration of classifier & associated actions";
    list classifier-action-entry-cfg {
        key "action-type";
        ordered-by user;
        description
            "Configuration of classifier & associated actions";
        leaf action-type {
            type identityref {
                base action-type;
            }
            description
                "This defines action type ";
        }
        choice action-cfg-params {
            description
                "Choice of action types";
        }
    }
}
container policies {
    description
        "list of policy templates";
    list policy-entry {
        key "policy-name policy-type";
        description
            "policy template";
        uses policy-generic-attr;
        list classifier-entry {
            key "classifier-entry-name";
            ordered-by user;
            description
                "Classifier entry configuration in a policy";
            leaf classifier-entry-name {
```

```

        type string;
        description
            "classifier entry name";
    }
    uses classifier:classifier-entry-inline-attr;
    uses classifier-action-entry-cfg;
}
}
}
}
}
<CODE ENDS>

```

### 6.3. IETF-QOS-ACTION

```

<CODE BEGINS>file "ietf-qos-action@2019-03-13.yang"
module iETF-qos-action {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-qos-action";
  prefix action;
  import iETF-inet-types {
    prefix inet;
    reference "RFC 6991: Common YANG Data Types";
  }
  import iETF-qos-policy {
    prefix policy;
    reference "RFC XXXX: YANG Model for QoS";
  }
  organization "IETF RTG (Routing Area) Working Group";
  contact
    "WG Web:  <http://tools.ietf.org/wg/rtgwg/>
    WG List:  <mailto:rtgwg@ietf.org>
    WG Chair: Chris Bowers
              <mailto:cbowers@juniper.net>
    WG Chair: Jeff Tantsura
              <mailto:jefftant.ietf@gmail.com>
    Editor:   Aseem Choudhary
              <mailto:asechoud@cisco.com>
    Editor:   Mahesh Jethanandani
              <mailto:mjethanandani@gmail.com>
    Editor:   Norm Strahle
              <mailto:nstrahle@juniper.net>";
  description
    "This module contains a collection of YANG definitions for
    configuring qos specification implementations.
    Copyright (c) 2019 IETF Trust and the persons identified as
    authors of the code. All rights reserved.
    Redistribution and use in source and binary forms, with or
    without modification, is permitted pursuant to, and subject

```



```
to the license terms contained in, the Simplified BSD License
set forth in Section 4.c of the IETF Trust's Legal Provisions
Relating to IETF Documents
(http://trustee.ietf.org/license-info).
This version of this YANG module is part of RFC XXXX; see
the RFC itself for full legal notices.";
revision 2019-03-13 {
  description
    "Latest revision for qos actions";
  reference "RFC XXXX: YANG Model for QoS";
}
feature meter-template-support {
  description
    " This feature allows support of meter-template.";
}
feature meter-inline-feature {
  description
    "This feature allows support of meter-inline configuration.";
}
feature meter-reference-feature {
  description
    "This feature allows support of meter by reference
    configuration.";
}
feature queue-action-support {
  description
    " This feature allows support of queue action configuration
    in policy.";
}
feature scheduler-action-support {
  description
    " This feature allows support of scheduler configuration
    in policy.";
}
feature child-policy-feature {
  description
    " This feature allows configuration of hierarchical policy.";
}
feature count-feature {
  description
    "This feature allows action configuration to enable
    counter in a classifier";
}
feature named-counter-feature {
  description
    "This feature allows action configuration to enable
    named counter in a classifier";
}
```

```
feature traffic-group-feature {
  description
    "traffic-group action support";
}
feature burst-time-unit-support {
  description
    "This feature allows burst unit to be configured as
    time duration.";
}

identity rate-unit-type {
  description
    "base rate-unit type";
}
identity bits-per-second {
  base rate-unit-type;
  description
    "bits per second identity";
}
identity kilo-bits-per-second {
  base rate-unit-type;
  description
    "kilo bits per second identity";
}
identity mega-bits-per-second {
  base rate-unit-type;
  description
    "mega bits per second identity";
}
identity giga-bits-per-second {
  base rate-unit-type;
  description
    "mega bits per second identity";
}
identity percent {
  base rate-unit-type;
  description
    "percentage";
}
identity burst-unit-type {
  description
    "base burst-unit type";
}
identity bytes {
  base burst-unit-type;
  description
    "bytes";
}
```

```
identity kilo-bytes {
  base burst-unit-type;
  description
    "kilo bytes";
}
identity mega-bytes {
  base burst-unit-type;
  description
    "mega bytes";
}
identity millisecond {
  base burst-unit-type;
  if-feature burst-time-unit-support;
  description
    "milli seconds";
}
identity microsecond {
  base burst-unit-type;
  if-feature burst-time-unit-support;
  description
    "micro seconds";
}
identity dscp-marking {
  base policy:action-type;
  description
    "dscp marking action type";
}
identity meter-inline {
  base policy:action-type;
  if-feature meter-inline-feature;
  description
    "meter-inline action type";
}
identity meter-reference {
  base policy:action-type;
  if-feature meter-reference-feature;
  description
    "meter reference action type";
}
identity queue {
  base policy:action-type;
  if-feature queue-action-support;
  description
    "queue action type";
}
identity scheduler {
  base policy:action-type;
  if-feature scheduler-action-support;
```

```
    description
      "scheduler action type";
  }
  identity discard {
    base policy:action-type;
    description
      "discard action type";
  }
  identity child-policy {
    base policy:action-type;
    if-feature child-policy-feature;
    description
      "child-policy action type";
  }
  identity count {
    base policy:action-type;
    if-feature count-feature;
    description
      "count action type";
  }
  identity named-counter {
    base policy:action-type;
    if-feature named-counter-feature;
    description
      "name counter action type";
  }
  identity meter-type {
    description
      "This base identity type defines meter types";
  }
  identity one-rate-two-color-meter-type {
    base meter-type;
    description
      "one rate two color meter type";
  }
  identity one-rate-tri-color-meter-type {
    base meter-type;
    description
      "one rate three color meter type";
    reference
      "RFC2697: A Single Rate Three Color Marker";
  }
  identity two-rate-tri-color-meter-type {
    base meter-type;
    description
      "two rate three color meter action type";
    reference
```

```
    "RFC2698: A Two Rate Three Color Marker";
}

identity drop-type {
  description
    "drop algorithm";
}
identity tail-drop {
  base drop-type;
  description
    "tail drop algorithm";
}

identity conform-2color-meter-action-type {
  description
    "action type in a meter";
}
identity exceed-2color-meter-action-type {
  description
    "action type in a meter";
}
identity conform-3color-meter-action-type {
  description
    "action type in a meter";
}
identity exceed-3color-meter-action-type {
  description
    "action type in a meter";
}
identity violate-3color-meter-action-type {
  description
    "action type in a meter";
}

grouping rate-value-unit {
  leaf rate-value {
    type uint64;
    description
      "rate value";
  }
  leaf rate-unit {
    type identityref {
      base rate-unit-type;
    }
    description
      "rate unit";
  }
  description

```

```
        "rate value and unit grouping";
    }
    grouping burst {
        description
            "burst value and unit configuration";
        leaf burst-value {
            type uint64;
            description
                "burst value";
        }
        leaf burst-unit {
            type identityref {
                base burst-unit-type;
            }
            description
                "burst unit";
        }
    }
}

grouping threshold {
    description
        "Threshold Parameters";
    container threshold {
        description
            "threshold";
        choice threshold-type {
            case size {
                leaf threshold-size {
                    type uint64;
                    units "bytes";
                    description
                        "Threshold size";
                }
            }
            case interval {
                leaf threshold-interval {
                    type uint64;
                    units "microsecond";
                    description
                        "Threshold interval";
                }
            }
        }
        description
            "Choice of threshold type";
    }
}
}
```

```
grouping drop {
  container drop-cfg {
    leaf drop-action {
      type empty;
      description
        "always drop algorithm";
    }
    description
      "the drop action";
  }
  description
    "always drop grouping";
}

grouping queuelimit {
  container qlimit-thresh {
    uses threshold;
    description
      "the queue limit";
  }
  description
    "the queue limit beyond which queue will not hold any packet";
}

grouping conform-2color-meter-action-params {
  description
    "meter action parameters";
  list conform-2color-meter-action-params {
    key "conform-2color-meter-action-type";
    ordered-by user;
    description
      "Configuration of basic-meter & associated actions";
    leaf conform-2color-meter-action-type {
      type identityref {
        base conform-2color-meter-action-type;
      }
      description
        "meter action type";
    }
    choice conform-2color-meter-action-val {
      description
        " meter action based on choice of meter action type";
    }
  }
}

grouping exceed-2color-meter-action-params {
  description
```

```
    "meter action parameters";
  list exceed-2color-meter-action-params {
    key "exceed-2color-meter-action-type";
    ordered-by user;
    description
      "Configuration of basic-meter & associated actions";
    leaf exceed-2color-meter-action-type {
      type identityref {
        base exceed-2color-meter-action-type;
      }
      description
        "meter action type";
    }
    choice exceed-2color-meter-action-val {
      description
        " meter action based on choice of meter action type";
    }
  }
}

grouping conform-3color-meter-action-params {
  description
    "meter action parameters";
  list conform-3color-meter-action-params {
    key "conform-3color-meter-action-type";
    ordered-by user;
    description
      "Configuration of basic-meter & associated actions";
    leaf conform-3color-meter-action-type {
      type identityref {
        base conform-3color-meter-action-type;
      }
      description
        "meter action type";
    }
    choice conform-3color-meter-action-val {
      description
        " meter action based on choice of meter action type";
    }
  }
}

grouping exceed-3color-meter-action-params {
  description
    "meter action parameters";
  list exceed-3color-meter-action-params {
    key "exceed-3color-meter-action-type";
```



```
    ordered-by user;
    description
      "Configuration of basic-meter & associated actions";
    leaf exceed-3color-meter-action-type {
      type identityref {
        base exceed-3color-meter-action-type;
      }
      description
        "meter action type";
    }
    choice exceed-3color-meter-action-val {
      description
        " meter action based on choice of meter action type";
    }
  }
}

grouping violate-3color-meter-action-params {
  description
    "meter action parameters";
  list violate-3color-meter-action-params {
    key "violate-3color-meter-action-type";
    ordered-by user;
    description
      "Configuration of basic-meter & associated actions";
    leaf violate-3color-meter-action-type {
      type identityref {
        base violate-3color-meter-action-type;
      }
      description
        "meter action type";
    }
    choice violate-3color-meter-action-val {
      description
        " meter action based on choice of meter action type";
    }
  }
}

grouping one-rate-two-color-meter {
  container one-rate-two-color-meter {
    description
      "single rate two color marker meter";
    leaf committed-rate-value {
      type uint64;
      description
        "committed rate value";
    }
  }
}
```

```
    leaf committed-rate-unit {
      type identityref {
        base rate-unit-type;
      }
      description
        "committed rate unit";
    }
    leaf committed-burst-value {
      type uint64;
      description
        "burst value";
    }
    leaf committed-burst-unit {
      type identityref {
        base burst-unit-type;
      }
      description
        "committed burst unit";
    }
    container conform-action {
      uses conform-2color-meter-action-params;
      description
        "conform action";
    }
    container exceed-action {
      uses exceed-2color-meter-action-params;
      description
        "exceed action";
    }
  }
  description
    "single rate two color marker meter attributes";
}

grouping one-rate-tri-color-meter {
  container one-rate-tri-color-meter {
    description
      "single rate three color meter";
    reference
      "RFC2697: A Single Rate Three Color Marker";
  }
  leaf committed-rate-value {
    type uint64;
    description
      "meter rate";
  }
  leaf committed-rate-unit {
    type identityref {
      base rate-unit-type;
    }
  }
}
```

```
    }
    description
      "committed rate unit";
  }
  leaf committed-burst-value {
    type uint64;
    description
      "committed burst size";
  }
  leaf committed-burst-unit {
    type identityref {
      base burst-unit-type;
    }
    description
      "committed burst unit";
  }
  leaf excess-burst-value {
    type uint64;
    description
      "excess burst size";
  }
  leaf excess-burst-unit {
    type identityref {
      base burst-unit-type;
    }
    description
      "excess burst unit";
  }
  container conform-action {
    uses conform-3color-meter-action-params;
    description
      "conform, or green action";
  }
  container exceed-action {
    uses exceed-3color-meter-action-params;
    description
      "exceed, or yellow action";
  }
  container violate-action {
    uses violate-3color-meter-action-params;
    description
      "violate, or red action";
  }
}
description
  "one-rate-tri-color-meter attributes";
}
```

```
grouping two-rate-tri-color-meter {
  container two-rate-tri-color-meter {
    description
      "two rate three color meter";
    reference
      "RFC2698: A Two Rate Three Color Marker";
    leaf committed-rate-value {
      type uint64;
      units "bits-per-second";
      description
        "committed rate";
    }
    leaf committed-rate-unit {
      type identityref {
        base rate-unit-type;
      }
      description
        "committed rate unit";
    }
    leaf committed-burst-value {
      type uint64;
      description
        "committed burst size";
    }
    leaf committed-burst-unit {
      type identityref {
        base burst-unit-type;
      }
      description
        "committed burst unit";
    }
    leaf peak-rate-value {
      type uint64;
      description
        "peak rate";
    }
    leaf peak-rate-unit {
      type identityref {
        base rate-unit-type;
      }
      description
        "committed rate unit";
    }
    leaf peak-burst-value {
      type uint64;
      description
        "committed burst size";
    }
  }
}
```

```
leaf peak-burst-unit {
  type identityref {
    base burst-unit-type;
  }
  description
    "peak burst unit";
}
container conform-action {
  uses conform-3color-meter-action-params;
  description
    "conform, or green action";
}
container exceed-action {
  uses exceed-3color-meter-action-params;
  description
    "exceed, or yellow action";
}
container violate-action {
  uses violate-3color-meter-action-params;
  description
    "exceed, or red action";
}
}
description
  "two-rate-tri-color-meter attributes";
}

grouping meter {
  choice meter-type {
    case one-rate-two-color-meter-type {
      uses one-rate-two-color-meter;
      description
        "basic meter";
    }
    case one-rate-tri-color-meter-type {
      uses one-rate-tri-color-meter;
      description
        "one rate tri-color meter";
    }
    case two-rate-tri-color-meter-type {
      uses two-rate-tri-color-meter;
      description
        "two rate tri-color meter";
    }
  }
  description
    " meter action based on choice of meter action type";
}
description
```

```
        "meter attributes";
    }

    container meter-template {
        description
            "list of meter templates";
        list meter-entry {
            if-feature meter-template-support;
            key "meter-name";
            description
                "meter entry template";
            leaf meter-name {
                type string;
                description
                    "meter identifier";
            }
            uses meter;
        }
    }

    grouping meter-reference {
        container meter-reference-cfg {
            leaf meter-reference-name {
                type string ;
                mandatory true;
                description
                    "This leaf defines name of the meter referenced";
            }
            leaf meter-type {
                type identityref {
                    base meter-type;
                }
                mandatory true;
                description
                    "This leaf defines type of the meter";
            }
            description
                "meter reference name";
        }
        description
            "meter reference";
    }

    grouping count {
        container count-cfg {
            if-feature count-feature;
            leaf count-action {
                type empty;
            }
        }
    }
```

```
        description
            "count action";
    }
    description
        "the count action";
}
description
    "the count action grouping";
}

grouping named-counter {
    container named-counter-cfg {
        if-feature named-counter-feature;
        leaf count-name-action {
            type string;
            description
                "count action";
        }
        description
            "the count action";
    }
    description
        "the count action grouping";
}

grouping discard {
    container discard-cfg {
        leaf discard {
            type empty;
            description
                "discard action";
        }
        description
            "discard action";
    }
    description
        "discard grouping";
}

grouping priority {
    container priority-cfg {
        leaf priority-level {
            type uint8;
            description
                "priority level";
        }
        description
            "priority attributes";
    }
}
```

```
    }
    description
      "priority attributes grouping";
  }
  grouping min-rate {
    container min-rate-cfg {
      uses rate-value-unit;
      description
        "min guaranteed bandwidth";
      reference
        "RFC3289, section 3.5.3";
    }
    description
      "minimum rate grouping";
  }
  grouping dscp-marking {
    container dscp-cfg {
      leaf dscp {
        type inet:dscp;
        description
          "dscp marking";
      }
      description
        "dscp marking container";
    }
    description
      "dscp marking grouping";
  }
  grouping traffic-group-marking {
    container traffic-group-cfg {
      leaf traffic-group {
        type string;
        description
          "traffic group marking";
      }
      description
        "traffic group marking container";
    }
    description
      "traffic group marking grouping";
  }
  grouping child-policy {
    container child-policy-cfg {
      if-feature child-policy-feature;
      leaf policy-name {
        type string;
        description
          "Hierarchical Policy";
      }
    }
  }
```



```
    }
    description
      "Hierarchical Policy configuration container";
  }
  description
    "Grouping of Hierarchical Policy configuration";
}
grouping max-rate {
  container max-rate-cfg {
    uses rate-value-unit;
    uses burst;
    description
      "maximum rate attributes container";
    reference
      "RFC3289, section 3.5.4";
  }
  description
    "maximum rate attributes";
}
grouping queue {
  container queue-cfg {
    uses priority;
    uses min-rate;
    uses max-rate;
    container algorithmic-drop-cfg {
      choice drop-algorithm {
        case tail-drop {
          container tail-drop-cfg {
            leaf tail-drop-alg {
              type empty;
              description
                "tail drop algorithm";
            }
            description
              "Tail Drop configuration container";
          }
          description
            "Tail Drop choice";
        }
        description
          "Choice of Drop Algorithm";
      }
      description
        "Algorithmic Drop configuration container";
    }
  }
  description
    "Queue configuration container";
}
```

```
        description
            "Queue grouping";
    }
    grouping scheduler {
        container scheduler-cfg {
            uses min-rate;
            uses max-rate;
            description
                "Scheduler configuration container";
        }
        description
            "Scheduler configuration grouping";
    }
}
<CODE ENDS>
```

#### 6.4. IETF-QOS-TARGET

```
<CODE BEGINS>file "ietf-qos-target@2019-03-13.yang"
module iETF-qos-target {
    yang-version 1.1;
    namespace "urn:ietf:params:xml:ns:yang:ietf-qos-target";
    prefix target;

    import iETF-interfaces {
        prefix if;
        reference "RFC8343: A YANG Data Model for Interface Management";
    }
    import iETF-qos-policy {
        prefix policy;
        reference "RFC XXXX: YANG Model for QoS";
    }

    organization "IETF RTG (Routing Area) Working Group";
    contact
        "WG Web:    <http://tools.ietf.org/wg/rtgwg/>
        WG List:    <mailto:rtgwg@ietf.org>
        WG Chair:   Chris Bowers
                   <mailto:cbowers@juniper.net>
        WG Chair:   Jeff Tantsura
                   <mailto:jefftant.ietf@gmail.com>
        Editor:     Aseem Choudhary
                   <mailto:asechoud@cisco.com>
        Editor:     Mahesh Jethanandani
                   <mailto:mjethanandani@gmail.com>
        Editor:     Norm Strahle
                   <mailto:nstrahle@juniper.net>";
    description
```

"This module contains a collection of YANG definitions for configuring qos specification implementations.  
Copyright (c) 2019 IETF Trust and the persons identified as authors of the code. All rights reserved.  
Redistribution and use in source and binary forms, with or without modification, is permitted pursuant to, and subject to the license terms contained in, the Simplified BSD License set forth in Section 4.c of the IETF Trust's Legal Provisions Relating to IETF Documents  
(<http://trustee.ietf.org/license-info>).  
This version of this YANG module is part of RFC XXXX; see the RFC itself for full legal notices.";

```
revision 2019-03-13 {
  description
    "Latest revision qos based policy applied to a target";
  reference "RFC XXXX: YANG Model for QoS";
}

identity direction {
  description
    "This is identity of traffic direction";
}

identity inbound {
  base direction;
  description
    "Direction of traffic coming into the network entry";
}

identity outbound {
  base direction;
  description
    "Direction of traffic going out of the network entry";
}

augment "/if:interfaces/if:interface" {
  description
    "Augments Diffserv Target Entry to Interface module";
  list qos-target-entry {
    key "direction policy-type";
    description
      "policy target for inbound or outbound direction";
    leaf direction {
      type identityref {
        base direction;
      }
      description

```

```

        "Direction fo the traffic flow either inbound or outbound";
    }
    leaf policy-type {
        type identityref {
            base policy:policy-type;
        }
        description
            "Policy entry type";
    }
    leaf policy-name {
        type string;
        mandatory true;
        description
            "Policy entry name";
    }
}
}
}
<CODE ENDS>

```

#### 6.5. IETF-DIFFSERV

```

<CODE BEGINS>file "ietf-diffserv@2019-03-13.yang"
module ietf-diffserv {
    yang-version 1.1;
    namespace "urn:ietf:params:xml:ns:yang:ietf-diffserv";
    prefix diffserv;

    import ietf-qos-classifier {
        prefix classifier;
        reference "RFC XXXX: YANG Model for QoS";
    }
    import ietf-qos-policy {
        prefix policy;
        reference "RFC XXXX: YANG Model for QoS";
    }
    import ietf-qos-action {
        prefix action;
        reference "RFC XXXX: YANG Model for QoS";
    }
    import ietf-inet-types {
        prefix inet;
        reference "RFC 6991: Common YANG Data Types";
    }

    organization "IETF RTG (Routing Area) Working Group";
    contact
        "WG Web:  <http://tools.ietf.org/wg/rtgwg/>

```

```
WG List: <mailto:rtgwg@ietf.org>
WG Chair: Chris Bowers
         <mailto:cbowers@juniper.net>
WG Chair: Jeff Tantsura
         <mailto:jefftant.ietf@gmail.com>
Editor:   Aseem Choudhary
         <mailto:asechoud@cisco.com>
Editor:   Mahesh Jethanandani
         <mailto:mjethanandani@gmail.com>
Editor:   Norm Strahle
         <mailto:nstrahle@juniper.net>";
description
  "This module contains a collection of YANG definitions for
  configuring diffserv specification implementations.
  Copyright (c) 2019 IETF Trust and the persons identified as
  authors of the code. All rights reserved.
  Redistribution and use in source and binary forms, with or
  without modification, is permitted pursuant to, and subject
  to the license terms contained in, the Simplified BSD License
  set forth in Section 4.c of the IETF Trust's Legal Provisions
  Relating to IETF Documents
  (http://trustee.ietf.org/license-info).
  This version of this YANG module is part of RFC XXXX; see
  the RFC itself for full legal notices.";

revision 2019-03-13 {
  description
    "Latest revision of diffserv based classifier";
  reference "RFC XXXX: YANG Model for QoS";
}

feature diffserv-queue-inline-support {
  description
    "Queue inline support in diffserv policy";
}
feature diffserv-scheduler-inline-support {
  description
    "scheduler inline support in diffserv policy";
}
identity diffserv-policy-type {
  base policy:policy-type;
  description
    "This defines ip policy-type";
}
identity ipv4-diffserv-policy-type {
  base policy:policy-type;
  description
    "This defines ipv4 policy-type";
}
```

```
}
identity ipv6-diffserv-policy-type {
  base policy:policy-type;
  description
    "This defines ipv6 policy-type";
}

identity dscp {
  base classifier:filter-type;
  description
    "Differentiated services code point filter-type";
}
identity source-ipv4-address {
  base classifier:filter-type;
  description
    "source ipv4 address filter-type";
}
identity destination-ipv4-address {
  base classifier:filter-type;
  description
    "destination ipv4 address filter-type";
}
identity source-ipv6-address {
  base classifier:filter-type;
  description
    "source ipv6 address filter-type";
}
identity destination-ipv6-address {
  base classifier:filter-type;
  description
    "destination ipv6 address filter-type";
}
identity source-port {
  base classifier:filter-type;
  description
    "source port filter-type";
}
identity destination-port {
  base classifier:filter-type;
  description
    "destination port filter-type";
}
identity protocol {
  base classifier:filter-type;
  description
    "protocol type filter-type";
}
identity traffic-group-name {
```

```
    base classifier:filter-type;
    description
      "traffic-group filter type";
  }

  identity meter-type {
    description
      "This base identity type defines meter types";
  }
  identity one-rate-two-color-meter-type {
    base meter-type;
    description
      "one rate two color meter type";
  }
  identity one-rate-tri-color-meter-type {
    base meter-type;
    description
      "one rate three color meter type";
  }
  identity two-rate-tri-color-meter-type {
    base meter-type;
    description
      "two rate three color meter action type";
  }
  grouping dscp-cfg {
    list dscp-cfg {
      key "dscp-min dscp-max";
      description
        "list of dscp ranges";
      leaf dscp-min {
        type inet:dscp;
        description
          "Minimum value of dscp min-max range";
      }
      leaf dscp-max {
        type inet:dscp;
        description
          "maximum value of dscp min-max range";
      }
    }
    description
      "Filter grouping containing list of dscp ranges";
  }
  grouping source-ipv4-address-cfg {
    list source-ipv4-address-cfg {
      key "source-ipv4-addr";
      description
        "list of source ipv4 address";
    }
  }
```

```
        leaf source-ipv4-addr {
            type inet:ipv4-prefix;
            description
                "source ipv4 prefix";
        }
    }
    description
        "Filter grouping containing list of source ipv4 addresses";
}
grouping destination-ipv4-address-cfg {
    list destination-ipv4-address-cfg {
        key "destination-ipv4-addr";
        description
            "list of destination ipv4 address";
        leaf destination-ipv4-addr {
            type inet:ipv4-prefix;
            description
                "destination ipv4 prefix";
        }
    }
}
description
    "Filter grouping containing list of destination ipv4 address";
}
grouping source-ipv6-address-cfg {
    list source-ipv6-address-cfg {
        key "source-ipv6-addr";
        description
            "list of source ipv6 address";
        leaf source-ipv6-addr {
            type inet:ipv6-prefix;
            description
                "source ipv6 prefix";
        }
    }
}
description
    "Filter grouping containing list of source ipv6 addresses";
}
grouping destination-ipv6-address-cfg {
    list destination-ipv6-address-cfg {
        key "destination-ipv6-addr";
        description
            "list of destination ipv4 or ipv6 address";
        leaf destination-ipv6-addr {
            type inet:ipv6-prefix;
            description
                "destination ipv6 prefix";
        }
    }
}
}
```



```
    description
      "Filter grouping containing list of destination ipv6 address";
  }
  grouping source-port-cfg {
    list source-port-cfg {
      key "source-port-min source-port-max";
      description
        "list of ranges of source port";
      leaf source-port-min {
        type inet:port-number;
        description
          "minimum value of source port range";
      }
      leaf source-port-max {
        type inet:port-number;
        description
          "maximum value of source port range";
      }
    }
  }
  description
    "Filter grouping containing list of source port ranges";
}
grouping destination-port-cfg {
  list destination-port-cfg {
    key "destination-port-min destination-port-max";
    description
      "list of ranges of destination port";
    leaf destination-port-min {
      type inet:port-number;
      description
        "minimum value of destination port range";
    }
    leaf destination-port-max {
      type inet:port-number;
      description
        "maximum value of destination port range";
    }
  }
}
description
  "Filter grouping containing list of destination port ranges";
}
grouping protocol-cfg {
  list protocol-cfg {
    key "protocol-min protocol-max";
    description
      "list of ranges of protocol values";
    leaf protocol-min {
      type uint8 {
```

```
        range "0..255";
    }
    description
        "minimum value of protocol range";
}
leaf protocol-max {
    type uint8 {
        range "0..255";
    }
    description
        "maximum value of protocol range";
}
}
description
    "Filter grouping containing list of Protocol ranges";
}
grouping traffic-group-cfg {
    container traffic-group-cfg {
        leaf traffic-group-name {
            type string ;
            description
                "This leaf defines name of the traffic group referenced";
        }
    }
    description
        "traffic group container";
}
description
    "traffic group grouping";
}

augment "/classifier:classifier/classifier:classifier-entry" +
    "/classifier:filter-entry" {
    choice filter-param {
        description
            "Choice of filter types";
        case dscp {
            uses dscp-cfg;
            description
                "Filter containing list of dscp ranges";
        }
        case source-ipv4-address {
            uses source-ipv4-address-cfg;
            description
                "Filter containing list of source ipv4 addresses";
        }
        case destination-ipv4-address {
            uses destination-ipv4-address-cfg;
            description
                "Filter containing list of destination ipv4 addresses";
        }
    }
}
```

```

        "Filter containing list of destination ipv4 address";
    }
    case source-ipv6-address {
        uses source-ipv6-address-cfg;
        description
            "Filter containing list of source ipv6 addresses";
    }
    case destination-ipv6-address {
        uses destination-ipv6-address-cfg;
        description
            "Filter containing list of destination ipv6 address";
    }
    case source-port {
        uses source-port-cfg;
        description
            "Filter containing list of source-port ranges";
    }
    case destination-port {
        uses destination-port-cfg;
        description
            "Filter containing list of destination-port ranges";
    }
    case protocol {
        uses protocol-cfg;
        description
            "Filter Type Protocol";
    }
    case traffic-group {
        uses traffic-group-cfg;
        description
            "Filter Type traffic-group";
    }
}
description
    "augments diffserv filters to qos classifier";
}
augment "/policy:policies/policy:policy-entry" +
    "/policy:classifier-entry/policy:filter-entry" {
    when "../..../policy:policy-type =
        'diffserv:ipv4-diffserv-policy-type' or
        ../..../policy:policy-type =
        'diffserv:ipv6-diffserv-policy-type' or
        ../..../policy:policy-type =
        'diffserv:diffserv-policy-type'" {
        description
            "Filters can be augmented if policy type is
            ipv4, ipv6 or default diffserv policy types ";
    }
}

```

```
description
  "Augments Diffserv Classifier with common filter types";
choice filter-params {
  description
    "Choice of action types";
  case dscp {
    uses dscp-cfg;
    description
      "Filter containing list of dscp ranges";
  }
  case source-ipv4-address {
    when "../policy:policy-type !=
          'diffserv:ipv6-diffserv-policy-type'" {
      description
        "If policy type is v6, this filter cannot be used.";
    }
    uses source-ipv4-address-cfg;
    description
      "Filter containing list of source ipv4 addresses";
  }
  case destination-ipv4-address {
    when "../policy:policy-type !=
          'diffserv:ipv6-diffserv-policy-type'" {
      description
        "If policy type is v6, this filter cannot be used.";
    }
    uses destination-ipv4-address-cfg;
    description
      "Filter containing list of destination ipv4 address";
  }
  case source-ipv6-address {
    when "../policy:policy-type !=
          'diffserv:ipv4-diffserv-policy-type'" {
      description
        "If policy type is v4, this filter cannot be used.";
    }
    uses source-ipv6-address-cfg;
    description
      "Filter containing list of source ipv6 addresses";
  }
  case destination-ipv6-address {
    when "../policy:policy-type !=
          'diffserv:ipv4-diffserv-policy-type'" {
      description
        "If policy type is v4, this filter cannot be used.";
    }
    uses destination-ipv6-address-cfg;
    description
```

```

        "Filter containing list of destination ipv6 address";
    }
    case source-port {
        uses source-port-cfg;
        description
            "Filter containing list of source-port ranges";
    }
    case destination-port {
        uses destination-port-cfg;
        description
            "Filter containing list of destination-port ranges";
    }
    case protocol {
        uses protocol-cfg;
        description
            "Filter Type Protocol";
    }
    case traffic-group {
        uses traffic-group-cfg;
        description
            "Filter Type traffic-group";
    }
}
}
augment "/policy:policies/policy:policy-entry" +
    "/policy:classifier-entry" +
    "/policy:classifier-action-entry-cfg" +
    "/policy:action-cfg-params" {
    when "../..../policy:policy-type =
        'diffserv:ipv4-diffserv-policy-type' or
        ../..../policy:policy-type =
        'diffserv:ipv6-diffserv-policy-type' or
        ../..../policy:policy-type =
        'diffserv:diffserv-policy-type' " {
        description
            "Actions can be augmented if policy type is ipv4,
            ipv6 or default diffserv policy types ";
    }
    description
        "Augments Diffserv Policy with action configuration";
    case dscp-marking {
        uses action:dscp-marking;
    }
    case meter-inline {
        if-feature action:meter-inline-feature;
        uses action:meter;
    }
    case meter-reference {

```

```
        if-feature action:meter-reference-feature;
        uses action:meter-reference;
    }
    case child-policy {
        if-feature action:child-policy-feature;
        uses action:child-policy;
    }
    case count {
        if-feature action:count-feature;
        uses action:count;
    }
    case named-count {
        if-feature action:named-counter-feature;
        uses action:named-counter;
    }
    case queue-inline {
        if-feature diffserv-queue-inline-support;
        uses action:queue;
    }
    case scheduler-inline {
        if-feature diffserv-scheduler-inline-support;
        uses action:scheduler;
    }
}
}
<CODE ENDS>
```

#### 6.6. IETF-QUEUE-POLICY

```
<CODE BEGINS>file "ietf-queue-policy@2019-03-13.yang"
module iETF-queue-policy {
    yang-version 1.1;
    namespace "urn:ietf:params:xml:ns:yang:ietf-queue-policy";
    prefix queue-policy;

    import iETF-qos-policy {
        prefix policy;
        reference "RFC XXXX: YANG Model for QoS";
    }
    import iETF-qos-action {
        prefix action;
        reference "RFC XXXX: YANG Model for QoS";
    }
    import iETF-diffserv {
        prefix diffserv;
        reference "RFC XXXX: YANG Model for QoS";
    }
}
```

```
organization "IETF RTG (Routing Area) Working Group";
contact
  "WG Web:    <http://tools.ietf.org/wg/rtgwg/>
  WG List:    <mailto:rtgwg@ietf.org>
  WG Chair:   Chris Bowers
              <mailto:cbowers@juniper.net>
  WG Chair:   Jeff Tantsura
              <mailto:jefftant.ietf@gmail.com>
  Editor:     Aseem Choudhary
              <mailto:asechoud@cisco.com>
  Editor:     Mahesh Jethanandani
              <mailto:mjethanandani@gmail.com>
  Editor:     Norm Strahle
              <mailto:nstrahle@juniper.net>";
description
  "This module contains a collection of YANG definitions for
  configuring diffserv specification implementations.
  Copyright (c) 2019 IETF Trust and the persons identified as
  authors of the code. All rights reserved.
  Redistribution and use in source and binary forms, with or
  without modification, is permitted pursuant to, and subject
  to the license terms contained in, the Simplified BSD License
  set forth in Section 4.c of the IETF Trust's Legal Provisions
  Relating to IETF Documents
  (http://trustee.ietf.org/license-info).
  This version of this YANG module is part of RFC XXXX; see
  the RFC itself for full legal notices.";

revision 2019-03-13 {
  description
    "Latest revision of queuing policy module";
  reference "RFC XXXX: YANG Model for QoS";
}

feature queue-policy-support {
  description
    " This feature allows queue policy configuration
    as a separate policy type support.";
}

feature queue-inline-support {
  description
    "Queue inline support in Queue policy";
}

feature queue-template-support {
  description
    "Queue template support in Queue policy";
```

```
}

identity queue-policy-type {
  base policy:policy-type;
  description
    "This defines queue policy-type";
}

augment "/policy:policies/policy:policy-entry" +
  "/policy:classifier-entry/policy:filter-entry" {
  when "../..../policy:policy-type =
    'queue-policy:queue-policy-type'" {
    description
      "If policy type is v6, this filter cannot be used.";
  }
  if-feature queue-policy-support;
  choice filter-params {
    description
      "Choice of action types";
    case traffic-group-name {
      uses diffserv:traffic-group-cfg;
      description
        "traffic group name";
    }
  }
  description
    "Augments Queue policy Classifier with common filter types";
}

identity queue-template-name {
  base policy:action-type;
  description
    "queue template name";
}

grouping queue-template-reference {
  container queue-template-reference-cfg {
    leaf queue-template-name {
      type string ;
      mandatory true;
      description
        "This leaf defines name of the queue template referenced";
    }
  }
  description
    "queue template reference";
}
description
  "queue template reference grouping";
```



```

    }

    container queue-template {
        if-feature queue-policy-support;
        description
            "Queue template";
        leaf name {
            type string;
            description
                "A unique name identifying this queue template";
        }
        uses action:queue;
    }

    augment "/policy:policies/policy:policy-entry" +
        "/policy:classifier-entry" +
        "/policy:classifier-action-entry-cfg" +
        "/policy:action-cfg-params" {
        when "../.. /policy:policy-type =
            'queue-policy:queue-policy-type'" {
            description
                "queue policy actions.";
        }
        if-feature queue-policy-support;
        case queue-template-name {
            if-feature queue-template-support;
            uses queue-template-reference;
        }
        case queue-inline {
            if-feature queue-inline-support;
            uses action:queue;
        }
        description
            "augments queue template reference to queue policy";
    }
}
<CODE ENDS>

```

## 6.7. IETF-SCHEDULER-POLICY

```

<CODE BEGINS>file "ietf-scheduler-policy@2019-03-13.yang"
module ietf-scheduler-policy {
    yang-version 1.1;
    namespace "urn:ietf:params:xml:ns:yang:ietf-scheduler-policy";
    prefix scheduler-policy;

    import ietf-qos-classifier {

```

```
    prefix classifier;
    reference "RFC XXXX: YANG Model for QoS";
}
import ietf-qos-policy {
    prefix policy;
    reference "RFC XXXX: YANG Model for QoS";
}
import ietf-qos-action {
    prefix action;
    reference "RFC XXXX: YANG Model for QoS";
}

organization "IETF RTG (Routing Area) Working Group";
contact
  "WG Web:    <http://tools.ietf.org/wg/rtgwg/>
  WG List:    <mailto:rtgwg@ietf.org>
  WG Chair:   Chris Bowers
              <mailto:cbowers@juniper.net>
  WG Chair:   Jeff Tantsura
              <mailto:jefftant.ietf@gmail.com>
  Editor:     Norm Strahle
              <mailto:nstrahle@juniper.net>
  Editor:     Aseem Choudhary
              <mailto:asechoud@cisco.com>";
description
  "This module contains a collection of YANG definitions for
  configuring diffserv specification implementations.
  Copyright (c) 2019 IETF Trust and the persons identified as
  authors of the code. All rights reserved.
  Redistribution and use in source and binary forms, with or
  without modification, is permitted pursuant to, and subject
  to the license terms contained in, the Simplified BSD License
  set forth in Section 4.c of the IETF Trust's Legal Provisions
  Relating to IETF Documents
  (http://trustee.ietf.org/license-info).
  This version of this YANG module is part of RFC XXXX; see
  the RFC itself for full legal notices.";

revision 2019-03-13 {
    description
      "Latest revision of scheduler policy module";
    reference "RFC XXXX: YANG Model for QoS";
}
feature scheduler-policy-support {
    description
      " This feature allows sheduler policy configuration
      as a separate policy type support.";
}
```

```
identity scheduler-policy-type {
  base policy:policy-type;
  description
    "This defines scheduler policy-type";
}

identity filter-match-all {
  base classifier:filter-type;
  description
    "Traffic-group filter type";
}

grouping filter-match-all-cfg {
  container match-all-cfg {
    leaf match-all-action {
      type empty;
      description
        "match all packets";
    }
    description
      "the match-all action";
  }
  description
    "the match-all filter grouping";
}

augment "/policy:policies/policy:policy-entry" +
  "/policy:classifier-entry/policy:filter-entry" {
  when "../policy:policy-type =
    'scheduler-policy:scheduler-policy-type'" {
    description
      "Only when policy type is scheduler-policy";
  }
  choice filter-params {
    description
      "Choice of action types";
    case filter-match-all {
      uses filter-match-all-cfg;
      description
        "filter match-all";
    }
  }
  description
    "Augments Queue policy Classifier with common filter types";
}

identity queue-policy-name {
  base policy:action-type;
```

```
    description
      "queue policy name";
  }

  grouping queue-policy-name-cfg {
    container queue-policy-name {
      leaf queue-policy {
        type string ;
        mandatory true;
        description
          "This leaf defines name of the queue-policy";
      }
    }
    description
      "container for queue-policy name";
  }
  description
    "queue-policy name grouping";
}

augment "/policy:policies/policy:policy-entry" +
  "/policy:classifier-entry" +
  "/policy:classifier-action-entry-cfg" +
  "/policy:action-cfg-params" {
  when "../..../policy:policy-type =
    'scheduler-policy:scheduler-policy-type'" {
    description
      "Only when policy type is scheduler-policy";
  }
  case scheduler {
    uses action:scheduler;
  }
  case queue-policy-name {
    uses queue-policy-name-cfg;
  }
  description
    "augments scheduler template reference to scheduler policy";
}
}
<CODE ENDS>
```

## 7. IANA Considerations

TBD

## 8. Security Considerations

## 9. Acknowledgement

The authors wish to thank Ruediger Geib, Fred Baker, Greg Misky, Tom Petch, many others for their helpful comments.

## 10. References

### 10.1. 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>>.
- [RFC2697] Heinanen, J. and R. Guerin, "A Single Rate Three Color Marker", RFC 2697, DOI 10.17487/RFC2697, September 1999, <<https://www.rfc-editor.org/info/rfc2697>>.
- [RFC2698] Heinanen, J. and R. Guerin, "A Two Rate Three Color Marker", RFC 2698, DOI 10.17487/RFC2698, September 1999, <<https://www.rfc-editor.org/info/rfc2698>>.
- [RFC3289] Baker, F., Chan, K., and A. Smith, "Management Information Base for the Differentiated Services Architecture", RFC 3289, DOI 10.17487/RFC3289, May 2002, <<https://www.rfc-editor.org/info/rfc3289>>.
- [RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <<https://www.rfc-editor.org/info/rfc6020>>.
- [RFC6991] Schoenwaelder, J., Ed., "Common YANG Data Types", RFC 6991, DOI 10.17487/RFC6991, July 2013, <<https://www.rfc-editor.org/info/rfc6991>>.
- [RFC7950] Bjorklund, M., Ed., "The YANG 1.1 Data Modeling Language", RFC 7950, DOI 10.17487/RFC7950, August 2016, <<https://www.rfc-editor.org/info/rfc7950>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.

- [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>>.
- [RFC8343] Bjorklund, M., "A YANG Data Model for Interface Management", RFC 8343, DOI 10.17487/RFC8343, March 2018, <<https://www.rfc-editor.org/info/rfc8343>>.

## 10.2. Informative References

- [RFC2475] Blake, S., Black, D., Carlson, M., Davies, E., Wang, Z., and W. Weiss, "An Architecture for Differentiated Services", RFC 2475, DOI 10.17487/RFC2475, December 1998, <<https://www.rfc-editor.org/info/rfc2475>>.
- [RFC8340] Bjorklund, M. and L. Berger, Ed., "YANG Tree Diagrams", BCP 215, RFC 8340, DOI 10.17487/RFC8340, March 2018, <<https://www.rfc-editor.org/info/rfc8340>>.

## Appendix A. Company A, Company B and Company C examples

Company A, Company B and Company C Diffserv modules augments all the filter types of the QoS classifier module as well as the QoS policy module that allow it to define marking, metering, min-rate, max-rate actions. Queuing and metering counters are realized by augmenting of the QoS target module.

### A.1. Example of Company A Diffserv Model

The following Company A vendor example augments the qos and diffserv model, demonstrating some of the following functionality:

- use of template based classifier definitions
- use of single policy type modelling queue, scheduler policy, and a filter policy. All of these policies either augment the qos policy or the diffserv modules
- use of inline actions in a policy
- flexibility in marking dscp or metadata at ingress and/or egress.

```
module example-compa-diffserv {  
  yang-version 1.1;  
  namespace "urn:ietf:params:xml:ns:yang:example-compa-diffserv";  
  prefix example;
```

```
import ietf-qos-classifier {
  prefix classifier;
  reference "RFC XXXX: YANG Model for QoS";
}
import ietf-qos-policy {
  prefix policy;
  reference "RFC XXXX: YANG Model for QoS";
}
import ietf-qos-action {
  prefix action;
  reference "RFC XXXX: YANG Model for QoS";
}
import ietf-diffserv {
  prefix diffserv;
  reference "RFC XXXX: YANG Model for QoS";
}

organization "Company A";
contact
  "Editor:   XYZ
   <mailto:xyz@compa.com>";
description
  "This module contains a collection of YANG definitions of
  companyA diffserv specification extension.";
  Copyright (c) 2019 IETF Trust and the persons identified as
  authors of the code. All rights reserved.
  Redistribution and use in source and binary forms, with or
  without modification, is permitted pursuant to, and subject
  to the license terms contained in, the Simplified BSD License
  set forth in Section 4.c of the IETF Trust's Legal Provisions
  Relating to IETF Documents
  (http://trustee.ietf.org/license-info).

  This version of this YANG module is part of RFC XXXX; see
  the RFC itself for full legal notices.";

revision 2019-03-13 {
  description
    "Initial revision for diffserv actions on network packets";
  reference
    "RFC 6020: YANG - A Data Modeling Language for the
     Network Configuration Protocol (NETCONF)";
}

identity default-policy-type {
  base policy:policy-type;
  description
    "This defines default policy-type";
```

```
    }

    identity qos-group {
      base classifier:filter-type;
      description
        "qos-group filter-type";
    }

    grouping qos-group-cfg {
      list qos-group-cfg {
        key "qos-group-min qos-group-max";
        description
          "list of dscp ranges";
        leaf qos-group-min {
          type uint8;
          description
            "Minimum value of qos-group range";
        }
        leaf qos-group-max {
          type uint8;
          description
            "maximum value of qos-group range";
        }
      }
      description
        "Filter containing list of qos-group ranges";
    }

    grouping wred-threshold {
      container wred-min-thresh {
        uses action:threshold;
        description
          "Minimum threshold";
      }
      container wred-max-thresh {
        uses action:threshold;
        description
          "Maximum threshold";
      }
      leaf mark-probability {
        type uint32 {
          range "1..1000";
        }
        description
          "Mark probability";
      }
      description
        "WRED threshold attributes";
    }
```



```
    }

    grouping randomdetect {
      leaf exp-weighting-const {
        type uint32;
        description
          "Exponential weighting constant factor for wred profile";
      }
      uses wred-threshold;
      description
        "Random detect attributes";
    }

    augment "/classifier:classifiers/" +
      "classifier:classifier-entry/" +
      "classifier:filter-entry/diffserv:filter-param" {
      case qos-group {
        uses qos-group-cfg;
        description
          "Filter containing list of qos-group ranges.
           Qos-group represent packet metadata information
           in a device. ";
      }
      description
        "augmentation of classifier filters";
    }

    augment "/policy:policies/policy:policy-entry/" +
      "policy:classifier-entry/" +
      "policy:classifier-action-entry-cfg/" +
      "policy:action-cfg-params" {
      case random-detect {
        uses randomdetect;
      }
      description
        "Augment the actions to policy entry";
    }

    augment "/policy:policies" +
      "/policy:policy-entry" +
      "/policy:classifier-entry" +
      "/policy:classifier-action-entry-cfg" +
      "/policy:action-cfg-params" +
      "/diffserv:meter-inline" +
      "/diffserv:meter-type" +
      "/diffserv:one-rate-two-color-meter-type" +
      "/diffserv:one-rate-two-color-meter" +
      "/diffserv:conform-action" +
      "/diffserv:conform-2color-meter-action-params" +
```

```
        "/diffserv:conform-2color-meter-action-val" {

    description
        "augment the one-rate-two-color meter conform
        with actions";
    case meter-action-drop {
        description
            "meter drop";
            uses action:drop;
        }
    case meter-action-mark-dscp {
        description
            "meter action dscp marking";
            uses action:dscp-marking;
        }
    }
    augment "/policy:policies" +
        "/policy:policy-entry" +
        "/policy:classifier-entry" +
        "/policy:classifier-action-entry-cfg" +
        "/policy:action-cfg-params" +
        "/diffserv:meter-inline" +
        "/diffserv:meter-type" +
        "/diffserv:one-rate-two-color-meter-type" +
        "/diffserv:one-rate-two-color-meter" +
        "/diffserv:exceed-action" +
        "/diffserv:exceed-2color-meter-action-params" +
        "/diffserv:exceed-2color-meter-action-val" {

    description
        "augment the one-rate-two-color meter exceed
        with actions";
    case meter-action-drop {
        description
            "meter drop";
            uses action:drop;
        }
    case meter-action-mark-dscp {
        description
            "meter action dscp marking";
            uses action:dscp-marking;
        }
    }
    augment "/policy:policies" +
        "/policy:policy-entry" +
        "/policy:classifier-entry" +
        "/policy:classifier-action-entry-cfg" +
        "/policy:action-cfg-params" +
```

```
        "/diffserv:meter-inline" +
        "/diffserv:meter-type" +
        "/diffserv:one-rate-tri-color-meter-type" +
        "/diffserv:one-rate-tri-color-meter" +
        "/diffserv:conform-action" +
        "/diffserv:conform-3color-meter-action-params" +
        "/diffserv:conform-3color-meter-action-val" {

description
    "augment the one-rate-tri-color meter conform
    with actions";
case meter-action-drop {
    description
        "meter drop";
        uses action:drop;
}
case meter-action-mark-dscp {
    description
        "meter action dscp marking";
        uses action:dscp-marking;
}
}
augment "/policy:policies" +
        "/policy:policy-entry" +
        "/policy:classifier-entry" +
        "/policy:classifier-action-entry-cfg" +
        "/policy:action-cfg-params" +
        "/diffserv:meter-inline" +
        "/diffserv:meter-type" +
        "/diffserv:one-rate-tri-color-meter-type" +
        "/diffserv:one-rate-tri-color-meter" +
        "/diffserv:exceed-action" +
        "/diffserv:exceed-3color-meter-action-params" +
        "/diffserv:exceed-3color-meter-action-val" {

description
    "augment the one-rate-tri-color meter exceed
    with actions";
case meter-action-drop {
    description
        "meter drop";
        uses action:drop;
}
case meter-action-mark-dscp {
    description
        "meter action dscp marking";
        uses action:dscp-marking;
}
}
```

```
}
augment "/policy:policies" +
  "/policy:policy-entry" +
  "/policy:classifier-entry" +
  "/policy:classifier-action-entry-cfg" +
  "/policy:action-cfg-params" +
  "/diffserv:meter-inline" +
  "/diffserv:meter-type" +
  "/diffserv:one-rate-tri-color-meter-type" +
  "/diffserv:one-rate-tri-color-meter" +
  "/diffserv:violate-action" +
  "/diffserv:violate-3color-meter-action-params" +
  "/diffserv:violate-3color-meter-action-val" {
  description
    "augment the one-rate-tri-color meter conform
    with actions";
  case meter-action-drop {
    description
      "meter drop";
    uses action:drop;
  }
  case meter-action-mark-dscp {
    description
      "meter action dscp marking";
    uses action:dscp-marking;
  }
}

augment "/policy:policies" +
  "/policy:policy-entry" +
  "/policy:classifier-entry" +
  "/policy:classifier-action-entry-cfg" +
  "/policy:action-cfg-params" +
  "/diffserv:meter-inline" +
  "/diffserv:meter-type" +
  "/diffserv:two-rate-tri-color-meter-type" +
  "/diffserv:two-rate-tri-color-meter" +
  "/diffserv:conform-action" +
  "/diffserv:conform-3color-meter-action-params" +
  "/diffserv:conform-3color-meter-action-val" {

  description
    "augment the one-rate-tri-color meter conform
    with actions";
  case meter-action-drop {
    description
      "meter drop";
    uses action:drop;
```

```
    }
    case meter-action-mark-dscp {
        description
            "meter action dscp marking";
            uses action:dscp-marking;
    }
}
augment "/policy:policies" +
    "/policy:policy-entry" +
    "/policy:classifier-entry" +
    "/policy:classifier-action-entry-cfg" +
    "/policy:action-cfg-params" +
    "/diffserv:meter-inline" +
    "/diffserv:meter-type" +
    "/diffserv:two-rate-tri-color-meter-type" +
    "/diffserv:two-rate-tri-color-meter" +
    "/diffserv:exceed-action" +
    "/diffserv:exceed-3color-meter-action-params" +
    "/diffserv:exceed-3color-meter-action-val" {

    description
        "augment the two-rate-tri-color meter exceed
        with actions";
    case meter-action-drop {
        description
            "meter drop";
            uses action:drop;
    }
    case meter-action-mark-dscp {
        description
            "meter action dscp marking";
            uses action:dscp-marking;
    }
}
augment "/policy:policies" +
    "/policy:policy-entry" +
    "/policy:classifier-entry" +
    "/policy:classifier-action-entry-cfg" +
    "/policy:action-cfg-params" +
    "/diffserv:meter-inline" +
    "/diffserv:meter-type" +
    "/diffserv:two-rate-tri-color-meter-type" +
    "/diffserv:two-rate-tri-color-meter" +
    "/diffserv:violate-action" +
    "/diffserv:violate-3color-meter-action-params" +
    "/diffserv:violate-3color-meter-action-val" {
    description
        "augment the two-rate-tri-color meter violate
```

```
        with actions";
    case meter-action-drop {
        description
            "meter drop";
        uses action:drop;
    }
    case meter-action-mark-dscp {
        description
            "meter action dscp marking";
        uses action:dscp-marking;
    }
}
augment "/policy:policies" +
    "/policy:policy-entry" +
    "/policy:classifier-entry" +
    "/policy:classifier-action-entry-cfg" +
    "/policy:action-cfg-params" +
    "/diffserv:meter-inline" +
    "/diffserv:meter-type" +
    "/diffserv:one-rate-two-color-meter-type" +
    "/diffserv:one-rate-two-color-meter" {
    description
        "augment the one-rate-two-color meter with" +
        "color classifiers";
    container conform-color {
        uses classifier:classifier-entry-generic-attr;
        description
            "conform color classifier container";
    }
    container exceed-color {
        uses classifier:classifier-entry-generic-attr;
        description
            "exceed color classifier container";
    }
}
augment "/policy:policies" +
    "/policy:policy-entry" +
    "/policy:classifier-entry" +
    "/policy:classifier-action-entry-cfg" +
    "/policy:action-cfg-params" +
    "/diffserv:meter-inline" +
    "/diffserv:meter-type" +
    "/diffserv:one-rate-tri-color-meter-type" +
    "/diffserv:one-rate-tri-color-meter" {
    description
        "augment the one-rate-tri-color meter with" +
        "color classifiers";
    container conform-color {
```

```
        uses classifier:classifier-entry-generic-attr;
        description
            "conform color classifier container";
    }
    container exceed-color {
        uses classifier:classifier-entry-generic-attr;
        description
            "exceed color classifier container";
    }
    container violate-color {
        uses classifier:classifier-entry-generic-attr;
        description
            "violate color classifier container";
    }
}
augment "/policy:policies" +
    "/policy:policy-entry" +
    "/policy:classifier-entry" +
    "/policy:classifier-action-entry-cfg" +
    "/policy:action-cfg-params" +
    "/diffserv:meter-inline" +
    "/diffserv:meter-type" +
    "/diffserv:two-rate-tri-color-meter-type" +
    "/diffserv:two-rate-tri-color-meter" {
description
    "augment the two-rate-tri-color meter with" +
    "color classifiers";
    container conform-color {
        uses classifier:classifier-entry-generic-attr;
        description
            "conform color classifier container";
    }
    container exceed-color {
        uses classifier:classifier-entry-generic-attr;
        description
            "exceed color classifier container";
    }
    container violate-color {
        uses classifier:classifier-entry-generic-attr;
        description
            "violate color classifier container";
    }
}
}
```

## A.2. Example of Company B Diffserv Model

The following vendor example augments the qos and diffserv model, demonstrating some of the following functionality:

- use of inline classifier definitions (defined inline in the policy vs referencing an externally defined classifier)
- use of multiple policy types, e.g. a queue policy, a scheduler policy, and a filter policy. All of these policies either augment the qos policy or the diffserv modules
- use of a queue module, which uses and extends the queue grouping from the ietf-qos-action module
- use of meter templates (v.s. meter inline)
- use of internal meta data for classification and marking

```
module example-compb-diffserv-filter-policy {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:" +
    "example-compb-diffserv-filter-policy";
  prefix compb-filter-policy;

  import ietf-qos-classifier {
    prefix classifier;
    reference "RFC XXXX: YANG Model for QoS";
  }
  import ietf-qos-policy {
    prefix policy;
    reference "RFC XXXX: YANG Model for QoS";
  }
  import ietf-qos-action {
    prefix action;
    reference "RFC XXXX: YANG Model for QoS";
  }
  import ietf-diffserv {
    prefix diffserv;
    reference "RFC XXXX: YANG Model for QoS";
  }

  organization "Company B";
  contact
    "Editor:   XYZ
     <mailto:xyz@compb.com>";

  description
```



"This module contains a collection of YANG definitions for configuring diffserv specification implementations. Copyright (c) 2019 IETF Trust and the persons identified as authors of the code. All rights reserved. Redistribution and use in source and binary forms, with or without modification, is permitted pursuant to, and subject to the license terms contained in, the Simplified BSD License set forth in Section 4.c of the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>).

This version of this YANG module is part of RFC XXXX; see the RFC itself for full legal notices."

```
revision 2019-03-13 {
  description
    "Latest revision of diffserv policy";
  reference "RFC XXXX";
}

/*****
 * Classification types
 *****/

identity forwarding-class {
  base classifier:filter-type;
  description
    "Forwarding class filter type";
}

identity internal-loss-priority {
  base classifier:filter-type;
  description
    "Internal loss priority filter type";
}

grouping forwarding-class-cfg {
  list forwarding-class-cfg {
    key "forwarding-class";
    description
      "list of forwarding-classes";
    leaf forwarding-class {
      type string;
      description
        "Forwarding class name";
    }
  }
}
```

```
    description
      "Filter containing list of forwarding classes";
  }

  grouping loss-priority-cfg {
    list loss-priority-cfg {
      key "loss-priority";
      description
        "list of loss-priorities";
      leaf loss-priority {
        type enumeration {
          enum high {
            description "High Loss Priority";
          }
          enum medium-high {
            description "Medium-high Loss Priority";
          }
          enum medium-low {
            description "Medium-low Loss Priority";
          }
          enum low {
            description "Low Loss Priority";
          }
        }
      }
      description
        "Loss-priority";
    }
  }
  description
    "Filter containing list of loss priorities";
}

augment "/policy:policies" +
  "/policy:policy-entry" +
  "/policy:classifier-entry" +
  "/policy:filter-entry" +
  "/diffserv:filter-params" {
  case forwarding-class {
    uses forwarding-class-cfg;
    description
      "Filter Type Internal-loss-priority";
  }
  case internal-loss-priority {
    uses loss-priority-cfg;
    description
      "Filter Type Internal-loss-priority";
  }
}
description
```

```
        "Augments Diffserv Classifier with vendor" +
        " specific types";
    }

/*****
 * Actions
 *****/

identity mark-fwd-class {
    base policy:action-type;
    description
        "mark forwarding class action type";
}

identity mark-loss-priority {
    base policy:action-type;
    description
        "mark loss-priority action type";
}

grouping mark-fwd-class {
    container mark-fwd-class-cfg {
        leaf forwarding-class {
            type string;
            description
                "Forwarding class name";
        }
        description
            "mark-fwd-class container";
    }
    description
        "mark-fwd-class grouping";
}

grouping mark-loss-priority {
    container mark-loss-priority-cfg {
        leaf loss-priority {
            type enumeration {
                enum high {
                    description "High Loss Priority";
                }
                enum medium-high {
                    description "Medium-high Loss Priority";
                }
                enum medium-low {
                    description "Medium-low Loss Priority";
                }
                enum low {
```

```
        description "Low Loss Priority";
      }
    }
    description
      "Loss-priority";
  }
  description
    "mark-loss-priority container";
}
description
  "mark-loss-priority grouping";
}

identity exceed-2color-meter-action-drop {
  base action:exceed-2color-meter-action-type;
  description
    "drop action type in a meter";
}

identity meter-action-mark-fwd-class {
  base action:exceed-2color-meter-action-type;
  description
    "mark forwarding class action type";
}

identity meter-action-mark-loss-priority {
  base action:exceed-2color-meter-action-type;
  description
    "mark loss-priority action type";
}

identity violate-3color-meter-action-drop {
  base action:violate-3color-meter-action-type;
  description
    "drop action type in a meter";
}

augment "/policy:policies/policy:policy-entry/" +
  "policy:classifier-entry/" +
  "policy:classifier-action-entry-cfg/" +
  "policy:action-cfg-params" {
  case mark-fwd-class {
    uses mark-fwd-class;
    description
      "Mark forwarding class in the packet";
  }
  case mark-loss-priority {
    uses mark-loss-priority;
  }
}
```

```
        description
            "Mark loss priority in the packet";
    }
    case discard {
        uses action:discard;
        description
            "Discard action";
    }
    description
        "Augments common diffserv policy actions";
}

augment "/action:meter-template" +
    "/action:meter-entry" +
    "/action:meter-type" +
    "/action:one-rate-tri-color-meter-type" +
    "/action:one-rate-tri-color-meter" {
    leaf one-rate-color-aware {
        type boolean;
        description
            "This defines if the meter is color-aware";
    }
}
augment "/action:meter-template" +
    "/action:meter-entry" +
    "/action:meter-type" +
    "/action:two-rate-tri-color-meter-type" +
    "/action:two-rate-tri-color-meter" {
    leaf two-rate-color-aware {
        type boolean;
        description
            "This defines if the meter is color-aware";
    }
}

/* example of augmenting a meter template with a
/* vendor specific action */
augment "/action:meter-template" +
    "/action:meter-entry" +
    "/action:meter-type" +
    "/action:one-rate-two-color-meter-type" +
    "/action:one-rate-two-color-meter" +
    "/action:exceed-action" +
    "/action:exceed-2color-meter-action-params" +
    "/action:exceed-2color-meter-action-val" {

    case exceed-2color-meter-action-drop {
```

```
        description
            "meter drop";
            uses action:drop;
    }
    case meter-action-mark-fwd-class {
        uses mark-fwd-class;
        description
            "Mark forwarding class in the packet";
    }
    case meter-action-mark-loss-priority {
        uses mark-loss-priority;
        description
            "Mark loss priority in the packet";
    }
}

augment "/action:meter-template" +
    "/action:meter-entry" +
    "/action:meter-type" +
    "/action:two-rate-tri-color-meter-type" +
    "/action:two-rate-tri-color-meter" +
    "/action:violate-action" +
    "/action:violate-3color-meter-action-params" +
    "/action:violate-3color-meter-action-val" {
    case exceed-3color-meter-action-drop {
        description
            "meter drop";
        uses action:drop;
    }

    description
        "Augment the actions to the two-color meter";
}

augment "/action:meter-template" +
    "/action:meter-entry" +
    "/action:meter-type" +
    "/action:one-rate-tri-color-meter-type" +
    "/action:one-rate-tri-color-meter" +
    "/action:violate-action" +
    "/action:violate-3color-meter-action-params" +
    "/action:violate-3color-meter-action-val" {
    case exceed-3color-meter-action-drop {
        description
            "meter drop";
        uses action:drop;
    }
}
```

```
    description
      "Augment the actions to basic meter";
  }
}
module example-compb-queue-policy {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:example-compb-queue-policy";
  prefix queue-plcy;

  import ietf-qos-classifier {
    prefix classifier;
    reference "RFC XXXX: YANG Model for QoS";
  }
  import ietf-qos-policy {
    prefix policy;
    reference "RFC XXXX: YANG Model for QoS";
  }

  organization "Company B";
  contact
    "Editor:   XYZ
     <mailto:xyz@compb.com>";

  description
    "This module defines a queue policy. The classification
     is based on a forwarding class, and the actions are queues.
     Copyright (c) 2019 IETF Trust and the persons identified as
     authors of the code. All rights reserved.
     Redistribution and use in source and binary forms, with or
     without modification, is permitted pursuant to, and subject
     to the license terms contained in, the Simplified BSD License
     set forth in Section 4.c of the IETF Trust's Legal Provisions
     Relating to IETF Documents
     (http://trustee.ietf.org/license-info).
     This version of this YANG module is part of RFC XXXX; see
     the RFC itself for full legal notices.";

  revision 2019-03-13 {
    description
      "Latest revision of diffserv policy";
    reference "RFC XXXX";
  }

  identity forwarding-class {
    base classifier:filter-type;
    description
      "Forwarding class filter type";
```

```
}

grouping forwarding-class-cfg {
  leaf forwarding-class-cfg {
    type string;
    description
      "forwarding-class name";
  }
  description
    "Forwarding class filter";
}

augment "/policy:policies" +
  "/policy:policy-entry" +
  "/policy:classifier-entry" +
  "/policy:filter-entry" {
  /* Does NOT support "logical-not" of forwarding class.
     Use "must"? */
  choice filter-params {
    description
      "Choice of filters";
    case forwarding-class-cfg {
      uses forwarding-class-cfg;
      description
        "Filter Type Internal-loss-priority";
    }
  }
  description
    "Augments Diffserv Classifier with fwd class filter";
}

identity compb-queue {
  base policy:action-type;
  description
    "compb-queue action type";
}

grouping compb-queue-name {
  container queue-name {
    leaf name {
      type string;
      description
        "Queue class name";
    }
  }
  description
    "compb queue container";
}
description
```



```
        "compb-queue grouping";
    }

    augment "/policy:policies" +
        "/policy:policy-entry" +
        "/policy:classifier-entry" +
        "/policy:classifier-action-entry-cfg" {
        choice action-cfg-params {
            description
                "Choice of action types";
            case compb-queue {
                uses compb-queue-name;
            }
        }
        description
            "Augment the queue actions to queue policy entry";
    }
}

module example-compb-queue {
    yang-version 1.1;
    namespace "urn:ietf:params:xml:ns:yang:ietf-compb-queue";
    prefix compb-queue;

    import ietf-qos-action {
        prefix action;
        reference "RFC XXXX: YANG Model for QoS";
    }

    organization "Company B";
    contact
        "Editor:   XYZ
         <mailto:xyz@compb.com>";

    description
        "This module describes a compb queue module. This is a
        template for a queue within a queue policy, referenced
        by name.

        This version of this YANG module is part of RFC XXXX; see
        the RFC itself for full legal notices.";

    revision 2019-03-13 {
        description
            "Latest revision of diffserv based classifier";
        reference "RFC XXXX";
    }
}
```

```
container compb-queue {
  description
    "Queue used in compb architecture";
  leaf name {
    type string;
    description
      "A unique name identifying this queue";
  }
  uses action:queue;
  container excess-rate {
    choice excess-rate-type {
      case percent {
        leaf excess-rate-percent {
          type uint32 {
            range "1..100";
          }
          description
            "excess-rate-percent";
        }
      }
      case proportion {
        leaf excess-rate-proportion {
          type uint32 {
            range "1..1000";
          }
          description
            "excess-rate-proportion";
        }
      }
    }
    description
      "Choice of excess-rate type";
  }
  description
    "Excess rate value";
}
leaf excess-priority {
  type enumeration {
    enum high {
      description "High Loss Priority";
    }
    enum medium-high {
      description "Medium-high Loss Priority";
    }
    enum medium-low {
      description "Medium-low Loss Priority";
    }
    enum low {
      description "Low Loss Priority";
    }
  }
}
```

```
    }
    enum none {
      description "No excess priority";
    }
  }
  description
    "Priority of excess (above guaranted rate) traffic";
}
container buffer-size {
  choice buffer-size-type {
    case percent {
      leaf buffer-size-percent {
        type uint32 {
          range "1..100";
        }
        description
          "buffer-size-percent";
      }
    }
    case temporal {
      leaf buffer-size-temporal {
        type uint64;
        units "microsecond";
        description
          "buffer-size-temporal";
      }
    }
    case remainder {
      leaf buffer-size-remainder {
        type empty;
        description
          "use remaining of buffer";
      }
    }
  }
  description
    "Choice of buffer size type";
}
description
  "Buffer size value";
}

augment
  "/compb-queue" +
  "/queue-cfg" +
  "/algorithmic-drop-cfg" +
  "/drop-algorithm" {
  case random-detect {
```

```
list drop-profile-list {
  key "priority";
  description
    "map of priorities to drop-algorithms";
  leaf priority {
    type enumeration {
      enum any {
        description "Any priority mapped here";
      }
      enum high {
        description "High Priority Packet";
      }
      enum medium-high {
        description "Medium-high Priority Packet";
      }
      enum medium-low {
        description "Medium-low Priority Packet";
      }
      enum low {
        description "Low Priority Packet";
      }
    }
    description
      "Priority of guaranteed traffic";
  }
  leaf drop-profile {
    type string;
    description
      "drop profile to use for this priority";
  }
}
description
  "compb random detect drop algorithm config";
}

module example-compb-scheduler-policy {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:" +
    "example-compb-scheduler-policy";
  prefix scheduler-plcy;

  import ietf-qos-action {
    prefix action;
    reference "RFC XXXX: YANG Model for QoS";
  }
}
```

```
import ietf-qos-policy {
  prefix policy;
  reference "RFC XXXX: YANG Model for QoS";
}

organization "Company B";
contact
  "Editor:   XYZ
   <mailto:xyz@compb.com>";

description
  "This module defines a scheduler policy. The classification
   is based on classifier-any, and the action is a scheduler.";

revision 2019-03-13 {
  description
    "Latest revision of diffserv policy";
  reference "RFC XXXX";
}

identity queue-policy {
  base policy:action-type;
  description
    "forwarding-class-queue action type";
}

grouping queue-policy-name {
  container compb-queue-policy-name {
    leaf name {
      type string;
      description
        "Queue policy name";
    }
  }
  description
    "compb-queue-policy container";
}
description
  "compb-queue policy grouping";
}

augment "/policy:policies" +
  "/policy:policy-entry" +
  "/policy:classifier-entry" +
  "/policy:classifier-action-entry-cfg" {
  choice action-cfg-params {
    case scheduler {
      uses action:scheduler;
    }
  }
}
```

```
        case queue-policy {
            uses queue-policy-name;
        }
        description
            "Augment the scheduler policy with a queue policy";
    }
}
```

### A.3. Example of Company C Diffserv Model

Company C vendor augmentation is based on Ericsson's implementation differentiated QoS. This implementation first sorts traffic based on a classifier, which can sort traffic into one or more traffic forwarding classes. Then, a policer or meter policy references the classifier and its traffic forwarding classes to specify different service levels for each traffic forwarding class.

Because each classifier sorts traffic into one or more traffic forwarding classes, this type of classifier does not align with `ietf-qos-classifier.yang`, which defines one traffic forwarding class per classifier. Additionally, Company C's policing and metering policies relies on the classifier's pre-defined traffic forwarding classes to provide differentiated services, rather than redefining the patterns within a policing or metering policy, as is defined in `ietf-diffserv.yang`.

Due to these differences, even though Company C uses all the building blocks of classifier and policy, Company C's augmentation does not use `ietf-diffserv.yang` to provide differentiated service levels. Instead, Company C's augmentation uses the basic building blocks, `ietf-qos-policy.yang` to provide differentiated services.

```
module example-compq-qos-policy {
    yang-version 1.1;
    namespace "urn:example-compq-qos-policy";
    prefix "compqos";

    import ietf-qos-policy {
        prefix "pol";
        reference "RFC XXXX: YANG Model for QoS";
    }

    import ietf-qos-action {
        prefix "action";
        reference "RFC XXXX: YANG Model for QoS";
    }
}
```

```
organization "";
contact "";
description "";

revision 2019-03-13 {
    description "";
    reference "";
}

/* identities */

identity compc-qos-policy {
    base pol:policy-type;
}

identity mdr-queuing-policy {
    base compc-qos-policy;
}

identity pwfq-queuing-policy {
    base compc-qos-policy;
}

identity policing-policy {
    base compc-qos-policy;
}

identity metering-policy {
    base compc-qos-policy;
}

identity forwarding-policy {
    base compc-qos-policy;
}

identity overhead-profile-policy {
    base compc-qos-policy;
}

identity resource-profile-policy {
    base compc-qos-policy;
}

identity protocol-rate-limit-policy {
    base compc-qos-policy;
}

identity compc-qos-action {
```

```

    base pol:action-type;
}

/* groupings */

grouping redirect-action-grp {
    container redirect {
        /* Redirect options */
    }
}

/* deviations */

deviation "/pol:policies/pol:policy-entry" {
    deviate add {
        must "pol:type = compc-qos-policy" {
            description
                "Only policy types driven from compc-qos-policy " +
                "are supported";
        }
    }
}

deviation "/pol:policies/pol:policy-entry/pol:classifier-entry" {
    deviate add {
        must "../per-class-action = 'true'" {
            description
                "Only policies with per-class actions have classifiers";
        }
        must "((../sub-type != 'mdrr-queuing-policy') and " +
            " (../sub-type != 'pwfq-queuing-policy')) or " +
            "(((../sub-type = 'mdrr-queuing-policy') or " +
            " (../sub-type = 'pwfq-queueing-policy')) and " +
            " ((classifier-entry-name = '0') or " +
            " (classifier-entry-name = '1') or " +
            " (classifier-entry-name = '2') or " +
            " (classifier-entry-name = '3') or " +
            " (classifier-entry-name = '4') or " +
            " (classifier-entry-name = '5') or " +
            " (classifier-entry-name = '6') or " +
            " (classifier-entry-name = '7') or " +
            " (classifier-entry-name = '8')))" {
            description
                "MDRR queuing policy's or PWFQ queuing policy's " +
                "classifier-entry-name is limited to the listed values";
        }
    }
}

```



```

deviation "/pol:policies/pol:policy-entry/pol:classifier-entry" +
    "/pol:classifier-action-entry-cfg" {
    deviate add {
        max-elements 1;
        must "action-type = 'compc-qos-action'" {
            description
                "Only compc-qos-action is allowed";
        }
    }
}

/* augments */

augment "/pol:policies/pol:policy-entry" {
    when "pol:type = 'compc-qos-policy'" {
        description
            "Additional nodes only for diffserv-policy";
    }
    leaf sub-type {
        type identityref {
            base compc-qos-policy;
        }
        mandatory true;
        /* The value of this leaf must not change once configured */
    }
    leaf per-class-action {
        mandatory true;
        type boolean;
        must "(((. = 'true') and " +
            " (../sub-type = 'policing-policy') or " +
            " (../sub-type = 'metering-policy') or " +
            " (../sub-type = 'mdrr-queuing-policy') or " +
            " (../sub-type = 'pwfq-queuing-policy') or " +
            " (../sub-type = 'forwarding-policy')) or " +
            " ((. = 'false') and " +
            " (../sub-type = 'overhead-profile-policy') or " +
            " (../sub-type = 'resource-profile-policy') or " +
            " (../sub-type = 'protocol-rate-limit-policy')))" {
            description
                "Only certain policies have per-class action";
        }
    }
}
container traffic-classifier {
    presence true;
    when "../sub-type = 'policing-policy' or " +
        "../sub-type = 'metering-policy' or " +
        "../sub-type = 'forwarding-policy'" {
        description

```

```
        "A classifier for policing-policy or metering-policy";
    }
    leaf name {
        type string;
        mandatory true;
        description
            "Traffic classifier name";
    }
    leaf type {
        type enumeration {
            enum 'internal-dscp-only-classifier' {
                value 0;
                description
                    "Classify traffic based on (internal) dscp only";
            }
            enum 'ipv4-header-based-classifier' {
                value 1;
                description
                    "Classify traffic based on IPv4 packet header fields";
            }
            enum 'ipv6-header-based-classifier' {
                value 2;
                description
                    "Classify traffic based on IPv6 packet header fields";
            }
        }
        mandatory true;
        description
            "Traffic classifier type";
    }
}

container traffic-queue {
    when "(../sub-type = 'mdrr-queuing-policy') or " +
        "(../sub-type = 'pwfq-queuing-policy') " {
        description
            "Queuing policy properties";
    }
    leaf queue-map {
        type string;
        description
            "Traffic queue map for queuing policy";
    }
}

container overhead-profile {
    when "(../sub-type = 'overhead-profile-policy') " {
        description
            "Overhead profile policy properties";
    }
}
```

```
    }
    container resource-profile {
      when "../sub-type = 'resource-profile-policy'" {
        description
          "Resource profile policy properties";
      }
    }
    container protocol-rate-limit {
      when "../sub-type = 'protocol-rate-limit-policy'" {
        description
          "Protocol rate limit policy properties";
      }
    }
  }
}

augment "/pol:policies/pol:policy-entry/pol:classifier-entry" +
  "/pol:classifier-action-entry-cfg/pol:action-cfg-params" {
  when "../.../pol:type = 'compq-qos-policy'" {
    description
      "Configurations for a classifier-policy-type policy";
  }
  case metering-or-policing-policy {
    when "../.../sub-type = 'policing-policy' or "
      + "../.../sub-type = 'metering-policy'" {
    }
    container dscp-marking {
      uses action:dscp-marking;
    }
    container precedence-marking {
      uses action:dscp-marking;
    }
    container priority-marking {
      uses action:priority;
    }
    container rate-limiting {
      uses action:one-rate-two-color-meter;
    }
  }
  case mdrd-queuing-policy {
    when "../.../sub-type = 'mdrd-queuing-policy'" {
      description
        "MDRR queue handling properties for the traffic " +
        "classified into current queue";
    }
    leaf mdrd-queue-weight {
      type uint8 {
        range "20..100";
      }
    }
  }
}
```

```
        units percentage;
    }
}
case pwfq-queuing-policy {
    when "../../../sub-type = 'pwfq-queuing-policy'" {
        description
            "PWFQ queue handling properties for traffic " +
            "classified into current queue";
    }
    leaf pwfq-queue-weight {
        type uint8 {
            range "20..100";
        }
        units percentage;
    }
    leaf pwfq-queue-priority {
        type uint8;
    }
    leaf pwfq-queue-rate {
        type uint8;
    }
}
case forwarding-policy {
    when "../../../sub-type = 'forwarding-policy'" {
        description
            "Forward policy handling properties for traffic " +
            "in this classifier";
    }
    uses redirect-action-grp;
}
description
    "Add the classify action configuration";
}
}
```

#### Authors' Addresses

Aseem Choudhary  
Cisco Systems  
170 W. Tasman Drive  
San Jose, CA 95134  
US

Email: asechoud@cisco.com

Mahesh Jethanandani  
Cisco Systems  
170 W. Tasman Drive  
San Jose, CA 95134  
US

Email: mjethanandani@gmail.com

Norm Strahle  
Juniper Networks  
1194 North Mathilda Avenue  
Sunnyvale, CA 94089  
US

Email: nstrahle@juniper.net

Ebben Aries  
Juniper Networks  
1194 North Mathilda Avenue  
Sunnyvale, CA 94089  
US

Email: exa@juniper.net

Ing-Wher Chen  
Jabil

Email: ing-wher\_chen@jabil.com

Routing Area Working Group  
Internet-Draft  
Intended status: Informational  
Expires: January 3, 2020

S. Bryant  
U. Chunduri  
T. Eckert  
Futurewei Technologies Inc  
July 02, 2019

Preferred Path Loop-Free Alternate (pLFA)  
draft-bryant-rtgwg-plfa-00

Abstract

Fast re-route (FRR) is a technique that allows productive forwarding to continue in a network after a failure has occurred, but before the network has time to re-converge. This is achieved by forwarding a packet on an alternate path that will not result in the packet looping. Preferred Path Routing (PPR) provides a method of injecting explicit paths into the routing protocol. The use of PPR to support FRR has a number of advantages. This document describes the advantages of using PPR to provide a loop-free alternate FRR path, and provides a framework for its use in this application.

Status of This Memo

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

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

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

This Internet-Draft will expire on January 3, 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 (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents

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

## Table of Contents

1. Introduction . . . . .	2
1.1. A Note on the term IPFRR . . . . .	3
2. PPR Overview . . . . .	4
3. Preferred Path LFA (pLFA) Deployment Advantages . . . . .	4
4. Simple Repair Using pLFA . . . . .	6
4.1. Link Repair . . . . .	6
4.2. Node Repair . . . . .	7
4.3. Shared Risk Link Groups . . . . .	8
4.4. Local Area Networks . . . . .	9
4.5. Multiple Independent Failures . . . . .	9
4.6. Multi-homed Prefixes . . . . .	9
4.7. ECMP . . . . .	9
5. Repair To A Traffic Engineered Alternate Path . . . . .	10
6. Use of a Repair Graph . . . . .	11
6.1. Single Repair Graph . . . . .	11
6.2. Multiple Disjoint Graphs . . . . .	11
7. Centralized and Decentralized Approaches . . . . .	13
8. Independence of operation . . . . .	14
9. Data-plane Considerations . . . . .	14
9.1. Traditional IP . . . . .	15
9.2. Segment Routing over an IPv6 Data Plane (SRv6) . . . . .	15
9.3. MPLS . . . . .	15
10. Loop Free Convergence . . . . .	16
11. OAM Considerations . . . . .	16
12. Privacy Considerations . . . . .	16
13. Security Considerations . . . . .	17
14. IANA Considerations . . . . .	17
15. References . . . . .	17
15.1. Normative References . . . . .	17
15.2. Informative References . . . . .	18
Authors' Addresses . . . . .	19

## 1. Introduction

### Preferred Path Routing (PPR)

[I-D.chunduri-lsr-isis-preferred-path-routing] is a method of introducing explicit paths to a network. Such a path may be any loop-free path between two points in the network that satisfies the need for which the path was created. The PPR path is not constrained to be the shortest path between any points in the network, although

the use of shortest path segments is provided for in order to compress the size of the path description flooded by the routing protocol. The advantages of PPR over alternate methods of creating such paths is described in [I-D.chunduri-lsr-isis-preferred-path-routing].

A packet is carried over the network in an appropriate form using the Preferred Path Routing Identifier (PPR-ID) as the data plane identifier to map the packet to the PPR path, and hence the resources and next-hop (NH). One way of adding a PPR-ID to a packet would be to encapsulate it, but PPR does not restrict the user to the use of encapsulation. How the PPR-ID is carried in the general case is outside the scope of this document. Various methods of adding the PPR-ID to a packet for the purposes of Fast-Reroute (FRR) are described in Section 9.

IP Fast Re-route (IPFRR) Section 1.1 and the methods known at the time of its writing is described in [RFC5714]. A number of later methods are described in [RFC6981], [RFC7490], [RFC7812] and [I-D.ietf-rtgwg-segment-routing-ti-lfa].

This document is a framework describing various methods whereby PPR can be used to provide IP Fast Reroute (IPFRR) paths. PPR can provide IPFRR in a number of ways.

- o Signaling pre-computed preferred alternatives for the primary path
- o Signaling individual segments on the repair path.
- o Selective overriding of locally computed Loop Free Alternates (LFA) for the NH failure.
- o Local repair to a Traffic Engineered paths avoiding the need for multi-hop Bidirectional Forwarding Detection (BFD) [RFC5880].
- o Micro-loop elimination [RFC5715].

These are described in more detail within this memo.

#### 1.1. A Note on the term IPFRR

The term IP fast re-route (IPFRR) was adopted by the IETF as the general name for best-effort Fast Re-route (FRR) in best effort IP and MPLS networks. This was to distinguish this new work from the then established FRR as described in [RFC4090] which uses RSVP Traffic Engineered (RSVP-TE) MPLS paths [RFC3936].

Within this document the terms IPFRR and FRR are used interchangeably.



## 2. PPR Overview

PPR works by injecting into the network a path or a graph and a corresponding forwarding identifier (PPR-ID). A node examines each PPR path description and if it is on the path it inserts into the Forwarding Information Database (FIB) an entry for the PPR-ID with the next hop as either the next entry along the PPR path, or if a loose path is specified, the next hop on the shortest path to the next hop along the PPR path. This is described in [I-D.chunduri-lsr-isis-preferred-path-routing].

PPR also has the ability to inject into a network a tree rooted at a node identified a PPR-ID. This is described in [I-D.ce-lsr-ppr-graph]. This graph mechanism provides a compact representation of a set of paths to a given PPR-ID. This works in a similar manner to the linear path case, in which a node on the graph inserts a FIB entry for the PPR-ID with the next hop as either the next node in the graph, or the next hop on the shortest path to the next node in the graph. Clearly the graph needs to be a spanning tree and must not contain a cycle.

In the description of the FRR methods provided in the text, the term encapsulation (and decapsulation) is frequently used in connection with the addition (and removal) of a PPR-ID to be used by the forwarding later to identify to the forwarders the PPR path that the packet needs to traverse to be follow the repair path. Encapsulation is only one of a number of methods that can be used and is used in this memo as a convenience without loss of generality. For more information see Section 9.

## 3. Preferred Path LFA (pLFA) Deployment Advantages

PPR allows the construction of arbitrary engineered backup paths. In this respect it is like similar to RSVP-TE and Topology Independent Loop-Free Alternates (TI-LFA) [I-D.ietf-rtgwg-segment-routing-ti-lfa]. However, unlike those approaches PPR is applicable to any forwarding plane. For example, it is possible to support MPLS, both IPv4 and IPv6 and Ethernet.

Like Segment Routing (SR) [RFC8402], PPR uses extensions to the existing IGP, however, unlike SR, PPR requires no extension to the data plane. Again, unlike SR, which requires a Segment Identifier (SID) in the network layer header for every non-shortest path forwarding instruction, an arbitrary path does not require expansion of the user data packet beyond that needed for the initial insertion of the PPR-ID. This mitigates the MTU stress that SR introduces to the network.

PPR based IPFRR supports 100% failure coverage similar to RSVP-TE [RFC4090], TI-LFA, Maximally Redundant Trees (MRT) [RFC7812] and Not-Via [RFC6981]. It does not have the coverage restrictions that apply to Loop-Free Alternate (LFA) [RFC5286] and Remote LFA (RLFA) [RFC7490].

Shared Risk Link Groups (SRLGs) make it more difficult find repairs in LFA and RLFA reducing repair coverage. TI-LFA can address this, but only at a cost of expanding the number of SIDs and hence the packet size.

Supporting multiple concurrent failures is difficult in all of the IPRFF approaches except MRT, which can repair two concurrent failures. However unlike MRT, which is constrained by its network wide algorithm, PPR allows individual, arbitrary repair paths to be instantiated, for any failure.

In the current TI-LFA design, priority is given to repairing connectivity rather than conforming to the operator traffic policy. A PPR based FFR approach can apply policy to the repaired traffic, including, if required multiple policies to an individual failure.

One of the main advantages of TI-LFA compared to other IPFRR approaches is that it creates repair paths that are congruent with the post convergence path from the Point of Local Repair (PLR) [RFC4090] to the destination. These paths, which may be longer than strictly necessary to reach Q-space [I-D.bryant-ipfrr-tunnels], stop micro-loops from forming along the repair path during re-convergence. PPR can also create these congruent paths without the need to introduce SR into the network.

One of the limitations in TI-LFA, RLFA and LFA is that they do not have a method of selectively creating alternative next-hops or indeed full repair paths based on policy, or traffic engineering information known to the operator. PPR provides a simple way to inject arbitrary paths. It may therefore be used to enhance an existing LFA/RLFA/TI-LFA IPFRR enabled network by selectively injecting paths to provide a repair for business critical links with a policy in the PLR that where provided a PPR path should be preferred over a local calculated LFA based paths.

PPR is applicable to both centralized and PLR computed repair paths each of which has advantages in different circumstances. A centrally computed repair path only requires interaction with one network node which then floods the instruction. This differs from the normal SDN approach which requires interaction with all of the nodes along the path and RSVP-TE which requires interaction with at least one end-point of every repair.

Like TI-LFA, pLFA is based on a small extension to the IGP. It uses the IGP flooding mechanism and in-built state maintenance and consistency checks. This contrasts with RSVP-TE which needs its own separate Signaling and soft-state maintenance method.

The requirement that the pLFA solution addresses is thus the ability to construct repair paths that conform to operator policy without data-plane changes or significant MTU increase, and without introducing any control plane changes other than a small addition to the existing IGP.

A more detailed technical comparison between pLFA and the existing solutions is provided in the technical description of pLFA that follows.

#### 4. Simple Repair Using pLFA

##### 4.1. Link Repair

In this, the most basic, scenario Figure 1 we assume that we have a path A-B-C-D that the packet must traverse. This may be a normal best effort path or a traffic engineered path.

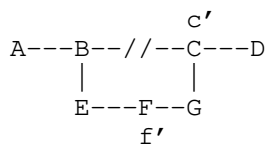


Figure 1: Simple IPFRR Using pLFA

PPR is used to inject the repair path B->E->F->G->C into the network with a PPR-ID of c'. B is monitoring the health of link B->C, for example looking for loss-of-light, or using Bidirectional Forwarding Detection (BFD) [RFC5880]. When B detects a failure it encapsulates the packet to C by adding to the packet the PPR-ID for c' and sending it to E. At C the packet is decapsulated and sent to D. The path C->E->F->G->C may be a traffic engineered path or it may be a best effort path. B may have at its disposal multiple paths to C with different properties for different traffic classes. In this case each path to be used would require its own PPR-ID (c', c'' etc).

In some circumstances, the repair path may be terminated at another point in Q-space or at a node between C and D. For example, in Figure 1 if all costs are 1, F is in Q-space with respect to a B->C failure (F->G->C cost = 2, whilst F->E->B->C cost = 3) and thus the packet can safely be encapsulated and sent to F with a PPR-ID of f'. Releasing the packet early in Q-space has two advantages, firstly the



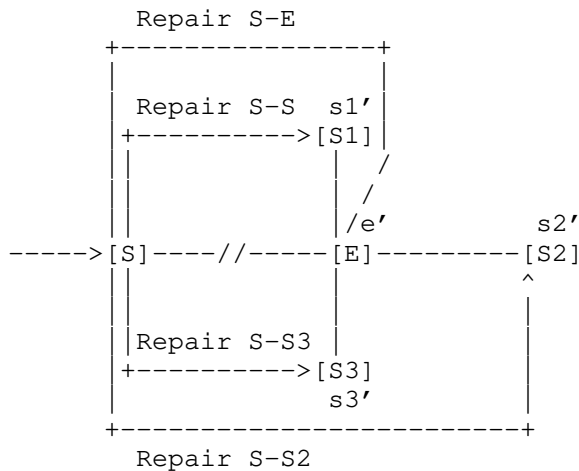


Figure 3: Simple IPFRR - Node Failure

Node S needs the use of four repair paths to address the failure of node E, one repair to each of E's neighbours for which E is on the path to that neighbour. In Figure 3 there are three of these next-next-hop repairs, noted as Repair S-Sx in the figure. In addition a repair to E (Repair S->E) using a path other than along the path S-E> should be installed for traffic to E on the basis that the problem may be a failure of link S->E rather than a failure of the node.

The three repair paths to the next-next-hops of E can be installed as PPR path S-Sx with a PPR-ID of sx'. The link repair for E a PPR path to E which avoids link S-E with a PPR-ID of e'.

#### 4.3. Shared Risk Link Groups

A shared risk link group (SRLG) is a set of links that are believed to have some systematic connection such that when one fails there is a high probability of all of them failing. This occurs, for example, where all of the members of the group run in a common cable duct. Where this relationship is known, and the simultaneous failure does not partition the network, PPR can install paths such that all members of the SRLG are avoided. pLFA has fewer constraints than other methods in constructing arbitrary repair paths in the network. [RFC6981] Section 6.1 describes the SRLG problem as it applies to IPFRR. pLFA can address all of the cases described in [RFC6981].

SRLG avoiding IPFRR paths can be complex. Since a packet can be attracted towards the failure whenever it is released from a strict path, the repair path may need a number of segments to steer it safely into Q-space. If this is done in the data-plane this can

stress the MTU. pLFA creates the path in the control plane and its encapsulation is invariant with respect to the complexity of the path. Furthermore, if the need to reduce the data-plane encapsulation side means that the repair path needs to use a sequence of loose hops it is necessary to determine the behaviour of each router on the chosen path. This contrasts with pLFA which can determine the path using whatever metrics and policy is appropriate, and then simply impose it without any data-plane overhead beyond that needed for a simple repair.

#### 4.4. Local Area Networks

LANs are a special type of SRLG and are solved using the SRLG mechanisms outlined above. With all SRLGs, there is a trade-off between the sophistication of the fault detection and the size of the SRLG. [RFC6981] Section 6.2 describes the LAN problem as it applies to IPFRR. pLFA can address all of the cases described in [RFC6981].

#### 4.5. Multiple Independent Failures

The Multiple Independent Failure cases described in [RFC6981] Section 6.3 will be analyzed in a future version of this document.

#### 4.6. Multi-homed Prefixes

The Multi-Homed Prefix (MHP) problem is described in [RFC5286] Section 6.1, [RFC6981] Section 5.3 and [RFC8518]. MHP will be addressed in a future version of this document.

#### 4.7. ECMP

Equal Cost Multi-Path (ECMP) is a consideration in any IPFRR method that does not use strict paths, and can be both an opportunity and threat. It is an opportunity in that it allows for the repair traffic to be distributed over a number of alternative paths to minimize congestion. If a loose pLFA path is injected into the network, then any available ECMP paths that fulfill the PPR path constraints can be installed following the same procedure used in normal IGP path computation.

However, care must be taken that a packet is not in a position where it is released from a repair at an ECMP point such that one of the ECMP paths is back via the failure. This can never happen if the correct definition of Q-space [RFC7490] is used in calculating the repair path.

## 5. Repair To A Traffic Engineered Alternate Path

In this approach there are two traffic engineered paths from A to D (Figure 4).

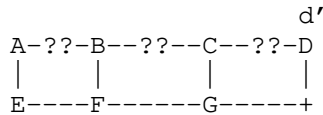


Figure 4: Traffic Engineered IPFRR Using pLFA

The primary path A->B->C->D is protected by a traffic engineered path A->F->G->D (PPR-ID d') with traffic engineered connectors from B (B->F) and C (C->G). The path A->F->G->D and its connectors can be created and injected by any node with access to the IGP, but it is more likely to be created by a traffic engineering controller.

If link B->C fails, B re-routes packets destined for D to the traffic engineered path A->F->G->D via connector B->F. It does this by encapsulating the packet with a PPR-ID of d'.

Clearly there is nothing special needed to get the packet from B to F as they are adjacent but if there is a node say X on the path from B to F an explicit path needs to be created from B to F via X. Normally the repair would be created as a single PPR path (i.e. B->F->G->D) with a PPR-ID of d'. In this approach the repair from A would be A->F->G->D with a PPR-ID of d' also. Similarly C->G->D would again share the PPR-ID d'.

If preferred the repair path could also be constructed using double encapsulation or using an SR approach in which the first segment was B-F with a PPR-ID/SID f' and the second segment was F-D with PPR-ID of d'.

In the example shown in Figure 4 the proposed B-/->C protection path was B->F->G->D. This is node protecting on C since the repair path avoids C. Although link failures tend to be more common than node failures some critical applications would prefer node protection where possible. Node avoidance may not be possible within the network, and may come at a cost of increased path repair path length. However, whether to include node protection and at what cost to accept its inclusion is a matter of network operator policy.

The repair constructed in this section required the inclusion of a set of PPR defined links to construct the repair. PPR has the ability to construct graphs [I-D.ce-lsr-ppr-graph] which can simplify

the specification of the required repair topology. This is discussed in Section 6.

## 6. Use of a Repair Graph

PPR has the ability to inject graphs into a network as well as linear paths [I-D.ce-lsr-ppr-graph]. PPR graphs specify the paths from a set of nodes to a single node, and are a compact method of representing a set of paths to that destination with shared properties.

### 6.1. Single Repair Graph

In [I-D.ce-lsr-ppr-graph] the S bit in the PPR Path Description Element (PDE) specifies that a network node is a Source and a D bit specifies that it is a destination. A graph with all S bits set on the leaves and a D bit on the root is a unidirectional tree.

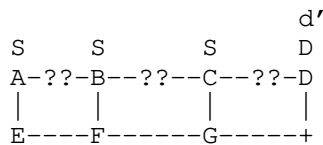


Figure 5: pLFA using PPR Graphs

Consider the network fragment shown in Figure 5. A graph with a PPR-ID  $d'$  is constructed attaching each of the nodes A, B, and C to D. Should any of the nodes A, B or C fail the packet can be forwarded on the PPR graph to D with the PPR-ID of  $d'$ . In the unidirectional repair graph A, B, and C are all sources (signaled with the S bit set), and D is the only destination (signaled with the D bit set).

### 6.2. Multiple Disjoint Graphs

Consider Figure 1 from [RFC7812] which illustrates the problem of IPFRR in a network that is 2-connected.



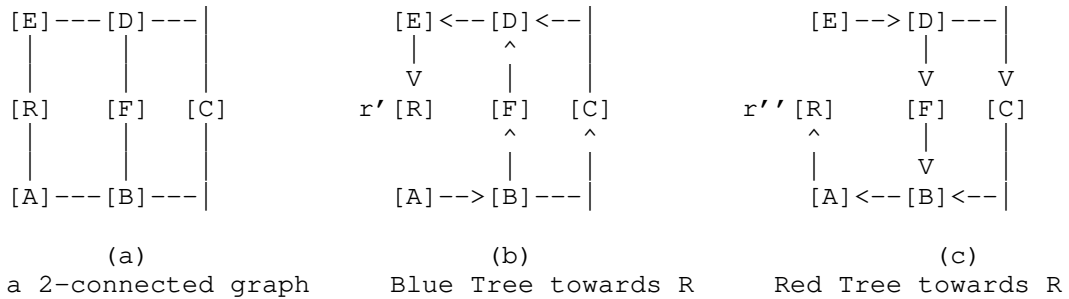


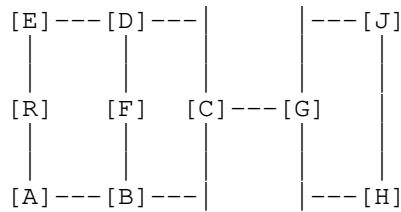
Figure 6: A 2-Connected Network

Figure 6(a) is the full network, and Figure 6(b) and (b) are two corresponding redundant trees from [RFC7812]. Using the Red and Blue trees towards R every node has at least two paths to R. We give R a PPR-ID of  $r'$  in the Blue tree and a PPR-ID of  $r''$  in the Red tree. R is the only destination in the PPR graph (D bit set), but all other nodes are sources (S bit set). For clarity this bit setting is not shown in Figure 6.

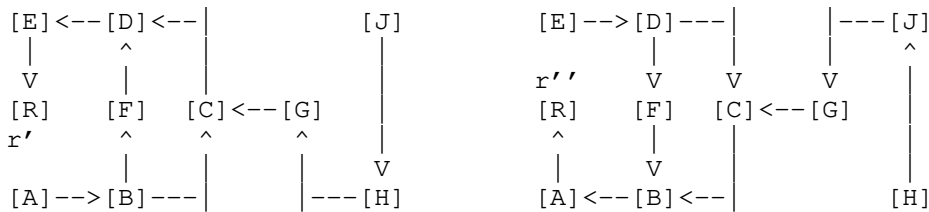
It is worth noting what happens at nodes B and D in Figure 6(b). B is an ECMP to D via F and C. What happens at node B is a matter of implementation and operator preference. Either B can choose one of the next-hops, or it use them as an ECMP pair. It can also use the availability of the pair to protect against B->F or B->C being an unexpected SRLG with respect to link A->R. D is a merge point for traffic destined for  $r'$  arriving from F and from C. It simply forwards the traffic to  $r'$  as normal. Similarly in Figure 6(c) D can sent traffic to  $r''$  via F or C.

Whilst in this example the Red and Blue trees use exactly the same links and nodes used by the main topology, a repair graph could use available nodes and links outside this network fragment.

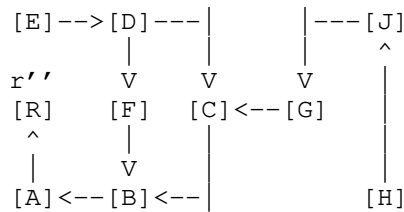
Now consider Figure 2 from [RFC7812] which illustrates the problem of IPFRR in a network that is not 2-connected.



(a) a graph that is not 2-connected



(b) Blue Tree towards R



(c) Red Tree towards R

Figure 7: A Network That Is Not 2-Connected

Again there are two paths (with PPR-IDs  $r'$  and  $r''$ ) to R from all nodes except that G, J and H all depend on link  $G \rightarrow C$  and node C which is a single point of failure in the network.

Again note that B in the Blue tree and D in the Red tree has two paths to  $r'$  and  $r''$  respectively that it may use according to configuration or preference.

## 7. Centralized and Decentralized Approaches

pLFA paths can be established through both centralized and decentralized approaches.

A centralized system has a more holistic view of the network and its policies, its resource constraints and resource usage. A decentralized system is inherently more resilient to failure and is a good fit where the network is a simple best effort network as is commonly deployed.

A centralized system gathers the network state, just as any SDN system does, and computes the FRR paths needed. However, unlike normal SDN operation where the controller needs to individually instruct every entity on the path for every path, in a PPR network it is only necessary to inject the PPR path at one point. In practice, for reliability, it would inject the PPR paths in a small number of places, and the naturally reliability of the IGP would ensure the

complete distribution of the paths. Furthermore, the system collecting the network state would naturally send the PPR LSPs back to the SDN controller providing quality assurance that the FRR paths had been distributed.

In a decentralized approach the pLFA path is computed within the network, normally by the PLR. Further details of this approach will be provided in a future version of this document.

## 8. Independence of operation

Each PPR path is independent of all other paths in the network. This means that there is no constraint on how the path is calculated, and a different algorithm can be used on every path. Some of the other FRR approaches have this property, but not all. For example LFA is constrained by the properties of the base IGP as to a large degree is RLFA. PPR can incorporate best effort segments if required, but from a data-plane perspective there is no advantage in doing so. In this case there is a dependence on the path choice in the base routing protocol.

MRT and Not-Via can use any algorithm to calculate the repair, but it needs to be common across the network, although the expectation in the case of Not-Via is that the algorithm would be a Dijkstra based SPF calculation. In both these cases to change the algorithm would require turning off FRR for the whole network, re-configuring and then restarting FRR.

RSVP-TE based FRR can specify any path, but at the cost of maintaining the soft-state.

A PLR in a TI-LFA or any SR based approach can also compute paths independent of each other, but they tend to need to do this as a concatenation of a series of shortest paths in order to reduce the number of SIDs they need to form the path. TI-LFA is thus highly dependent on the underlying best effort paths.

pLFA can be used as a method of converting classic LFA or RLFA to full coverage by providing the paths that these methods are unable to support, or to provide any the sub-paths needed to reduce the number of TI-LFA SIDs.

## 9. Data-plane Considerations

This section is a survey of a number of data-planes in each case considering how a PPR-ID could added to map the packet to required FRR path.

### 9.1. Traditional IP

Where the data-plane is "traditional" IP the user packet needs to be encapsulated such that the outer IP address is the PPR-ID. Any preferred encapsulation can be used such as: IP in IP, IP in GRE, or IP in UDP.

The tunnel capabilities of a node can be advertised using the method described in [I-D.xu-isis-encapsulation-cap] allowing different tunnel types to be used for different PPR paths, depending on the capability of the various nodes in the network.

A common operational issue with this type of encapsulation for IPFRR has been the shortage of IP addresses. However this is not an issue in an IPv6 network.

### 9.2. Segment Routing over an IPv6 Data Plane (SRv6)

Where the data-plane is SRv6 [I-D.ietf-6man-segment-routing-header] pLFA would be used to steer a packet towards the next segment endpoint. Clearly an extra level of IP encapsulation could be used Section 9.1, but that expands the packet by adding at least 36 octets.

Where the packet is a "traditional" IP packet, and the repair endpoint is SRv6 capable, an alternative to the methods described in Section 9.1 is to insert an SRH into the IP packet setting the SID in the SRH to the original packet DA and replacing the outer DA with the PPR-ID. If this method is used the semantics of the PPR-ID must include the reconstruction of the packet, by replacing the DA with the original DA retrieved from the SRH and the removal of the SRH.

### 9.3. MPLS

Where the data-plane is MPLS any encapsulation needed is tiny (a label push), but the exact action depends on the repair strategy, and there is the usual FRR problem of the setting of the new value for the top label prior to pushing the PPR-ID label.

Where the FRR path terminates at an MPLS node other than the network egress provider edge (PE) in the type of pLFA repair described in Section 4, the original top label needed to be set to the label the node was expecting.

Consider the network fragment shown in Figure 1. This is straight forward case because node B swaps the top label the label it would have used without the failure and then pushes the label that corresponds to c'. If the repair strategy had been to exit Q-space

at the earliest opportunity for example at F, then B would have needed to know what label F required to reach the destination. A very similar problem occurs when a node repair is undertaken Figure 3, where S needs to know the label that the next-next-hops (S1, S2 and S3) need to reach the destination.

Where the traffic is being moved to a new path terminating at the egress PE as shown in Figure 4, the problem much simpler and only requires the swapping of the top label with the label that represents d'.

#### 10. Loop Free Convergence

Whilst IPFRR puts in place a temporary network repair, eventually the network needs to re-converge around the surviving network components. During this phase there is a danger that micro-loops will form and disrupt the traffic flowing across the network. A similar problem can occur when the failed component returns to service, or when a new component is introduced into the network. [RFC5715] describes the problem of loop-free convergence in detail and examines the methods known at the time of its writing. Since that time [RFC8333] has proposed a timer based loop mitigation (but not elimination) process, and [I-D.ietf-rtgwg-segment-routing-ti-lfa] has proposed that by making the IPFRR path congruent with the post convergence path loops can be eliminated along the repair path. However whilst these mitigation techniques address component failure, neither are targeted at the repair/new component case.

These problems only effect best effort paths and path segments, fully defined paths do not have this problem.

A network using pLFA is compatible with all of the know loop-free convergence and loop mitigation approaches.

#### 11. OAM Considerations

PPR may also be used in a way that provides an alternative to running multi-hop BFD from ingress on a traffic engineered (TE) path with reducing the complexities that arise from echo reply false alarms. In this use case pLFA works by locally detecting the failure and transferring the traffic to preferred TE backups which are in time replaced by the newly computed TE paths to the same PPR-ID.

#### 12. Privacy Considerations

As noted in Section 13 pLFA paths are constrained by the routing domain and thus the traffic will be no more subject to observation than it would in normal operation. Indeed PPR has the capability to

constrain the path of the traffic more tightly than other IPFRR approaches. pLFA therefore does not reduce the privacy of user traffic on the network.

### 13. Security Considerations

The security considerations of PPR are discussed in [I-D.chunduri-lsr-isis-preferred-path-routing] which in turn refers the reader to the security considerations of the underlying routing protocol and the data-plane in use. The pLFA application of PPR to IPFRR introduces no additional security regarding PPR itself.

General IPFRR security considerations are discussed in [RFC5714] and these apply to this solution.

One further consideration, is the whether policy that applied to the original path needs to be applied to the repair path. The decision is operator and application specific, however pLFA is better than some other IPFRR solution in that it is possible to precisely choose the repair path.

IPFRR is deployed within the scope of the routing protocol that underpins it which limits the security vulnerability. Furthermore it is unlikely that IPFRR would be deployed outside a well managed network. These restrictions in-turn significantly mitigate any security threat.

### 14. IANA Considerations

This document makes no IANA requests.

### 15. References

#### 15.1. Normative References

- [I-D.ce-lsr-ppr-graph]  
Chunduri, U. and T. Eckert, "Preferred Path Route Graph Structure", draft-ce-lsr-ppr-graph-02 (work in progress), May 2019.
- [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-03 (work in progress), May 2019.

## 15.2. Informative References

- [I-D.bryant-ipfrr-tunnels]  
Bryant, S., Filsfils, C., Previdi, S., and M. Shand, "IP Fast Reroute using tunnels", draft-bryant-ipfrr-tunnels-03 (work in progress), November 2007.
- [I-D.ietf-6man-segment-routing-header]  
Filsfils, C., Dukes, D., Previdi, S., Leddy, J., Matsushima, S., and d. daniel.voyer@bell.ca, "IPv6 Segment Routing Header (SRH)", draft-ietf-6man-segment-routing-header-21 (work in progress), June 2019.
- [I-D.ietf-rtgwg-segment-routing-ti-lfa]  
Litkowski, S., Bashandy, A., Filsfils, C., Decraene, B., Francois, P., daniel.voyer@bell.ca, d., Clad, F., and P. Camarillo, "Topology Independent Fast Reroute using Segment Routing", draft-ietf-rtgwg-segment-routing-ti-lfa-01 (work in progress), March 2019.
- [I-D.xu-isis-encapsulation-cap]  
Xu, X., Decraene, B., Raszuk, R., Chunduri, U., Contreras, L., and L. Jalil, "Advertising Tunnelling Capability in IS-IS", draft-xu-isis-encapsulation-cap-07 (work in progress), October 2016.
- [RFC3936] Kompella, K. and J. Lang, "Procedures for Modifying the Resource reSerVation Protocol (RSVP)", BCP 96, RFC 3936, DOI 10.17487/RFC3936, October 2004, <<https://www.rfc-editor.org/info/rfc3936>>.
- [RFC4090] Pan, P., Ed., Swallow, G., Ed., and A. Atlas, Ed., "Fast Reroute Extensions to RSVP-TE for LSP Tunnels", RFC 4090, DOI 10.17487/RFC4090, May 2005, <<https://www.rfc-editor.org/info/rfc4090>>.
- [RFC5286] Atlas, A., Ed. and A. Zinin, Ed., "Basic Specification for IP Fast Reroute: Loop-Free Alternates", RFC 5286, DOI 10.17487/RFC5286, September 2008, <<https://www.rfc-editor.org/info/rfc5286>>.
- [RFC5714] Shand, M. and S. Bryant, "IP Fast Reroute Framework", RFC 5714, DOI 10.17487/RFC5714, January 2010, <<https://www.rfc-editor.org/info/rfc5714>>.
- [RFC5715] Shand, M. and S. Bryant, "A Framework for Loop-Free Convergence", RFC 5715, DOI 10.17487/RFC5715, January 2010, <<https://www.rfc-editor.org/info/rfc5715>>.

- [RFC5880] Katz, D. and D. Ward, "Bidirectional Forwarding Detection (BFD)", RFC 5880, DOI 10.17487/RFC5880, June 2010, <<https://www.rfc-editor.org/info/rfc5880>>.
- [RFC6981] Bryant, S., Previdi, S., and M. Shand, "A Framework for IP and MPLS Fast Reroute Using Not-Via Addresses", RFC 6981, DOI 10.17487/RFC6981, August 2013, <<https://www.rfc-editor.org/info/rfc6981>>.
- [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>>.
- [RFC7812] Atlas, A., Bowers, C., and G. Enyedi, "An Architecture for IP/LDP Fast Reroute Using Maximally Redundant Trees (MRT-FRR)", RFC 7812, DOI 10.17487/RFC7812, June 2016, <<https://www.rfc-editor.org/info/rfc7812>>.
- [RFC8333] Litkowski, S., Decraene, B., Filsfils, C., and P. Francois, "Micro-loop Prevention by Introducing a Local Convergence Delay", RFC 8333, DOI 10.17487/RFC8333, March 2018, <<https://www.rfc-editor.org/info/rfc8333>>.
- [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>>.
- [RFC8518] Sarkar, P., Ed., Chunduri, U., Ed., Hegde, S., Tantsura, J., and H. Gredler, "Selection of Loop-Free Alternates for Multi-Homed Prefixes", RFC 8518, DOI 10.17487/RFC8518, March 2019, <<https://www.rfc-editor.org/info/rfc8518>>.

## Authors' Addresses

Stewart Bryant  
Futurewei Technologies Inc  
  
Email: [stewart.bryant@gmail.com](mailto:stewart.bryant@gmail.com)

Uma Chunduri  
Futurewei Technologies Inc  
  
Email: [uchundur@futurewei.com](mailto:uchundur@futurewei.com)



Toerless Eckert  
Futurewei Technologies Inc

Email: tte+ietf@cs.fau.de

INTERNET-DRAFT  
Intended status: Proposed Standard

S. Hu  
China Mobile  
D. Eastlake  
Futurewei Technologies  
M. Chen  
Huawei Technologies  
F. Qin  
Z. Li  
China Mobile  
T. Chua  
Singapore Telecommunications  
D. Huang  
ZTE  
July 3, 2019

Expires: January 2, 2020

Control-Plane and User-Plane Separation BNG  
Simple Control Channel Protocol (S-CUSP)  
draft-cuspd-rtgwg-cu-separation-bng-protocol-06

## Abstract

This document specifies the Simple Control Plane (CP) and User Plane (UP) Separation Broadband Network Gateway (BNG) control channel Protocol (S-CUSP) for communications between a CP and a UP. S-CUSP is designed to be flexible and extensible so as to easily allow for the addition of further messages and data items to meet future requirements.

## Status of This Memo

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

Distribution of this document is unlimited. Comments should be sent to the authors or the RGTWG working group mailing list: [rtgwg@ietf.org](mailto:rtgwg@ietf.org).

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

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."

The list of current Internet-Drafts can be accessed at  
<http://www.ietf.org/lid-abstracts.html>. The list of Internet-Draft  
Shadow Directories can be accessed at  
<http://www.ietf.org/shadow.html>.

## Table of Contents

1. Introduction.....	6
2. Terminology.....	7
2.1 Implementation Requirement Keywords.....	7
2.2 Terms.....	7
3. BNG CUPS Overview.....	10
3.1 BNG CUPS Motivation.....	10
3.2 BNG CUPS Architecture Overview.....	10
3.3 BNG CUPS Interfaces.....	12
3.3.1 Service Interface.....	13
3.3.2 Control Interface.....	14
3.3.3 Management Interface.....	14
3.4 BNG CUPS Procedure Overview.....	14
4. S-CUSP Protocol Overview.....	18
4.1 Control Channel Related Procedures.....	18
4.1.1 S-CUSP Session Establishment.....	18
4.1.2 Keep Alive.....	19
4.2 Node Related Procedures.....	20
4.2.1 UP Resource Report.....	20
4.2.2 Update BAS Function on Access Interface.....	21
4.2.3 Update Network Routing.....	21
4.2.4 CGN Public IP Address Allocation.....	22
4.2.5 Data Synchronization between the CP and UP.....	23
4.3 Subscriber Session Related Procedures.....	24
4.3.1 Create Subscriber Session.....	25
4.3.2 Update Subscriber Session.....	26
4.3.3 Delete Subscriber Session.....	27
4.3.4 Subscriber Session Events Report.....	27
5. S-CUSP Call Flows.....	29
5.1 IPoE.....	29
5.1.1 DHCPv4 Access.....	29
5.1.2 DHCPv6 Access.....	30
5.1.3 IPv6 SLAAC Access.....	32
5.1.4 DHCPv6 + SLAAC Access.....	33
5.1.5 DHCP Dual Stack Access.....	35
5.1.6 L2 Static Subscriber Access.....	37
5.2 PPPoE.....	40
5.2.1 IPv4 PPPoE Access.....	40
5.2.2 IPv6 PPPoE Access.....	41
5.2.3 PPPoE Dual Stack Access.....	43
5.3 WLAN Access.....	45
5.4 L2TP.....	47
5.4.1 L2TP LAC Access.....	47
5.4.2 L2TP LNS IPv4 Access.....	49
5.4.3 L2TP LNS IPv6 Access.....	51
5.5 CGN (Carrier Grade NAT).....	54

## Table of Contents (continued)

5.6 L3 Leased Line Access.....	55
5.6.1 Web Authentication.....	55
5.6.2 User Traffic Trigger.....	57
5.7 Multicast Access.....	58
6. S-CUSP Message Formats.....	60
6.1 Common Message Header.....	60
6.2 Control Messages.....	61
6.2.1 Hello Message.....	61
6.2.2 Keepalive Message.....	62
6.2.3 Sync_Request Message.....	62
6.2.4 Sync_Begin Message.....	62
6.2.5 Sync_Data Message.....	63
6.2.6 Sync_End Message.....	63
6.2.7 Update_Request Message.....	64
6.2.8 Update_Response Message.....	64
6.3 Event Message.....	65
6.4 Report Message.....	66
6.5 CGN Messages.....	66
6.5.1 Addr_Allocation_Req Message.....	66
6.5.2 Addr_Allocation_Ack Message.....	66
6.5.3 Addr_Renew_Req Message.....	67
6.5.4 Addr_Renew_Ack Message.....	67
6.5.5 Addr_Release_Req Message.....	67
6.5.6 Addr_Release_Ack Message.....	67
6.6 Vendor Message.....	67
6.7 Error Message.....	68
7. S-CUSP TLVs and Sub-TLVs.....	69
7.1 Common TLV Header.....	69
7.2 Basic Data Fields.....	70
7.3 Sub-TLV Format and Sub-TLVs.....	71
7.3.1 Name sub-TLVs.....	71
7.3.2 Ingress-CAR sub-TLV.....	72
7.3.3 Egress-CAR sub-TLV.....	72
7.3.4 If-Desc sub-TLV.....	73
7.3.5 IPv6 Address List sub-TLV.....	75
7.3.6 Vendor sub-TLV.....	75
7.4 The Hello TLV.....	77
7.5 The Keep Alive TLV.....	78
7.6 The Error Information TLV.....	79
7.7 BAS Function TLV.....	79
7.8 Routing TLVs.....	82
7.8.1 IPv4 Routing TLV.....	82
7.8.2 IPv6 Routing TLV.....	84
7.9 Subscriber TLVs.....	85
7.9.1 Basic Subscriber TLV.....	86
7.9.2 PPP Subscriber TLV.....	88
7.9.3 IPv4 Subscriber TLV.....	89

## Table of Contents (continued)

7.9.4 IPv6 Subscriber TLV.....	90
7.9.5 IPv4 Static Subscriber Detect TLV.....	91
7.9.6 IPv6 Static Subscriber Detect TLV.....	93
7.9.7 L2TP-LAC Subscriber TLV.....	94
7.9.8 L2TP-LNS Subscriber TLV.....	95
7.9.9 L2TP-LAC Tunnel TLV.....	95
7.9.10 L2TP-LNS Tunnel TLV.....	96
7.9.11 Update Response TLV.....	97
7.9.12 Subscriber Policy TLV.....	98
7.9.13 Subscriber CGN Port Range TLV.....	100
7.10 Device Status TLVs.....	100
7.10.1 Interface Status TLV.....	101
7.10.2 Board Status TLV.....	101
7.11 CGN TLVs.....	102
7.11.1 Address Allocation Request TLV.....	102
7.11.2 Address Allocation Response TLV.....	103
7.11.3 Address Renewal Request TLV.....	104
7.11.4 The Address Renewal Response TLV.....	105
7.11.5 Address Release Request TLV.....	106
7.11.6 The Address Release Response TLV.....	106
7.12 Event TLVs.....	107
7.12.1. Subscriber Traffic Statistics TLV.....	108
7.12.2 Subscriber Detection Result TLV.....	109
7.13 Vendor TLV.....	110
8. Implementation Status.....	112
8.1 Implementations.....	112
8.1.1 Huawei Technologies.....	112
8.1.2 ZTE.....	113
8.1.3 H3C.....	113
8.2 Hackathon.....	113
8.3 EANTC Testing.....	114
9. IANA Considerations.....	115
9.1 Message Types.....	115
9.2 TLV Types.....	115
9.3 TLV Operation Codes.....	117
9.4 Sub-TLV Types.....	118
9.5 Error Codes.....	118
10. Security Considerations.....	120
Contributors.....	121
Normative References.....	122
Informative References.....	123
Authors' Addresses.....	125

## 1. Introduction

A fixed network Broadband Network Gateway (BNG) is an Ethernet-centric IP edge router, and the aggregation point for user traffic. To provide centralized session management, flexible address allocation, high scalability for subscriber management capacity, and cost-efficient redundancy, the Control/User (CU) separated BNG framework is described in [TR-384]. The CU separated service Control Plane (CP), which is responsible for user access authentication and setting forwarding entries in User Planes (UPs), can be virtualized and centralized. The routing control and forwarding plane, i.e. the BNG user plane (local), can be distributed across the infrastructure. Other structures can also be supported such as both CP and UP being virtual or both being physical.

This document specifies the Simple CU Separation BNG control channel Protocol (S-CUSP) for communications between a BNG Control Plane (CP) and a set of User Planes (UPs). S-CUSP is designed to be flexible and extensible so as to easily allow for additional messages and data items, should further requirements be expressed in the future.

## 2. Terminology

This section specifies implementation requirement keywords and terms used in this document. S-CUSP messages are described in this document using Routing Backus-Naur Form (RBNF) as defined in [RFC5511].

### 2.1 Implementation Requirement Keywords

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

### 2.2 Terms

This section specifies terms used in this document.

AAA: Authentication Authorization Accounting.

ACK: Acknowledgement message.

BAS: Broadband Access Server (BRAS, BNG).

BNG: Broadband Network Gateway. A broadband remote access server (BRAS (BRoadband Access Server), B-RAS or BBRAS) routes traffic to and from broadband remote access devices such as digital subscriber line access multiplexers (DSLAM) on an Internet Service Provider's (ISP) network. BRAS can also be referred to as a Broadband Network Gateway (BNG).

BRAS: BRoadband Access Server (BNG).

CAR: Committed Access Rate.

CBS: Committed Burst Size.

CGN: Carrier Grade NAT.

Ci: Control Interface.

CIR: Committed Information Rate.

CoA: Change of Authorization.



CP: Control Plane.

CP is a user control management component which supports the management of the UP's resources such as the user entry and forwarding policy.

CPE: Customer Premises Equipment.

CU: Control-plane / User-plane.

CUSP: Control and User plane Separation Protocol.

DEI: Drop Eligibility Indicator. A bit in a VLAN tag after the priority and before the VLAN ID. (This bit was formerly the CFI (Canonical Format Indicator).) [802.1Q]

DHCP: Dynamic Host Configuration Protocol [RFC2131].

dial-up: This refers to the initial connection messages when a new user appears. The name is left over from when users literally dialed up on a modem equipped phone line but herein is applied to other initial connection techniques. Initial connection is frequently indicated by the receipt of packets over PPPoE [RFC2516] or IPoE.

EMS: Element Management System.

IPoE: IP over Ethernet.

L2TP: Layer 2 Tunneling Protocol [RFC2661].

LAC: L2TP Access Concentrator.

LNS: L2TP Network Server.

MAC: 48-bit Media Access Control address [RFC7042].

MANO: Management and Orchestration.

Mi: Management Interface.

MSS: Maximum Segment Size.

MRU: Maximum Receive Unit.

NAT: Network Address Translation [RFC3022].

ND: Neighbor Discovery.

NFV: Network Function Virtualization.

NFVI: NFV Infrastructure

PBS: Peak Burst Size.

PD: Prefix Delegation.

PIR: Peak Information Rate.

PPP: Point to Point Protocol [RFC1661].

PPPoE: PPP over Ethernet [RFC2516].

RBNF: Routing Backus-Naur Form [RFC5511].

RG: Residential Gateway.

S-CUSP: Simple Control and User Plane Separation Protocol.

Si: Service Interface.

TLV: Type, Length, Value. See Sections 7.1 and 7.3.

UP: User Plane. UP is a network edge and user policy implementation component. The traditional router's Control Plane and Forwarding Plane are both preserved on BNG devices in the form of a user plane.

URPF: Unicast Reverse Path Forwarding.

User: Equivalent to "customer" or "subscriber".

VRF: Virtual Routing and Forwarding.

### 3. BNG CUPS Overview

#### 3.1 BNG CUPS Motivation

The rapid development of new services, such as 4K TV, IoT, etc., and increasing numbers of home broadband service users present some new challenges for BNGs such as:

**Low resource utilization:** The traditional BNG acts as both a gateway for user access authentication and accounting and an IP network's Layer 3 edge. The mutually affecting nature of the tightly coupled control plane and forwarding plane makes it difficult to achieve the maximum performance of either plane.

**Complex management and maintenance:** Due to the large numbers of traditional BNGs, configuring each device in a network is very tedious when deploying global service policies. As the network expands and new services are introduced, this deployment mode will cease to be feasible as it is unable to manage services effectively and rectify faults rapidly.

**Slow service provisioning:** The coupling of control plane and forwarding plane, in addition to a distributed network control mechanism, means that any new technology has to rely heavily on the existing network devices.

To address these challenges for fixed networks, the framework for a cloud-based BNG with Control Plane and User Plane (CU) separation is described in [TR-384]. The main idea of CU separation is to extract and centralize the user management functions of multiple BNG devices, forming a unified and centralized Control Plane (CP). And the traditional router's Control Plane and Forwarding Plane are both preserved on BNG devices in the form of a User Plane (UP).

#### 3.2 BNG CUPS Architecture Overview

The functions in a traditional BNG can be divided into two parts: one is the user access management function, the other is the router function. The user management function can be centralized and deployed as a concentrated module or device, called the BNG Control Plane (BNG-CP). The other functions, such as the router function and forwarding engine, can be deployed in the form of the BNG User Plane (BNG-UP).

The following figure shows the architecture of CU separated BNG:

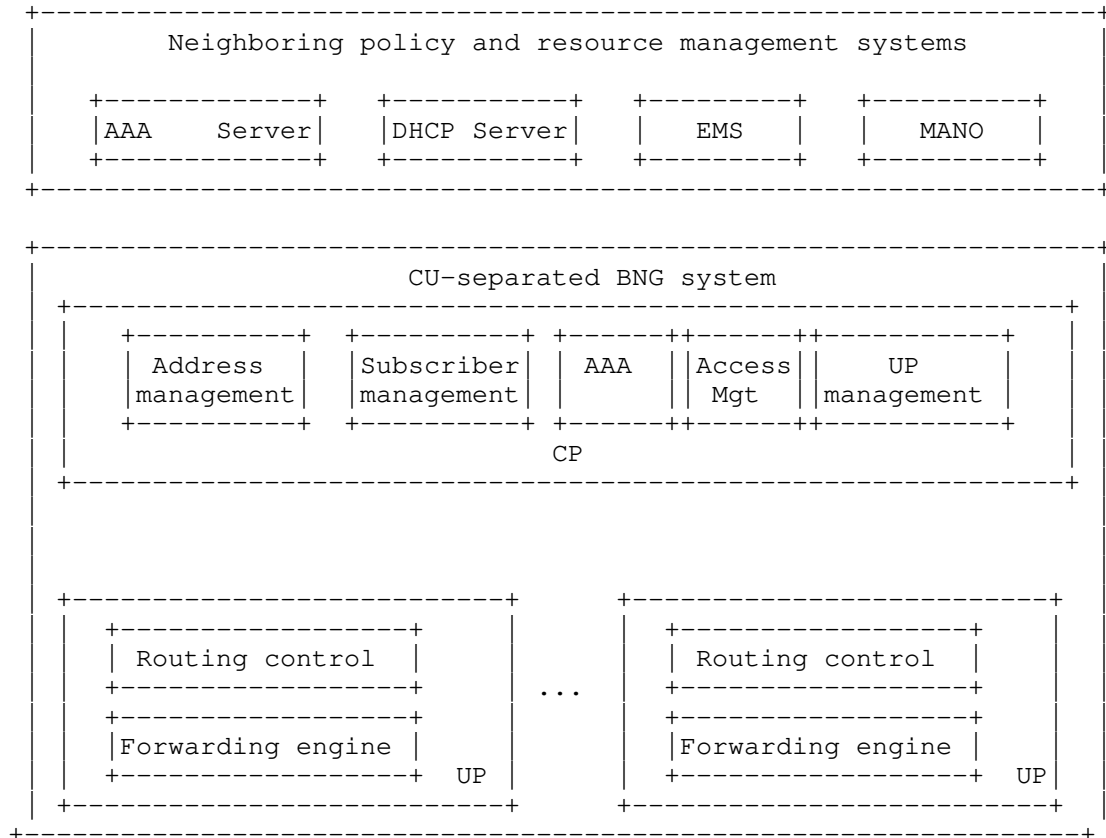


Figure 1: Architecture of CU Separated BNG

As shown in Figure 1, the BNG Control Plane could be virtualized and centralized, which provides benefits such as centralized session management, flexible address allocation, high scalability for subscriber management capacity, and cost-efficient redundancy, etc. The functional components inside the BNG Service Control Plane can be implemented as Virtual Network Functions (VNFs) and hosted in a Network Function Virtualization Infrastructure (NFVI).

The User Plane Management module in the BNG Control Plane centrally manages the distributed BNG User Planes (e.g. load balancing), as well as the setup, deletion, and maintenance of channels between Control Planes and User Planes. Other modules in the BNG control plane, such as address management, AAA, etc., are responsible for the connection with outside subsystems in order to fulfill those services. Note that the User Plane SHOULD support both physical and virtual network functions. For example, BNG user plane L3 forwarding

related network functions can be disaggregated and distributed across the physical infrastructure. And the other control plane and management plane functions in the CU Separation BNG can be moved into the NFVI for virtualization [TR-384].

The details of CU separated BNG's function components are as following:

The Control Plane is responsible for the following:

1. Address management: unified address pool management and CGN subscriber address traceability management.
2. AAA: This component performs Authentication, Authorization and Accounting, together with RADIUS/DIAMETER. The BNG communicates with the AAA server to check whether the subscriber who sent an Access-Request has network access authority. Once the subscriber goes online, this component together with the Service Control component implement accounting, data capacity limitation, and QoS enforcement policies.
3. Subscriber management: user entry management and forwarding policy management.
4. Access management: process user dial-up packets, such as PPPoE, DHCP, L2TP, etc.
5. UP management: management of UP interface status, and the setup, deletion, and maintenance of channels between CP and UP.

The User Plane is responsible for the following:

1. Routing control functions: responsible for constructing routing forwarding plane (e.g., routing, multicast, MPLS, etc.).
2. Routing and Service Forwarding plane functions: responsible including traffic forwarding, QoS and traffic statistics collection.

Subscriber detection: responsible for detecting whether a subscriber is still online.

### 3.3 BNG CUPS Interfaces

To support the communication between the Control Plane and User Plane, three interfaces are assumed. These are referred to as the Service Interface (Si), Control Interface (Ci), and Management Interface (Mi) as shown in Figure 2.

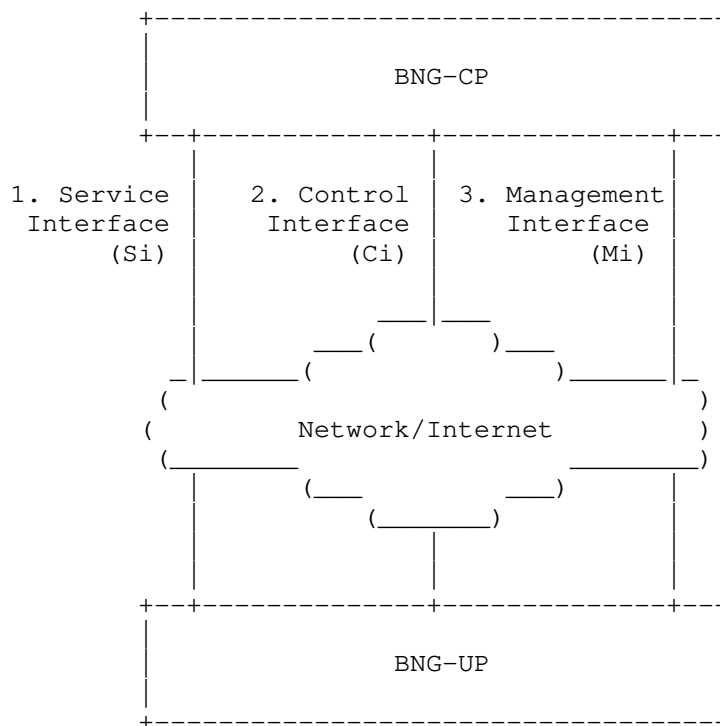


Figure 2: Interfaces Between the CP and UP of the BNG

### 3.3.1 Service Interface

For a traditional BNG (without CU separation), the user dial-up signals are terminated and processed by the control plane of a BNG. When the CP and UP of a BNG are separated, there needs to be a way to relay these signals between the CP and the UP.

The Service Interface (Si) is used to establish tunnels between the CP and UP. The tunnels are responsible for relaying the PPPoE, IPoE, and L2TP related control packets that are received from a Residential Gateway (RG) over those tunnels. An appropriate tunnel type is VXLAN [RFC7348].

The detailed definition of Si is out of scope for this document.

### 3.3.2 Control Interface

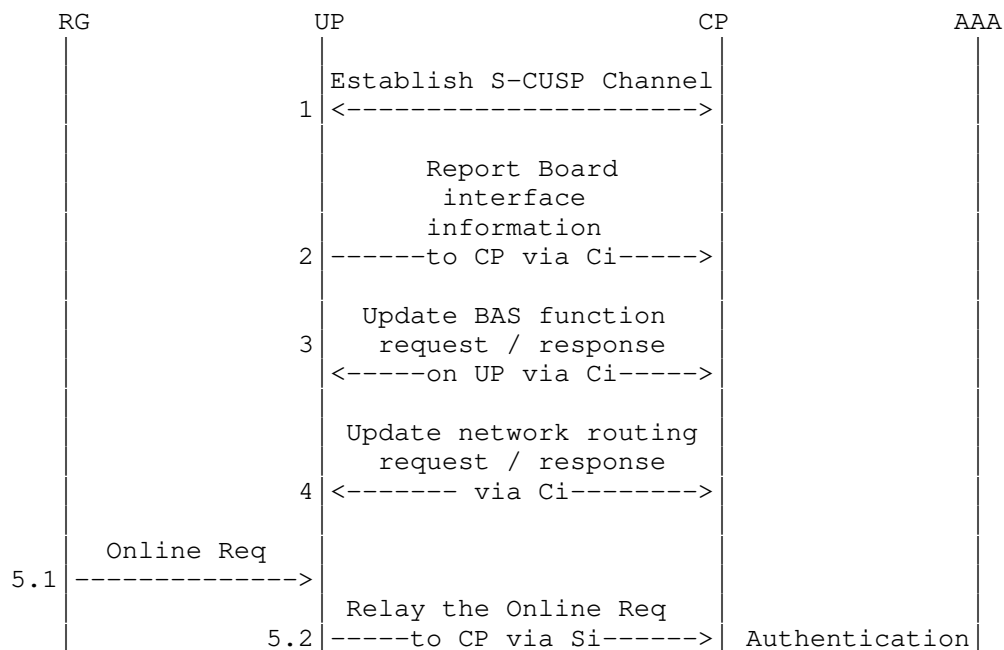
The CP uses the Control Interface to deliver subscriber session states, network routing entries, etc. to the UP (see Section 6.2.7)). The UP uses this interface to report subscriber service statistics, subscriber detection results, etc. to the CP (see Sections 6.3 and 6.4). A carrying protocol for this interface is specified in this document.

### 3.3.3 Management Interface

NETCONF [RFC6241] is the protocol used on the Management Interface between a CP and UP. It is used to configure the parameters of the Control Interface, Service Interface, the Access interfaces and QoS/ACL Templates. It is expected that implementations will make use of existing YANG models where possible, but that new YANG models specific to S-CUSP will need to be defined. The definitions of the parameters are out of scope for this document.

## 3.4 BNG CUPS Procedure Overview

The following numbered sequences (Figure 3) gives a high level view of the main BNG CUPS procedures.



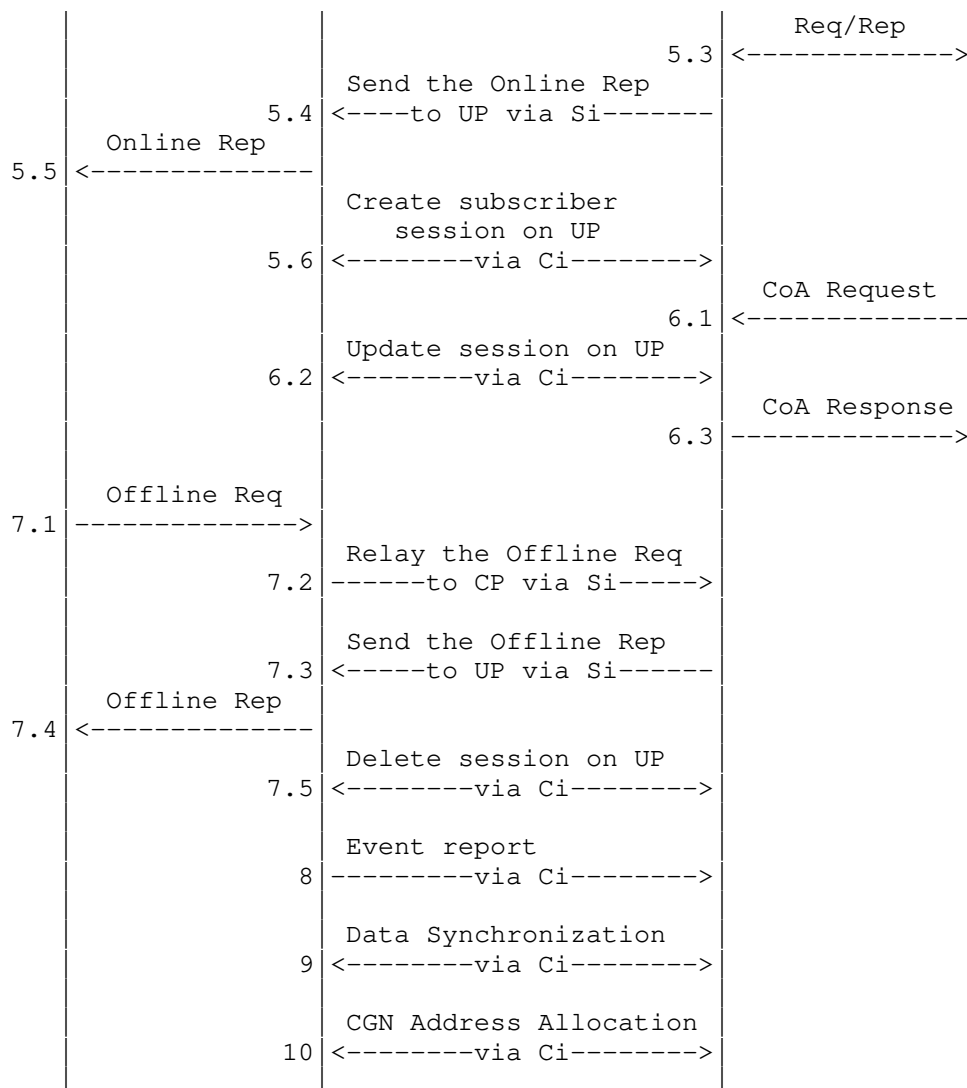


Figure 3: BNG CUPS Procedures Overview

1. S-CUSP session establishment: This is the first step of BNG CUPS procedures. Once the Control Interface parameters are configured on a UP. It will start to setup S-CUSP sessions with the specified CPs. The detailed definition of S-CUSP session establishment can be found in Section 4.1.1.
2. Board and interface report: Once the S-CUSP session is established between the UP and a CP, the UP will report status information on the boards and subscriber side interfaces of this UP to the CP. A board can also be called a Line/Service Process



Unit (LPU/SPU) card. The subscriber side interfaces refer to the interfaces that connect the Access Network nodes (e.g., OLT: Optical Line Terminal, DSLAM: Digital Subscriber Line Access Multiplexer, etc.). The CP can use this information to enable the Broadband Access Service (BAS) function (e.g., IPoE, PPPoE, etc.) on the specified interfaces. See Sections 4.2.1 and 7.10 for more details on Resource reporting.

3. BAS (Broadband Access Service) function enable: To enable the BAS function on the specified interfaces of a UP.
4. Subscriber network route advertisement: The CP will allocate one or more IP address blocks to a UP. Each address block contains a series of IP addresses. Those IP addresses will be allocated to subscribers who are dialing up from the UP. To enable other nodes in the network to learn how to reach the subscribers, the CP needs to notify the UP to advertise to the network the routes that can reach those IP addresses.
5. 5.1-5.6 is a complete call flow of a subscriber dial-up process. When a UP receives a dial-up request, it will relay the request packet to a CP through the Service Interface. The CP will parse the request. If everything is OK, it will send an authentication request to the AAA server to authenticate the subscriber. Once the subscriber passes the authentication, the AAA server will return a positive response to the CP. Then the CP will send the dial-up response packet to the UP and the UP will forward the response packet to the subscriber (RG). At the same time, the CP will create a subscriber session on the UP, which enables the subscriber to access the network. For different access types, the process may be a bit different. But the high-level process is similar. For each access type, the detail process can be found in Section 5.
6. 6.1-6.3 is the sequence when updating an existing subscriber session. The AAA server initiates a Change of Authorization (CoA) and sends the CoA to the CP. The CP will then update the session according to the CoA. See Section 4.3.2 for more detail on CP messages updating UP tables.
7. 7.1-7.5 is the sequence for deleting an existing subscriber session. When a UP receives an offline request, it will relay the request to a CP through the Service Interface. The CP will send back a response to the UP through the Service Interface. The UP will then forward the offline response to the subscriber. Then the CP will delete the session on the UP through the Control Interface.

8. Event reports include the following two parts (more detail can be found in Section 4.3.4) Both are reported using the Event message.
  - 8.1 Subscriber Traffic Statistics Report
  - 8.2 Subscriber Detection Result Report
9. Data synchronization: See Sections 4.2.5 for more detail on CP and UP Synchronization.
10. CGN address allocation: See Sections 4.2.4 for more detail on CGN Address Allocation.

## 4. S-CUSP Protocol Overview

### 4.1 Control Channel Related Procedures

#### 4.1.1 S-CUSP Session Establishment

A UP is associated with a CP and is controlled by that CP. In the case of a hot-standby or cold-standby, a UP is associated with two CPs, one called the Master CP and the other called the Standby CP. The association between a UP and its CPs is implemented by dynamic configuration.

Once a UP knows its CPs, the UP starts to establish S-CUSP sessions with those CPs as shown in Figure 4.

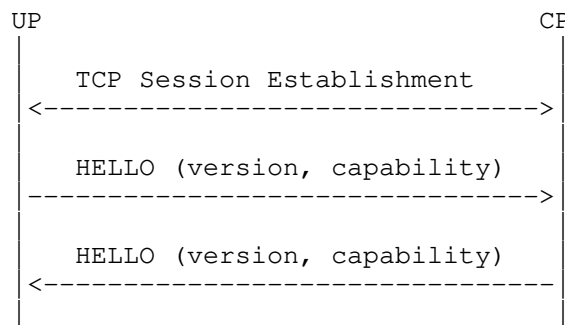


Figure 4: S-CUSP Session Establishment

The S-CUSP session establishment consists of two successive steps:

1. Establishment of a TCP [RFC793] connection (3-way handshake) between the CP and the UP using a configured port from the dynamic port range (49152-65535).
2. Establishment of a S-CUSP session over the TCP connection.

Once the TCP connection is established, the CP and the UP initialize the S-CUSP session during which the version and Keepalive timers are negotiated.

The version information (Hello TLV, see Section 7.4) is carried within Hello messages (see Section 6.2.1). A CP can support multiple versions, but a UP can only support one version. So, the version negotiation is based on whether a version can be supported by both the CP and the UP. For a CP or UP, if a Hello message is received that

does not indicate a version supported by both, a subsequent Hello message with an Error Information TLV will be sent to the peer to notify the peer of the "Version-Mismatch" error and the session establishment phase fails.

Keepalive negotiation is performed by carrying a Keepalive TLV in the Hello message. The Keepalive TLV includes a Keepalive timer and Dead Timer field. The CP and UP have to agree on the Keepalive Timer and Dead Timer. Otherwise, a subsequent Hello message with an Error Information TLV will be sent to its peer and the session establishment phase fails.

The S-CUSP session establishment phase fails if the CP or UP disagree on the version and keepalive parameters or if one of the CP or UP does not answer after the expiration of the Establishment timer. When the S-CUSP session establishment fails, the TCP connection is promptly closed. Successive retries are permitted but an implementation SHOULD make use of an exponential back-off session establishment retry procedure.

The S-CUSP session timer values that need to be configured are summarized in the table below.

Timer Name	Range in seconds	Default Value
Establishment	1-32767	45
Keepalive	0-255	30
DeadTimer	1-32767	4 * Keepalive

#### 4.1.2 Keep Alive

Once an S-CUSP session has been established, a UP or CP may want to know that its S-CUSP peer is still available for use.

Each end of a S-CUSP session runs a Keepalive timer. It restarts the timer every time it sends a message on the session. When the timer expires, it sends a Keepalive message.

The ends of the S-CUSP session also run DeadTimers, and they restart the timers whenever a message is received on the session. If one end of the session receives no message after the DeadTimer expires, it declares the session dead. The session will be closed.

The minimum value of the Keepalive timer is 1 second, and it is specified in units of 1 second. The RECOMMENDED default value is 30 seconds. The timer may be disabled by setting it to zero.

The recommended default for the DeadTimer is 4 times the value of the Keepalive timer used by the remote peer. This implies there is essentially no risk of TCP congestion due to excessive Keepalive messages.

The Keepalive timer and DeadTimer are initially negotiated through the Keepalive TLV carried in the Hello Message.

## 4.2 Node Related Procedures

### 4.2.1 UP Resource Report

Once an S-CUSP session has been established between a CP and an UP. The UP reports the information of the Boards and access side interfaces on this UP to the CP as shown in Figure 5. Report messages are unacknowledged and are assumed to be delivered because the session runs over TCP.

The CP can use that information to activate/enable the Broadband Access Service (BAS) functions (e.g., IPoE, PPPoE, etc.) on the specified interfaces.

In addition, the UP resource report may trigger a UP warm-standby process. In the case of warm-standby, a failure on an UP may trigger the CP to start a warm-standby process, by moving the on-line subscriber sessions to a standby UP and then direct the affected subscribers to access the Internet through the standby UP.

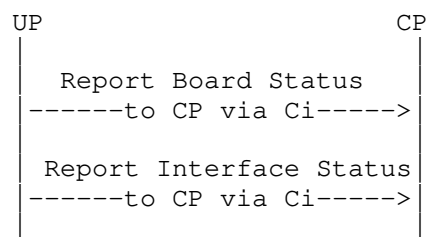


Figure 5: UP Board and Interface Report

Board status information is carried in the Board Status TLV (Section 7.10.2) and Interface status information is carried in Interface Status TLV (Section 7.10.1). Both Board and Interface Status TLVs are carried in the Report Message (Section 6.4).

#### 4.2.2 Update BAS Function on Access Interface

Once the CP collects the interface status of a UP, it will activate/de-activate/modify the BAS functions on specified interfaces through the Update\_Request and Update\_Response message (Section 6.2) exchanges carrying the BAS Function TLV (Section 7.7).

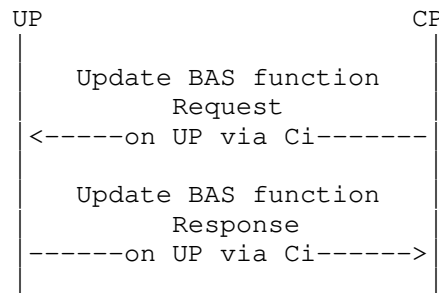


Figure 6: Update BAS Function

#### 4.2.3 Update Network Routing

The CP will allocate one or more address blocks to a UP. Each address block contains a series of IP addresses. Those IP addresses will be allocated to subscribers who are dialing up to the UP. To enable the other nodes in the network to learn how to reach the subscribers, the CP needs to install the routes on the UP and notify the UP to advertise the routes to the network.

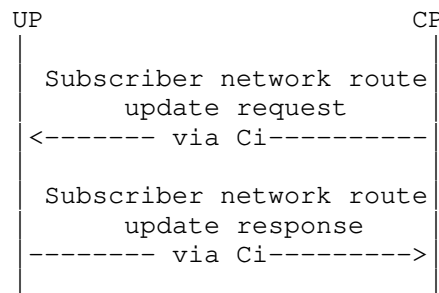


Figure 7: Update Network Routing

The subscriber network routing update request and response are achieved through the Update Request and Response Message exchanges by carrying the IPv4/IPv6 Routing Information TLVs (Section 7.8).

#### 4.2.4 CGN Public IP Address Allocation

The following sequences describe the CGN address management related procedures. Three independent procedures are defined, one each for CGN address allocation request/response, CGN address renewal request/response, and CGN address release request/response.

CGN address allocation/renew/release procedures are designed for the case where the CGN function is running on the UP. The UP has to map the subscriber private IP addresses to a public IP addresses, and such mapping is performed by the UP locally when a subscriber dials-up. That means the UP has to ask for public IPv4 address blocks for CGN subscribers from the CP.

In addition, when a public IP address is allocated to a UP, there will be a lease time (e.g., one day). Before the lease time expires, the UP can ask for renewal of the IP address lease from the CP. It is achieved by the exchange of the Addr\_Renew\_Req and Addr\_Renew\_Ack messages.

If the public IP address will not be used anymore, the UP SHOULD release the address by sending an Addr\_Release\_Req message to the CP.

If the CP wishes to withdraw addresses that it has previously leased to a UP, it uses the same procedures as above. The "Oper" code in the IPv4/IPv6 Routing TLV (see Section 7.1) determines whether the request is an update or withdraw.

The relevant messages are defined in Section 6.5.

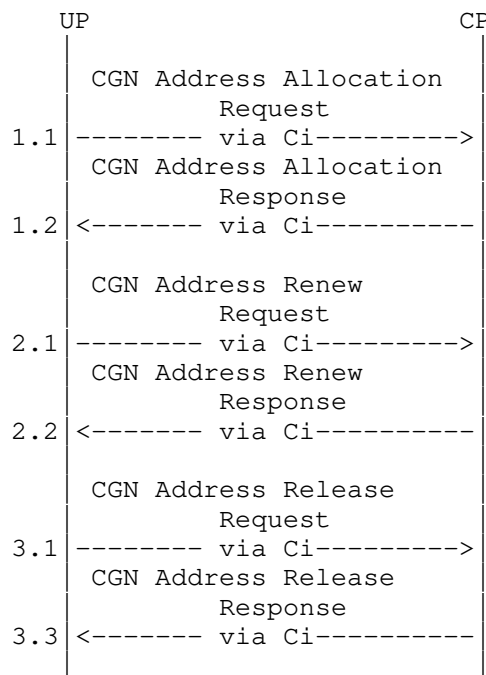


Figure 8: CGN Public IP Address Allocation

#### 4.2.5 Data Synchronization between the CP and UP

For a CU separated BNG, the UP will continue to function using the state that has been installed in it even if the CP fails or the session between the UP and CP fails.

Under some circumstances it is necessary to synchronize state between the CP and UP, for example if a CP fails and the UP is switched to a different CP.

Synchronization includes two directions. One direction is from UP to CP; in that case, the synchronization information is mainly about the board/interface status of the UP. The other direction is from CP to UP; in that case, the subscriber sessions, subscriber network routes, L2TP tunnels, etc. will be synchronized to the UP.

The synchronization is triggered by a Sync\_Request message, to which the receiver will (1) reply with a Sync\_Begin message to notify the requester that synchronization will begin, and (2) then start the synchronization using the Sync\_Data message. When synchronization finished, a Sync\_End message will be sent.



The following figure shows the process of data synchronization between a UP and a CP.

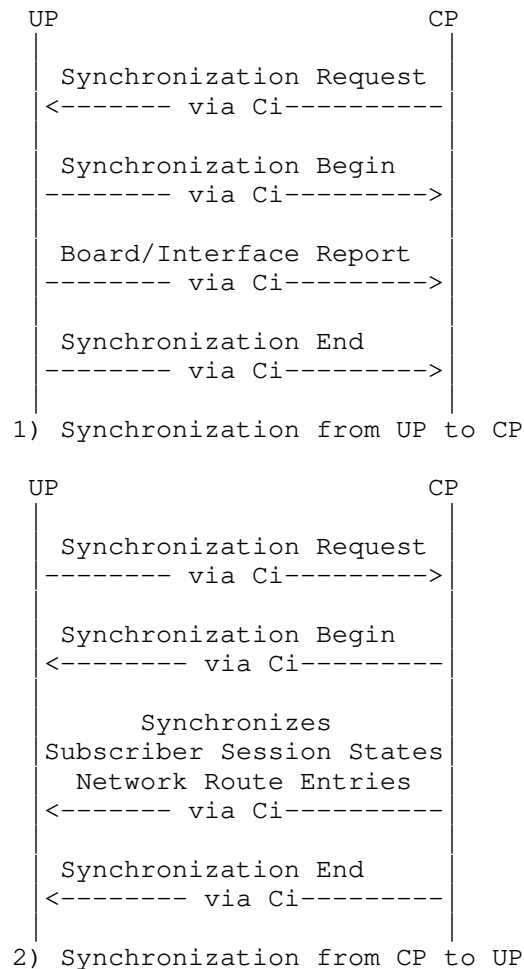


Figure 9: Data Synchronization

#### 4.3 Subscriber Session Related Procedures

A subscriber session consists of a set of forwarding states, policies, and security rules that are applied to the subscriber. It is used for forwarding subscriber traffic in a UP. To initialize a session on a UP, a set of hardware resource have to be allocated (e.g., NP, TCAM etc.) to a session.

Subscriber session related procedures include subscriber session

create, update, delete, and statistics report. The following sub-sections give a high level view of the procedures.

#### 4.3.1 Create Subscriber Session

The below sequence describes the DHCP IPv4 dial-up process, it is an example that shows how a subscriber session is created. (An example for IPv6 appears in Section 5.1.2.)

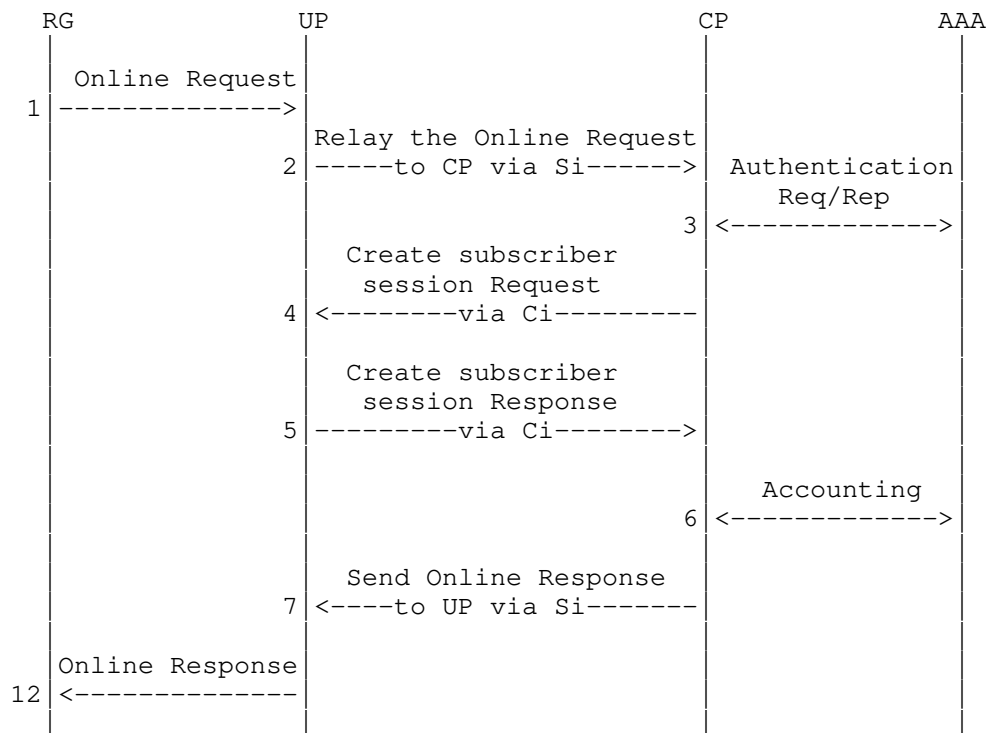


Figure 10: Subscriber Session Create

The request starts from an Online Request message (step 1) from the RG (for example, a DHCP Discovery packet). When the UP receives the Online Request from the RG, it will tunnel the Online Request to the CP through the Service Interface (Step 2). The Service Interface is implemented by a tunneling technology.

When the CP receives the Online Request from the UP, it will send an authentication request to the AAA server to authenticate and authorize the subscriber (step 3). When a positive reply is received from the AAA sever, the CP starts to create a subscriber session for the request. Relevant resources (e.g., IP address, bandwidth, etc.)

will be allocated to the subscriber, policies and security rules will be generated for the subscriber. Then the CP sends a session create request to the UP through the Control Interface (Ci) (step 4), and a response is expected from the UP to confirm the creation (step 5).

Finally, the CP will notify the AAA server to start accounting (step 6). At the same time, an Online Response message (for example, a DHCP Ack packet) will be sent to the UP through the Si (step 7). And the UP will forward the Online Response to the RG (step 8).

This completes the subscriber online process.

#### 4.3.2 Update Subscriber Session

The following numbered sequence shows the process of subscriber session update.

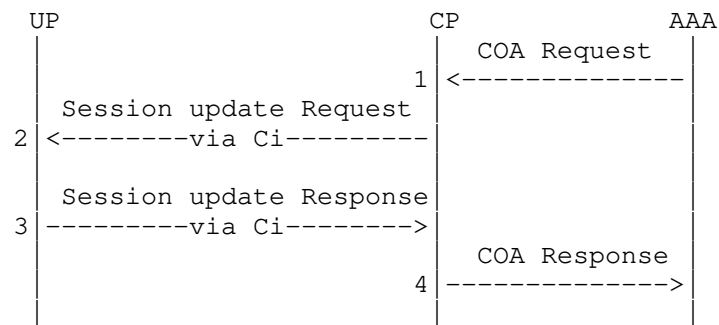


Figure 11: Subscriber Session Update

When a subscriber session has been created on a UP, there may be requirements to update the session with new parameters (e.g., Bandwidth, QoS, policies, etc.).

This procedure is triggered by a Change of Authorization (COA) request message sent by the AAA server. The CP will update the session on the UP according to the new parameters through the Control Interface.

#### 4.3.3 Delete Subscriber Session

The below call flow shows generally how S-CUPS deals with a subscriber offline request.

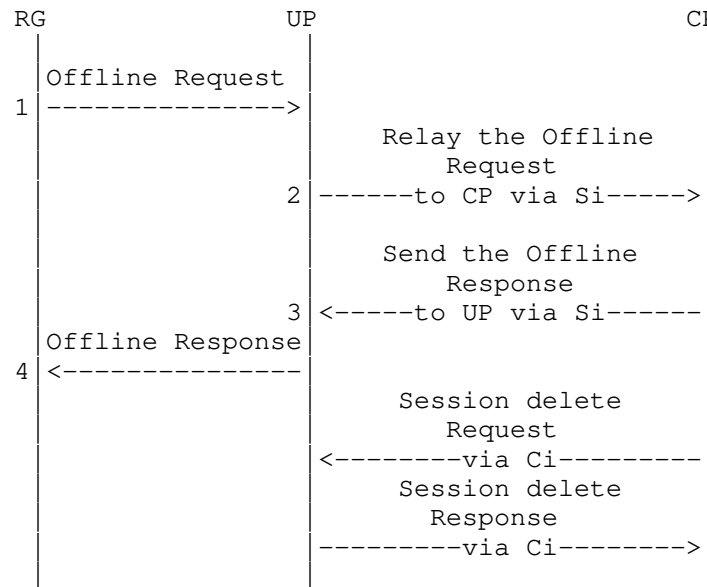


Figure 12: Subscriber Session Delete

Similar to the session creation process, when a UP receives an offline request from a RG, it will tunnel the request to a CP through the Si.

When the CP receives the offline request, it will withdraw/release the resources (e.g., IP address, bandwidth) that have been allocated to the subscriber. Then, it sends a reply to the UP through the Service Interface and the UP will forward the reply to the RG. At the same time, it will delete all the status of the session on the UP through the Ci.

#### 4.3.4 Subscriber Session Events Report

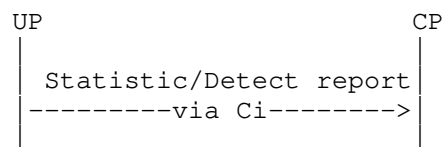


Figure 13: Events Report

When a session is created on an UP, the UP will periodically report statistics information and detect results of the session to the CP.

## 5. S-CUSP Call Flows

The subsections below give an overview of various "dial-up" interactions over the Service Interface followed by an overview of the setting of various information in the UP by the CP using S-CUSP over the Control Interface.

S-CUSP messages are described in this document using Routing Backus Naur Form (RBNF) as defined in [RFC5511].

### 5.1 IPoE

#### 5.1.1 DHCPv4 Access

The following sequence shows detailed procedures for DHCPv4 access.

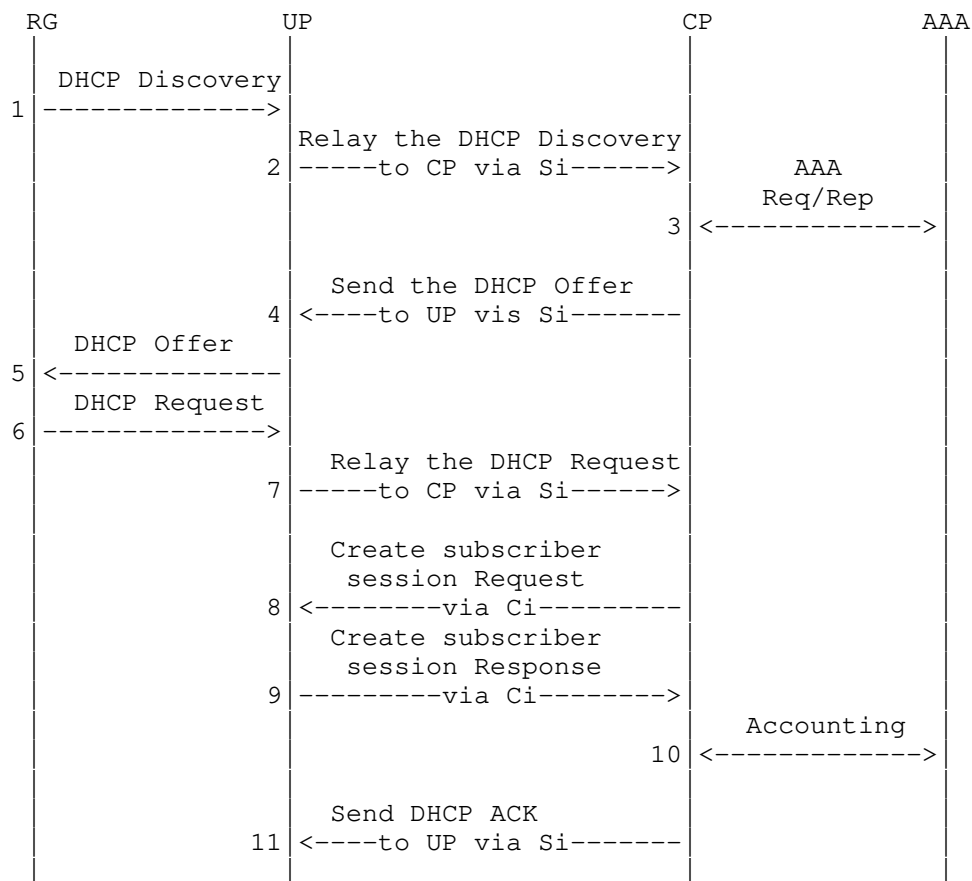




Figure 14: DHCPv4 Access

Step 8 and 9 are implemented by the S-CUSP protocol.

When a subscriber is authenticated and authorized by the AAA server, the CP will create a subscriber session on the UP. This is achieved by sending an Update\_Request message to the UP.

The format of the Update\_Request message is shown as follows using RBNF:

```

<Update_Request Message> ::= <Common Header>
                             <Basic Subscriber TLV>
                             <IPv4 Subscriber TLV>
                             <IPv4 Routing TLV>
                             [<Subscriber Policy TLV>]
  
```

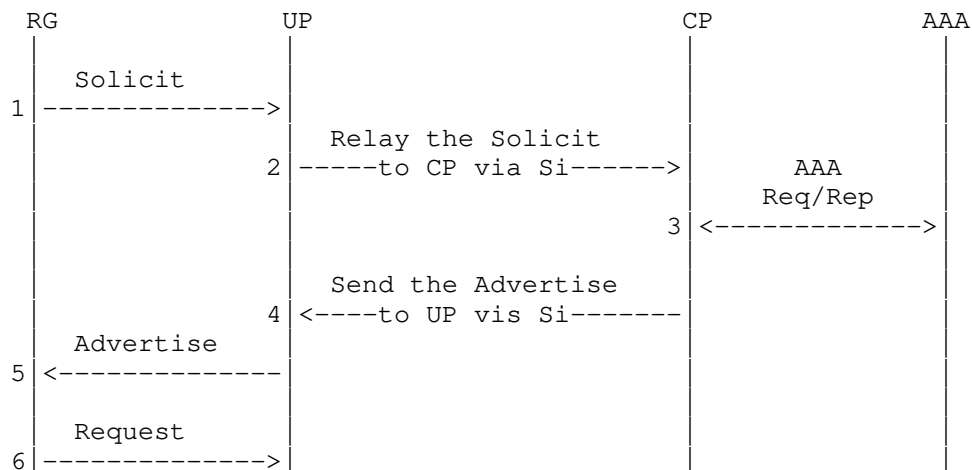
The UP will reply with an Update\_Response message, the format of the Update\_Response message is as follows:

```

<Update_Response Message> ::= <Common Header>
                              <Update Response TLV>
                              [<Subscriber CGN Port Range TLV>]
  
```

### 5.1.2 DHCPv6 Access

The following sequence shows detailed procedures for DHCPv6 access.



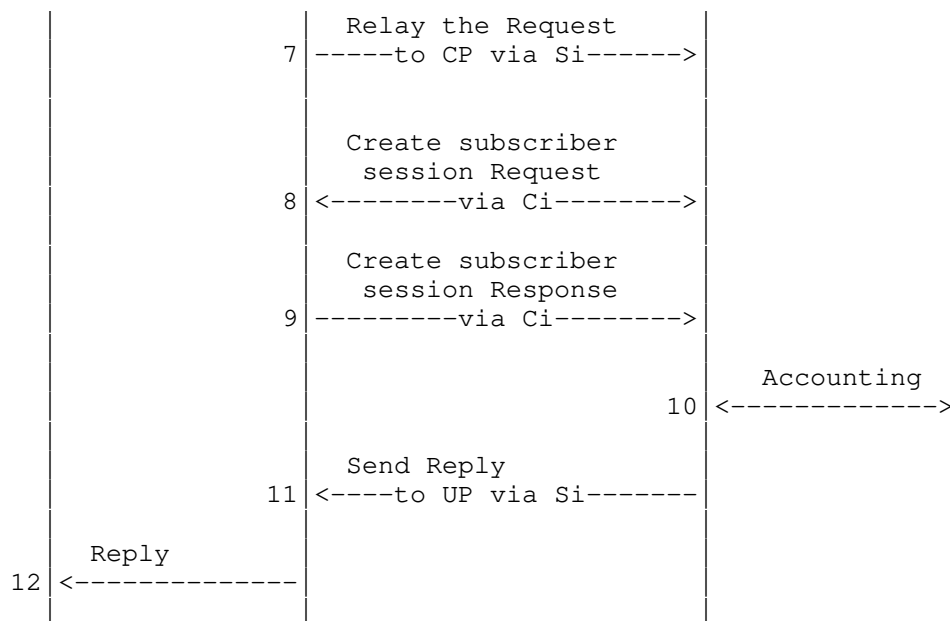


Figure 15: DHCPv6 Access

Steps 1-7 are a standard DHCP IPv6 access process. The subscriber creation is triggered by a DHCP IPv6 request message. When this message is received, it means that the subscriber has passed the AAA authentication and authorization. Then the CP will create a subscriber session on the UP. This is achieved by sending an Update\_Request message to the UP (Step 8).

The format of the Update\_Request message is as follows:

```

<Update_Request Message> ::= <Common Header>
                             <Basic Subscriber TLV>
                             <IPv6 Subscriber TLV>
                             <IPv6 Routing TLV>
                             [<Subscriber Policy TLV>]
  
```

The UP will reply with an Update\_Response message (Step 9). The format of the Update\_Response message is as follows:

```

<Update_Response Message> ::= <Common Header>
                             <Update Response TLV>
  
```



## 5.1.3 IPv6 SLAAC Access

The following flow shows the IPv6 SLAAC access process.

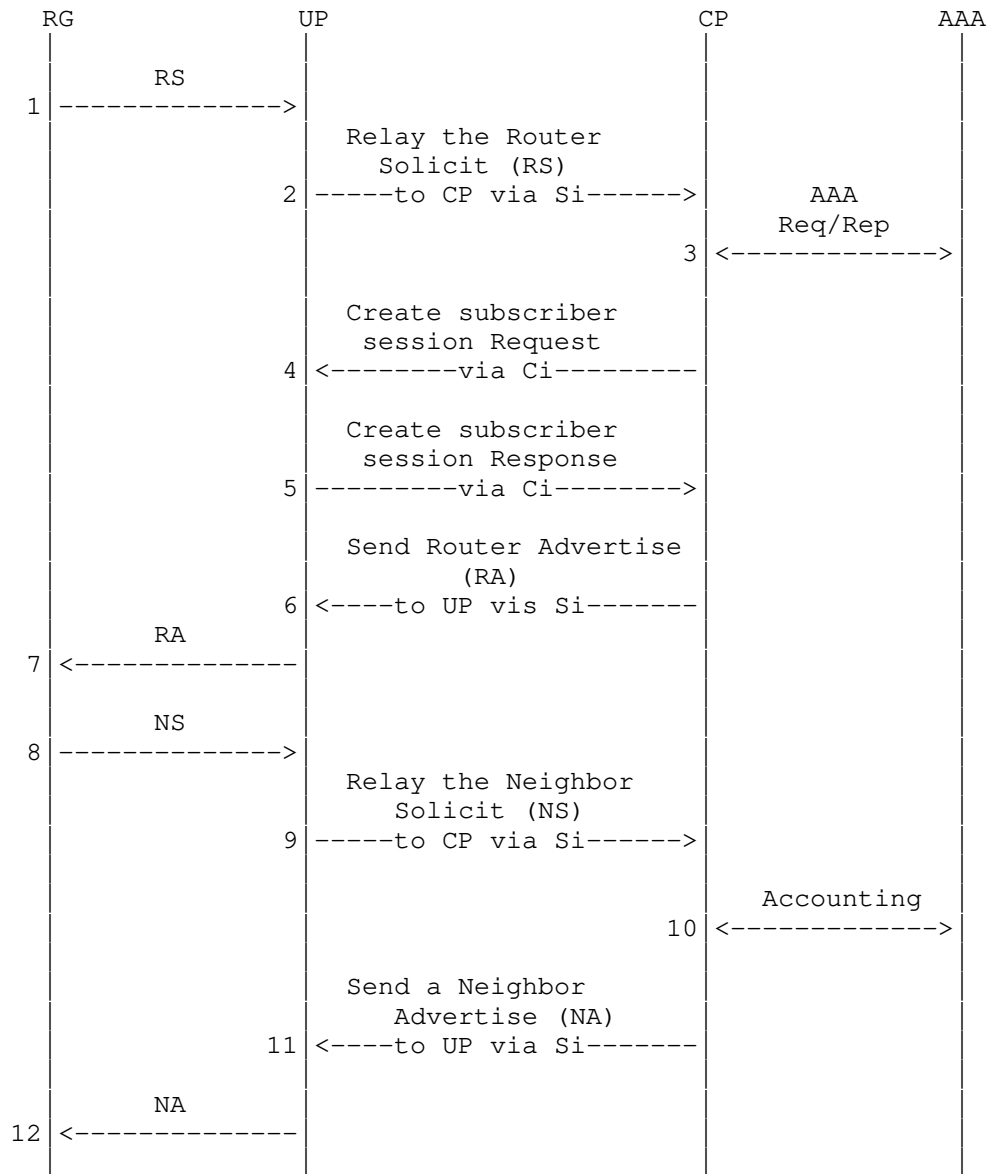


Figure 16: IPv6 SLAAC Access

It starts with a Router Solicit (RS) request from an RG that is tunneled to the CP by the UP. After the AAA authentication and authorization, the CP will create a subscriber session on the UP.

This is achieved by sending an Update\_Request message to the UP (step 4).

The format of the Update\_Request message is as follows:

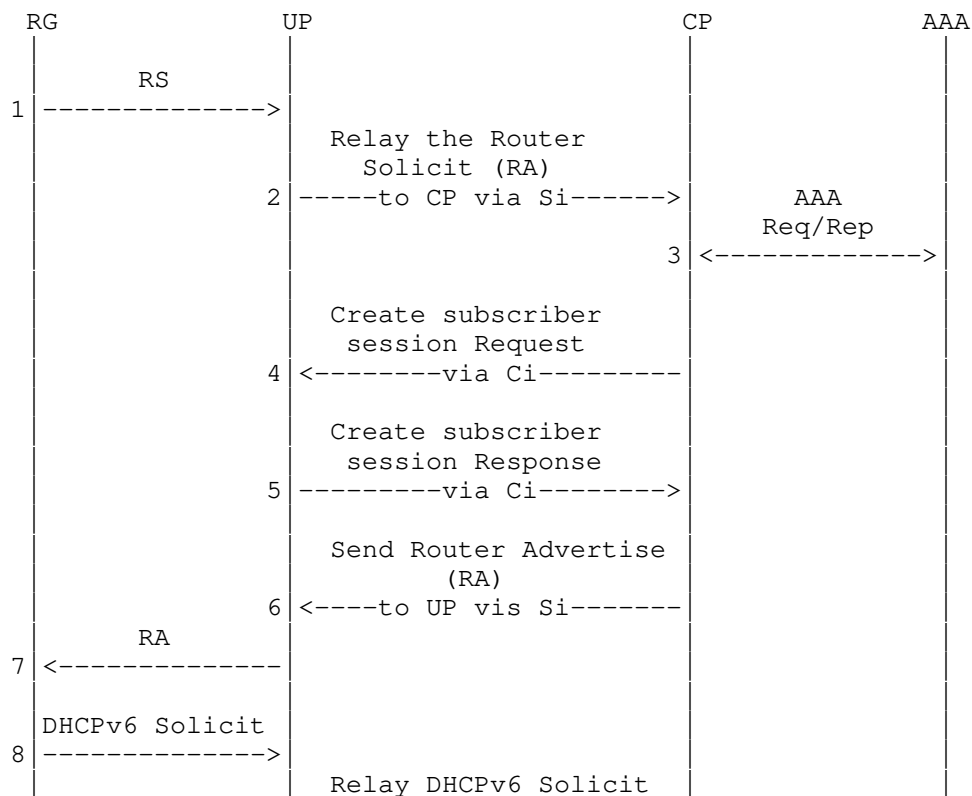
```
<Update_Request Message> ::= <Common Header>
                               <Basic Subscriber TLV>
                               <IPv6 Subscriber TLV>
                               <IPv6 Routing TLV>
                               [<Subscriber Policy TLV>]
```

The UP will reply with an Update\_Response message (step 5), the format of the Update\_Response message is as follows:

```
<Update_Response Message> ::= <Common Header>
                               <Update Response TLV>
```

#### 5.1.4 DHCPv6 + SLAAC Access

The following call flow shows the DHCP IPv6 and SLAAC access process.



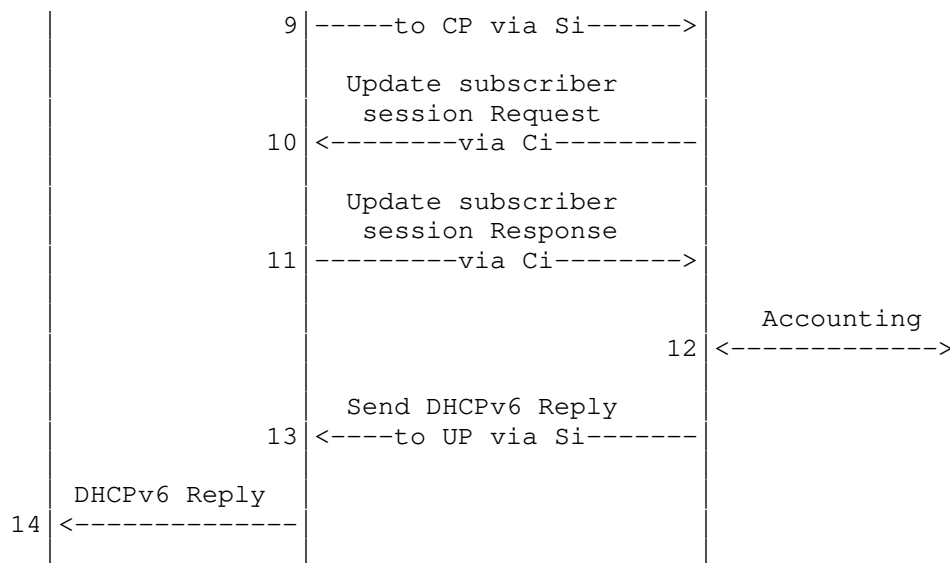


Figure 17: DHCPv6 + SLAAC Access

When a subscriber passes AAA authentication, the CP will create a subscriber session on the UP. This is achieved by sending an Update\_Request message to the UP (step 4).

```

<Update_Request Message> ::= <Common Header>
                             <Basic Subscriber TLV>
                             <IPv6 Subscriber TLV>
                             <IPv6 Routing TLV>
                             [<Subscriber Policy TLV>]
  
```

The UP will reply with an Update\_Response message (step 5). The format of the Update\_Response is as follows:

```

<Update_Response Message> ::= <Common Header>
                              <Update Response TLV>
  
```

After receiving a DHCPv6 Solicit, the CP will update the subscriber session by sending an Update\_Request message with new parameters to the UP (Step 10).

The format of the Update\_Request message is as follows:

```

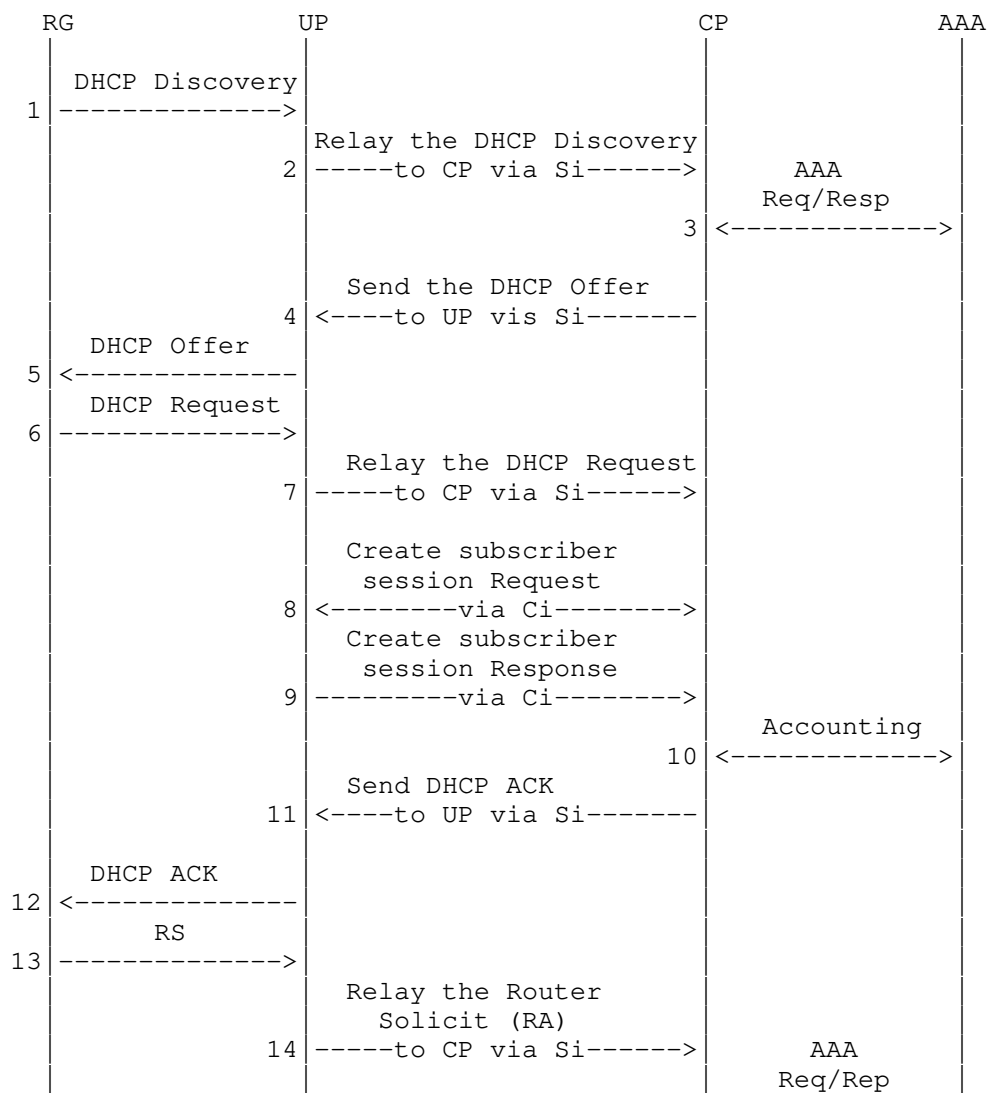
<Update_Request Message> ::= <Common Header>
                             <Basic Subscriber TLV>
                             <IPv6 Subscriber TLV>
                             <IPv6 Routing TLV>
                             [<Subscriber Policy TLV>]
  
```

The UP will reply with an Update\_Response message (step 11). The format of the Update\_Response is as follows:

```
<Update_Response Message> ::= <Common Header>
                               <Update Response TLV>
```

#### 5.1.5 DHCP Dual Stack Access

The following sequence is a combination of DHCP IPv4 and DHCP IPv6 access processes.



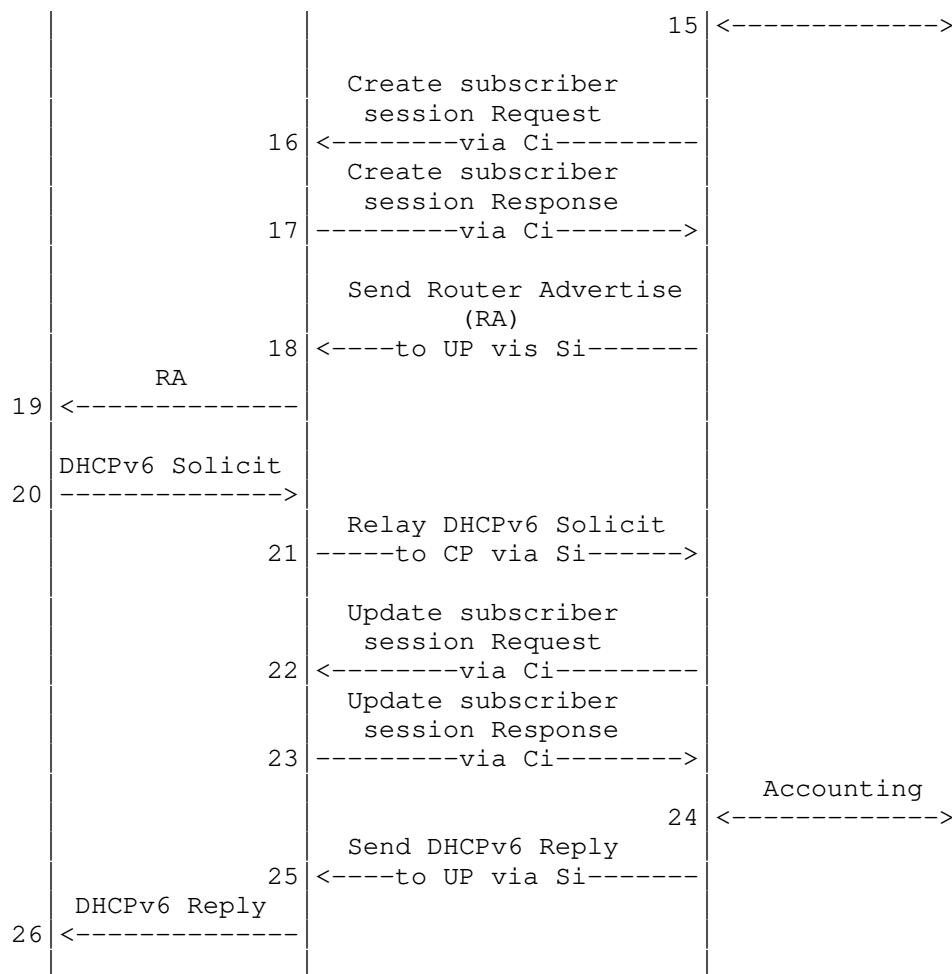


Figure 18: DHCP Dual Stack Access

The DHCP dual stack access includes three sets of Update\_Request / Update\_Response exchanges to create/update DHCPv4/v6 subscriber session.

1. Create DHCPv4 session (step 8 and 9)

```

<Update_Request Message> ::= <Common Header>
                             <Basic Subscriber TLV>
                             <IPv4 Subscriber TLV>
                             <IPv4 Routing TLV>
                             [<Subscriber Policy TLV>]
  
```

```

<Update_Response Message> ::= <Common Header>
                               <Update Response TLV>
                               [<Subscriber CGN Port Range TLV>]

```

## 2. Create DHCPv6 session (step 16 and 17)

```

<Update_Request Message> ::= <Common Header>
                               <Basic Subscriber TLV>
                               <IPv6 Subscriber TLV>
                               <IPv6 Routing TLV>
                               [<Subscriber Policy TLV>]

```

```

<Update_Response Message> ::= <Common Header>
                               <Update Response TLV>

```

## 3. Update DHCPv6 session (step 22 and 23)

```

<Update_Request Message> ::= <Common Header>
                               <Basic Subscriber TLV>
                               <IPv6 Subscriber TLV>
                               <IPv6 Routing TLV>
                               [<Subscriber Policy TLV>]

```

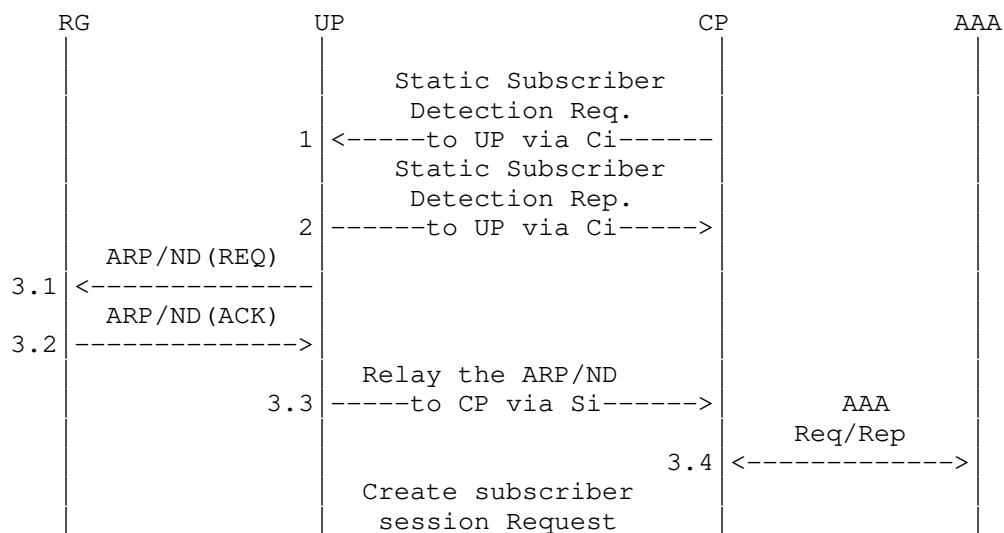
```

<Update_Response Message> ::= <Common Header>
                               <Update Response TLV>

```

### 5.1.6 L2 Static Subscriber Access

L2 static subscriber access processes are as follows:



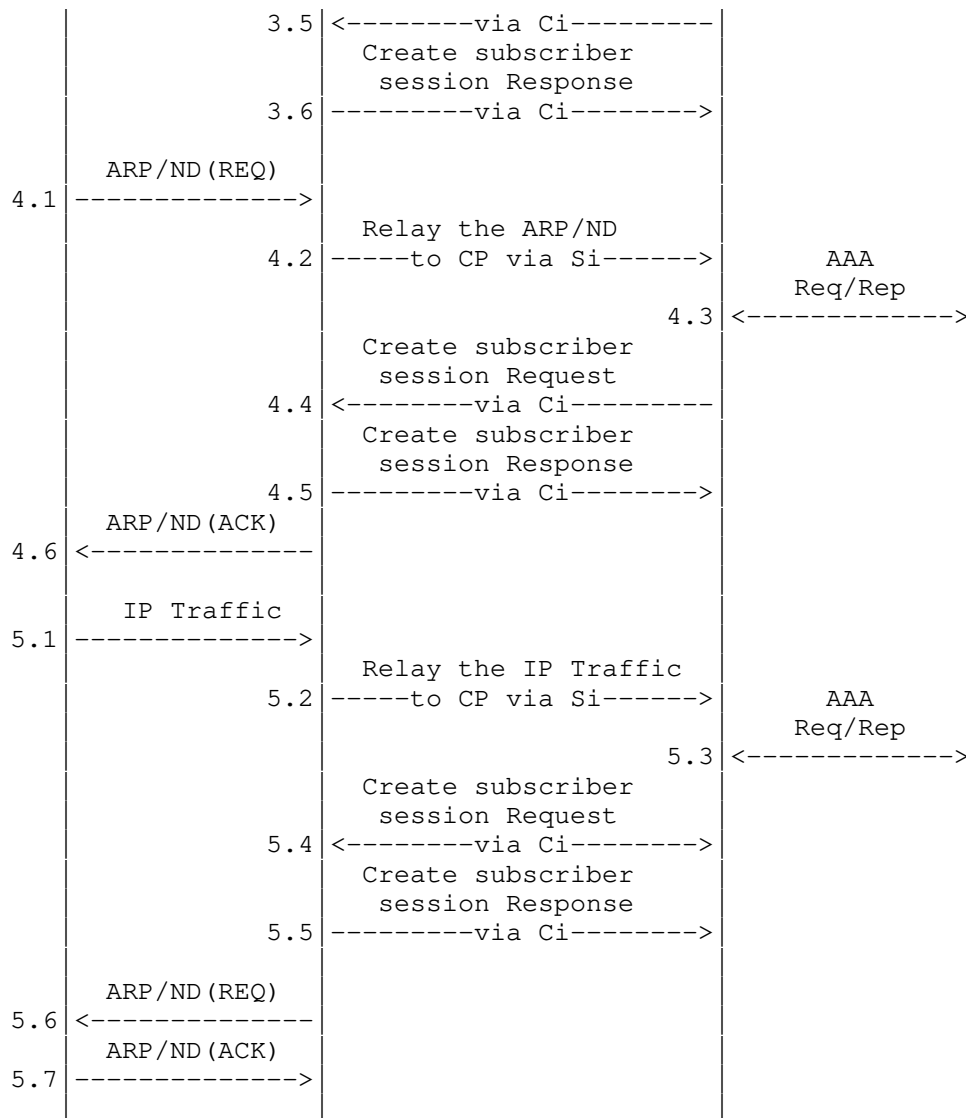


Figure 19: L2 Static Subscriber Access

For L2 static subscriber access, the process starts with a CP installing a static subscriber detection list on an UP. The list determines which subscribers will be detected. This is implemented by exchanging Update\_Request and Update\_Response messages between CP and UP. The format of the messages are as follows:

```

<Update_Request Message> ::= <Common Header>
                               <IPv4 Static Subscriber Detect TLVs>
                               <IPv6 Static Subscriber Detect TLVs>

```

```

<Update_Response Message> ::= <Common Header>
                               <Update Response TLV>

```

For L2 Static subscriber access, there are three ways to trigger the access process:

1. Triggered by UP (3.1-3.6): This assumes that the UP knows the IP address, the access interface, and VLAN of the RG. The UP will actively trigger the access flow by sending an ARP/ND packet to the RG. If the RG is online, it will reply with an ARP/ND to the UP. The UP will tunnel the ARP/ND to the CP through the Si. The CP then triggers the authentication process. If the authentication result is positive. The CP will create a corresponding subscriber session on the UP.
2. Triggered by RG ARP/ND (4.1-4.6): Most of the process is same as option 1 (triggered by UP). The difference is that the RG will actively send the ARP/ND to trigger the process.
3. Triggered by RG IP traffic (5.1-5.7): This is for the case where the RG has the ARP/ND information, but the subscriber session on the UP is lost (e.g., due to failure on the UP, or the UP restarted). That means the RG may keep sending IP packets to the UP. The packets will trigger the UP to start a new access process.

From a subscriber session point of view, the procedures and the message formats for the above three cases are the same, as follows:

IPv4 Case:

```

<Update_Request Message> ::= <Common Header>
                               <Basic Subscriber TLV>
                               <IPv4 Subscriber TLV>
                               <IPv4 Routing TLV>
                               [<Subscriber Policy TLV>]

<Update_Response Message> ::= <Common Header>
                               <Update Response TLV>
                               [<Subscriber CGN Port Range TLV>]

```

IPv6 Case:

```

<Update_Request Message> ::= <Common Header>
                               <Basic Subscriber TLV>
                               <IPv6 Subscriber TLV>
                               <IPv6 Routing TLV>
                               [<Subscriber Policy TLV>]

```



<Update\_Response Message> ::= <Common Header>  
                                   <Update Response TLV>

## 5.2 PPPoE

### 5.2.1 IPv4 PPPoE Access

The following figure shows the IPv4 PPPoE access call flow.

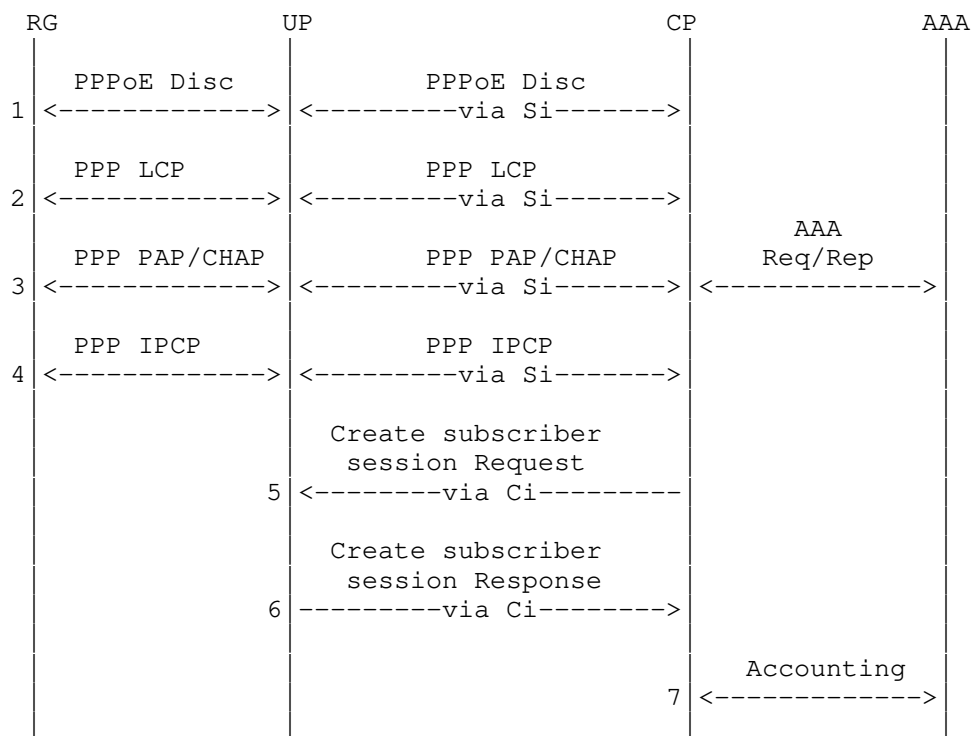


Figure 20: IPv4 PPPoE Access

From the above sequence, step 1-4 are the standard PPPoE call flow. The UP is responsible for redirecting the PPPoE control packets to the CP or RG. The PPPoE control packets are transmitted between the CP and UP through the Si.

After the PPPoE call flow, if the subscriber passed the AAA authentication and authorization, the CP will create a corresponding session on the UP through the Ci. The formats of the messages are as follows:

```

<Update_Request Message> ::= <Common Header>
                               <Basic Subscriber TLV>
                               <PPP Subscriber TLV>
                               <IPv4 Subscriber TLV>
                               <IPv4 Routing TLV>
                               [<Subscriber Policy TLV>]

```

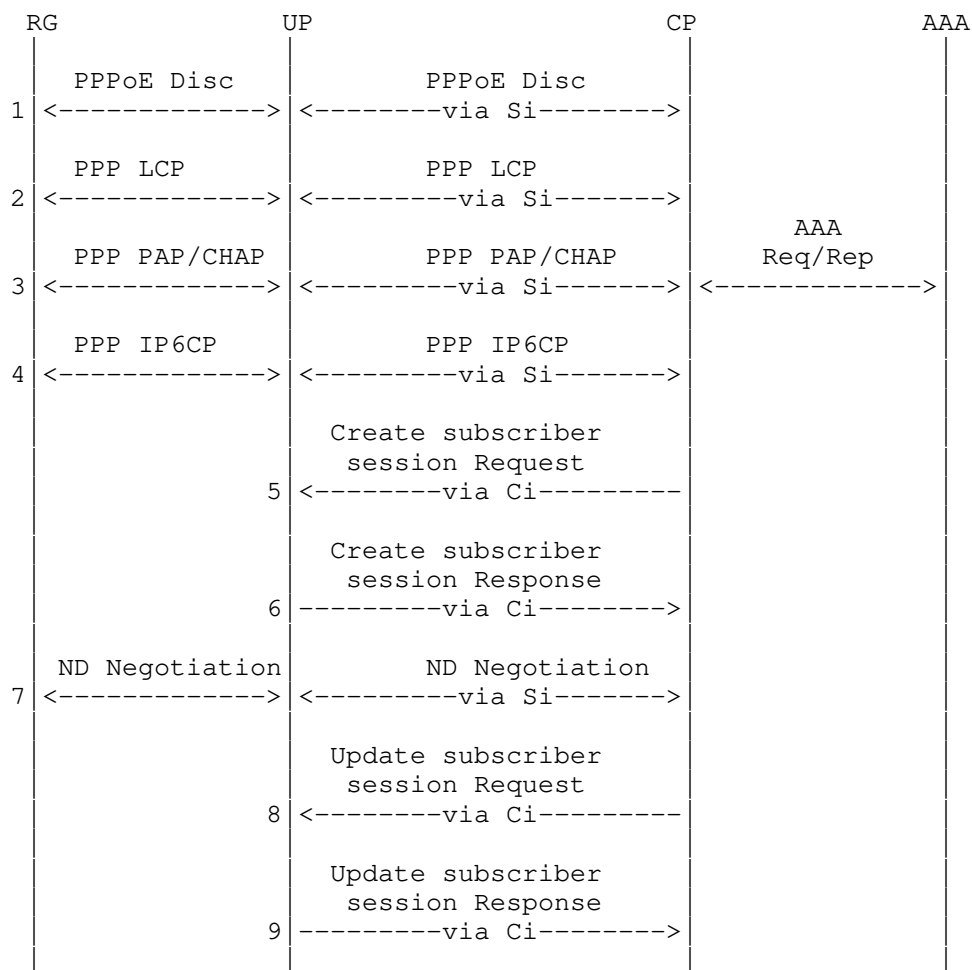
```

<Update_Response Message> ::= <Common Header>
                               <Update Response TLV>
                               [<Subscriber CGN Port Range TLV>]

```

### 5.2.2 IPv6 PPPoE Access

The following figure describes the IPv6 PPPoE access call flow.



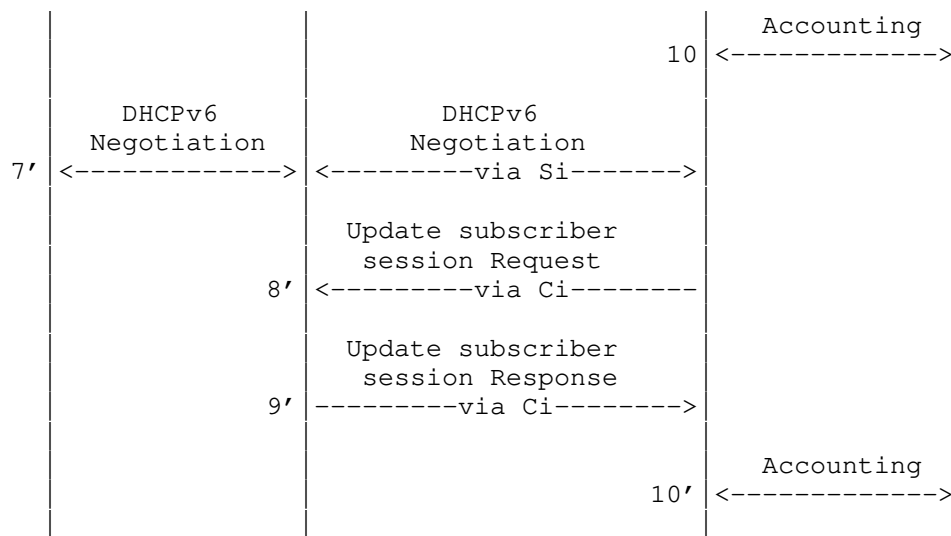


Figure 21: IPv6 PPPoE Access

From the above sequence, steps 1-4 are the standard PPPoE call flow. The UP is responsible for redirecting the PPPoE control packets to the CP or RG. The PPPoE control packets are transmitted between the CP and UP through the Si.

After the PPPoE call flow, if the subscriber passed the AAA authentication and authorization, the CP will create a corresponding session on the UP through the Ci. The formats of the messages are as follows:

```

<Update_Request Message> ::= <Common Header>
                             <Basic Subscriber TLV>
                             <PPP Subscriber TLV>
                             <IPv6 Subscriber TLV>
                             <IPv6 Routing TLV>
                             [<Subscriber Policy TLV>]
  
```

```

<Update_Response Message> ::= <Common Header>
                             <Update Response TLV>
  
```

Then, the RG will initialize a ND/DHCPv6 negotiation process with the CP (see step 7 and 7'), after that, it will trigger an update (8-9, 8'-9') to the subscriber session. The formats of the update messages are as follows:

```

<Update_Request Message> ::= <Common Header>
                               <Basic Subscriber TLV>
                               <PPP Subscriber TLV>
                               <IPv6 Subscriber TLV>
                               <IPv6 Routing TLV>
                               [<Subscriber Policy TLV>]

```

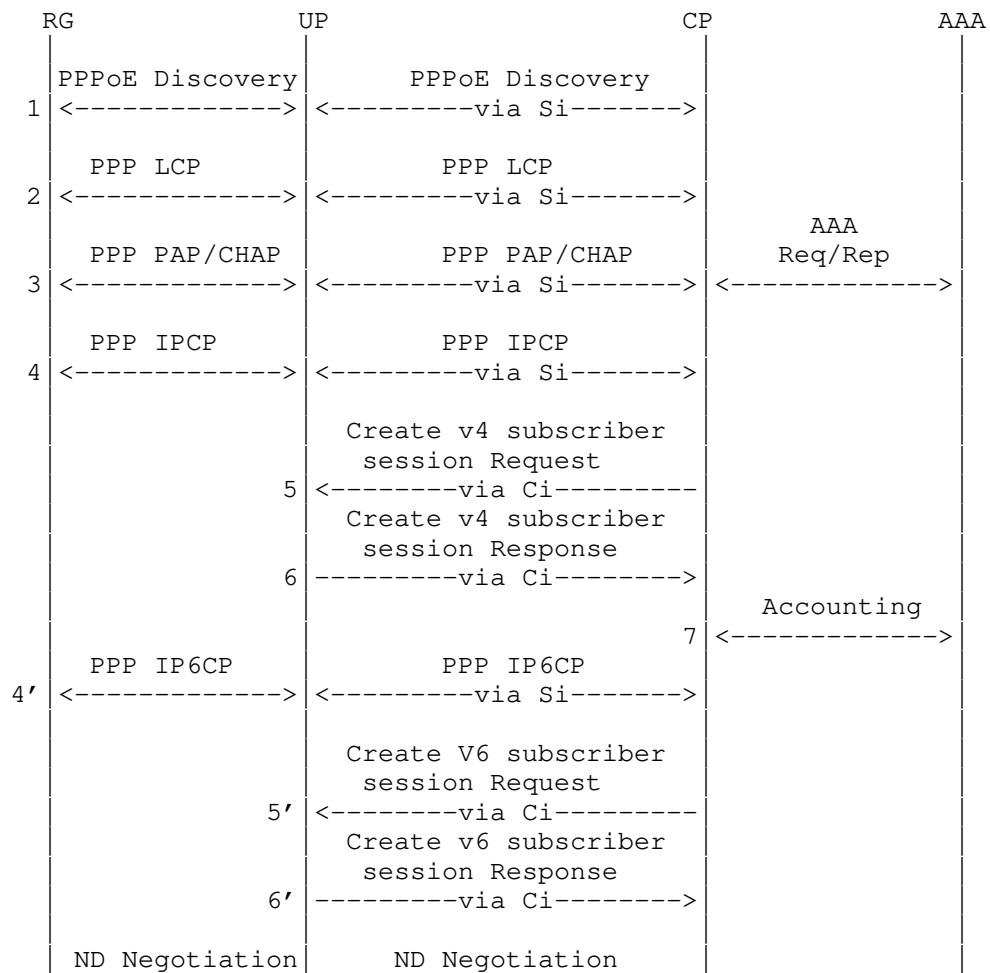
```

<Update_Response Message> ::= <Common Header>
                               <Update Response TLV>

```

### 5.2.3 PPPoE Dual Stack Access

The following figure shows a combination of IPv4 and IPv6 PPPoE access call flow.



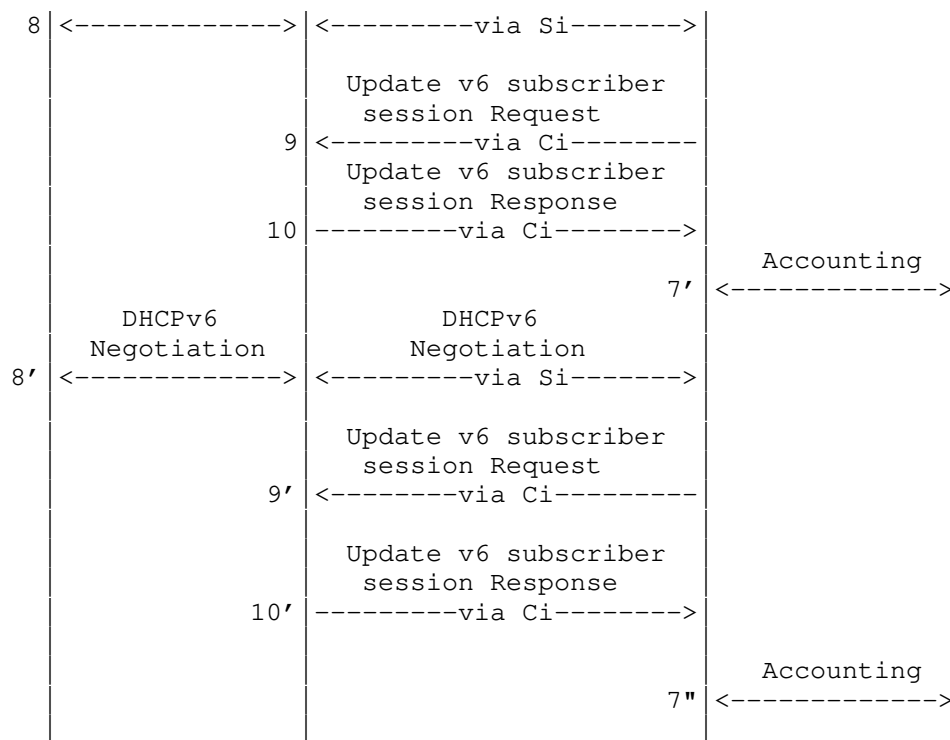


Figure 22: PPPoE Dual Stack Access

PPPoE dual stack is a combination of IPv4 PPPoE and IPv6 PPPoE access. The process is as above. The formats of the messages are as follows:

1. Create an IPv4 PPPoE subscriber session (5-6)

```

<Update_Request Message> ::= <Common Header>
    <Basic Subscriber TLV>
    <PPP Subscriber TLV>
    <IPv4 Subscriber TLV>
    <IPv4 Routing TLV>
    [<Subscriber Policy TLV>]
  
```

```

<Update_Response Message> ::= <Common Header>
    <Update Response TLV>
    [<Subscriber CGN Port Range TLV>]
  
```

2. Create an IPv6 PPPoE subscriber session (5'-6')

```

<Update_Request Message> ::= <Common Header>
    <Basic Subscriber TLV>
    <PPP Subscriber TLV>
  
```

```

    <IPv6 Subscriber TLV>
    <IPv6 Routing TLV>
    [<Subscriber Policy TLV>]

```

```

<Update_Response Message> ::= <Common Header>
    <Update Response TLV>

```

### 3. Update the IPv6 PPPoE subscriber session (9-10, 9'-10')

```

<Update_Request Message> ::= <Common Header>
    <Basic Subscriber TLV>
    <PPP Subscriber TLV>
    <IPv6 Subscriber TLV>
    <IPv6 Routing TLV>
    [<Subscriber Policy TLV>]

```

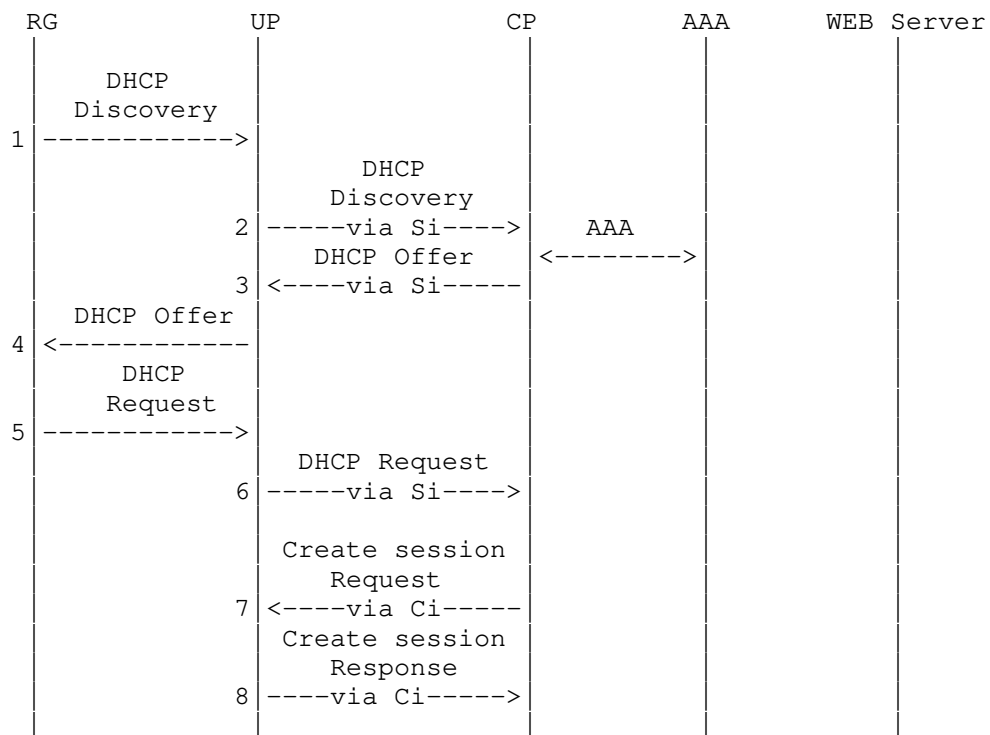
```

<Update_Response Message> ::= <Common Header>
    <Update Response TLV>

```

## 5.3 WLAN Access

The following figure shows the WLAN access call flow.



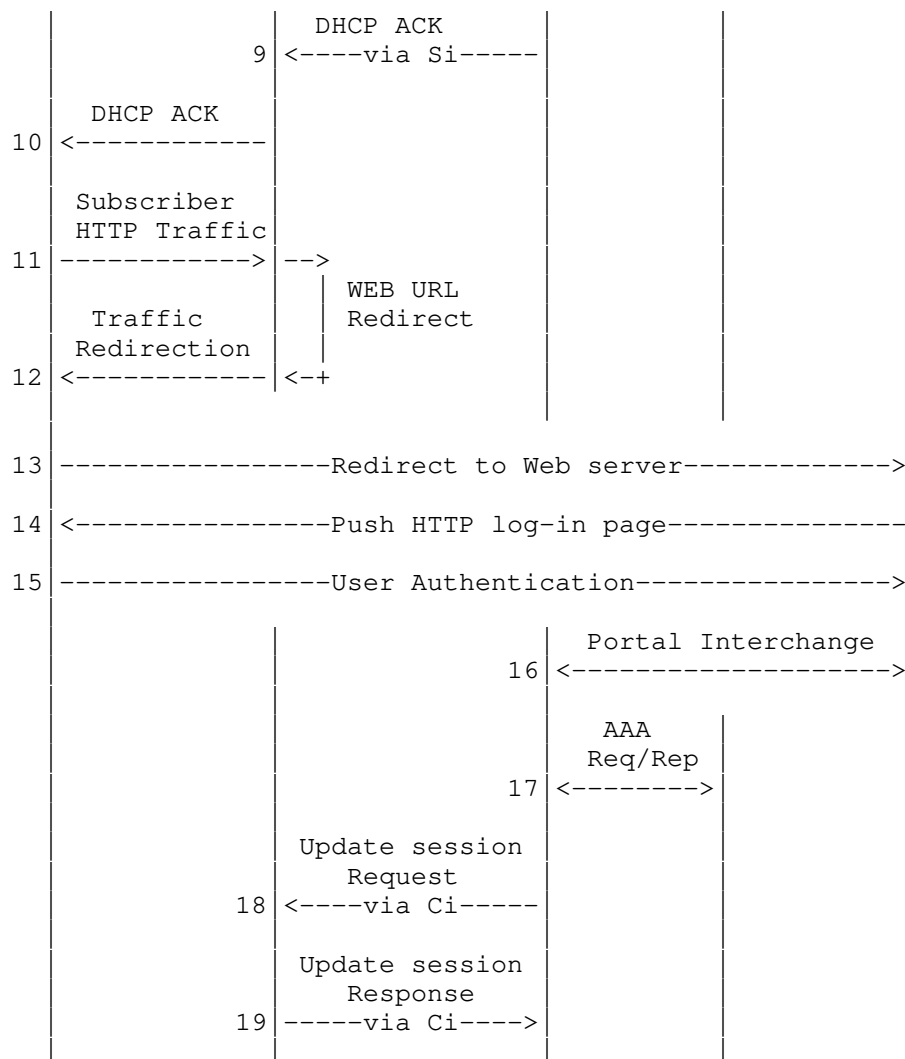


Figure 23: WLAN Access

WLAN access starts with the DHCP dial-up process (steps 1-6), after that the CP will create a subscriber session on the UP (steps 7-8). The formats of the session creation messages are as follows:

IPv4 Case:

```

<Update_Request Message> ::= <Common Header>
                             <Basic Subscriber TLV>
                             <IPv4 Subscriber TLV>
                             <IPv4 Routing TLV>
                             [<Subscriber Policy TLV>]
  
```

```

<Update_Response Message> ::= <Common Header>
                                <Update Response TLV>
                                [<Subscriber CGN Port Range TLV>]

```

IPv6 Case:

```

<Update_Request Message> ::= <Common Header>
                                <Basic Subscriber TLV>
                                <IPv6 Subscriber TLV>
                                <IPv6 Routing TLV>
                                [<Subscriber Policy TLV>]

```

```

<Update_Response Message> ::= <Common Header>
                                <Update Response TLV>

```

After step 10, the RG will be allocated an IP address and its first HTTP packet will be redirected to a WEB server for subscriber authentication (steps 11-17). After the WEB authentication, if the result is positive, the CP will update the subscriber session by using the following message exchanges:

```

IPv4 Case: <Update_Request Message> ::= <Common Header>
                                                <Basic Subscriber TLV>
                                                <IPv4 Subscriber TLV>
                                                <IPv4 Routing TLV>
                                                [<Subscriber Policy TLV>]

```

```

<Update_Response Message> ::= <Common Header>
                                <Update Response TLV>
                                [<Subscriber CGN Port Range TLV>]

```

```

IPv6 Case: <Update_Request Message> ::= <Common Header>
                                                <Basic Subscriber TLV>
                                                <IPv6 Subscriber TLV>
                                                <IPv6 Routing TLV>
                                                [<Subscriber Policy TLV>]

```

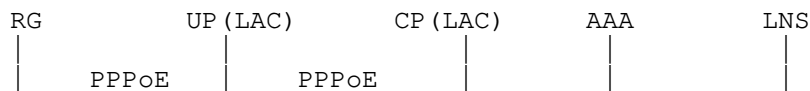
```

<Update_Response Message> ::= <Common Header>
                                <Update Response TLV>

```

## 5.4 L2TP

### 5.4.1 L2TP LAC Access





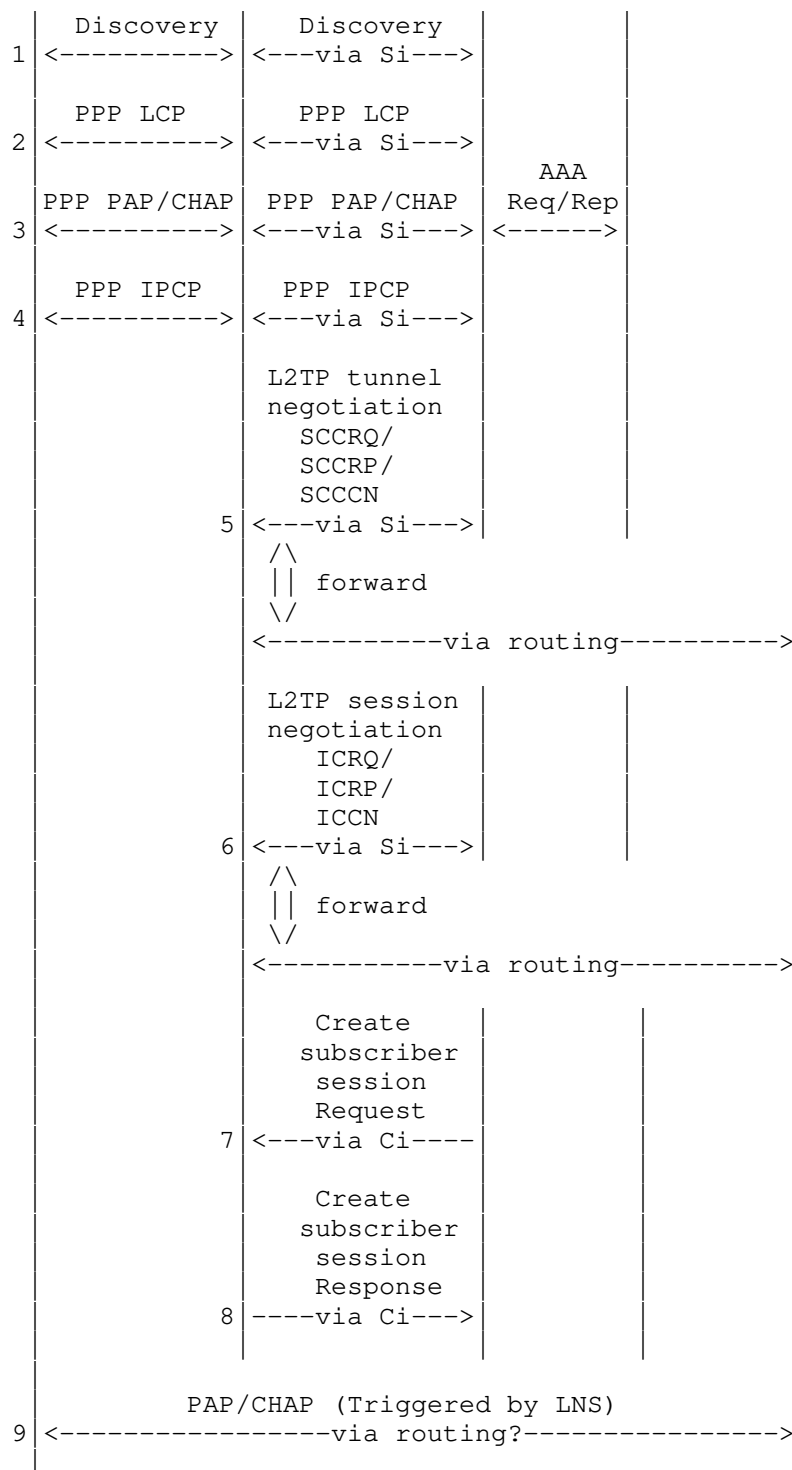


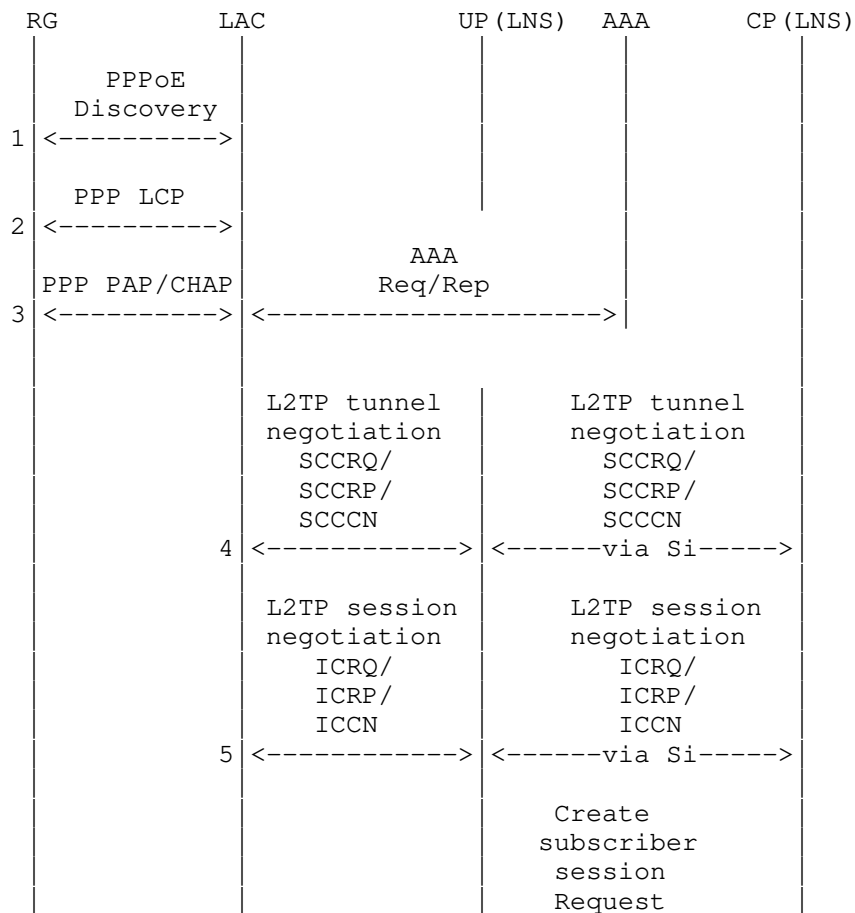
Figure 24: L2TP-LAC Access

Steps 1-4 are a standard PPPoE access process. After that the LAC-CP starts to negotiate an L2TP session and tunnel with the LNS. After the negotiation, the CP will create an L2TP LAC subscriber session on the UP through the following messages:

```
<Update_Request Message> ::= <Common Header>
                               <Basic Subscriber TLV>
                               <L2TP-LAC Subscriber TLV>
                               <L2TP-LAC Tunnel TLV>
```

```
<Update_Response Message> ::= <Common Header>
                               <Update Response TLV>
```

#### 5.4.2 L2TP LNS IPv4 Access



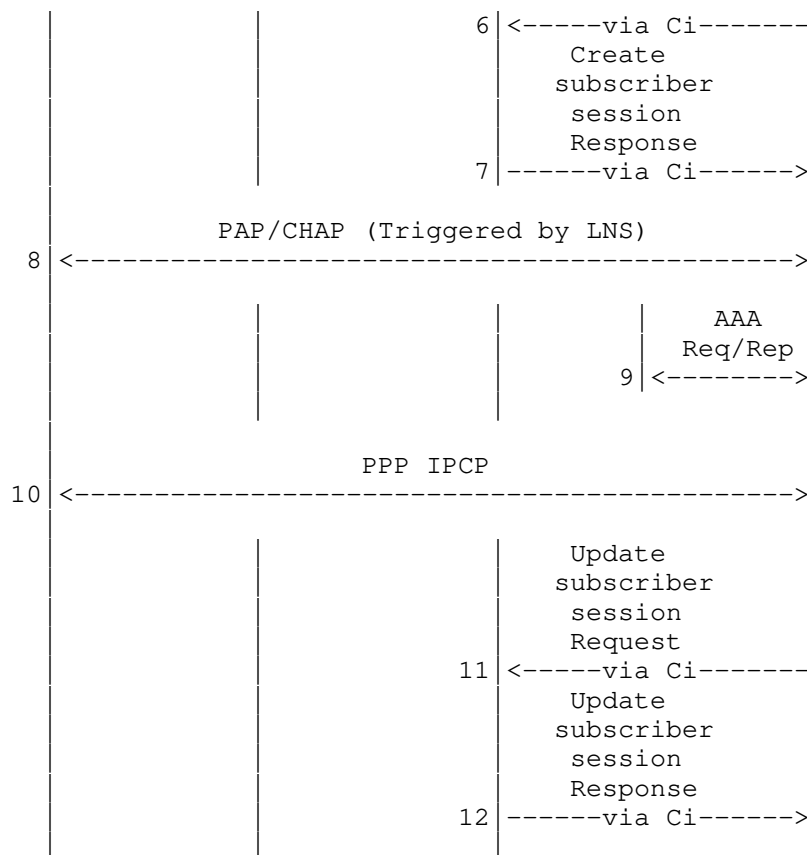


Figure 25: IPv4 L2TP-LNS Access

In this case, the BNG is running as an LNS and separated into LNS-CP and LNS-UP. Steps 1-5 finish the normal L2TP dial-up process. When the L2TP session and tunnel negotiations are finished, the LNS-CP will create an L2TP LNS subscriber session on the LNS-UP. The format of messages are as follows:

```

<Update_Request Message> ::= <Common Header>
                             <L2TP-LNS Subscriber TLV>
                             <Basic Subscriber TLV>
                             <PPP Subscriber TLV>
                             <IPv4 Subscriber TLV>
                             <IPv4 Routing TLV>
                             <L2TP-LNS Tunnel TLV>
                             [<Subscriber Policy TLV>]
  
```

```

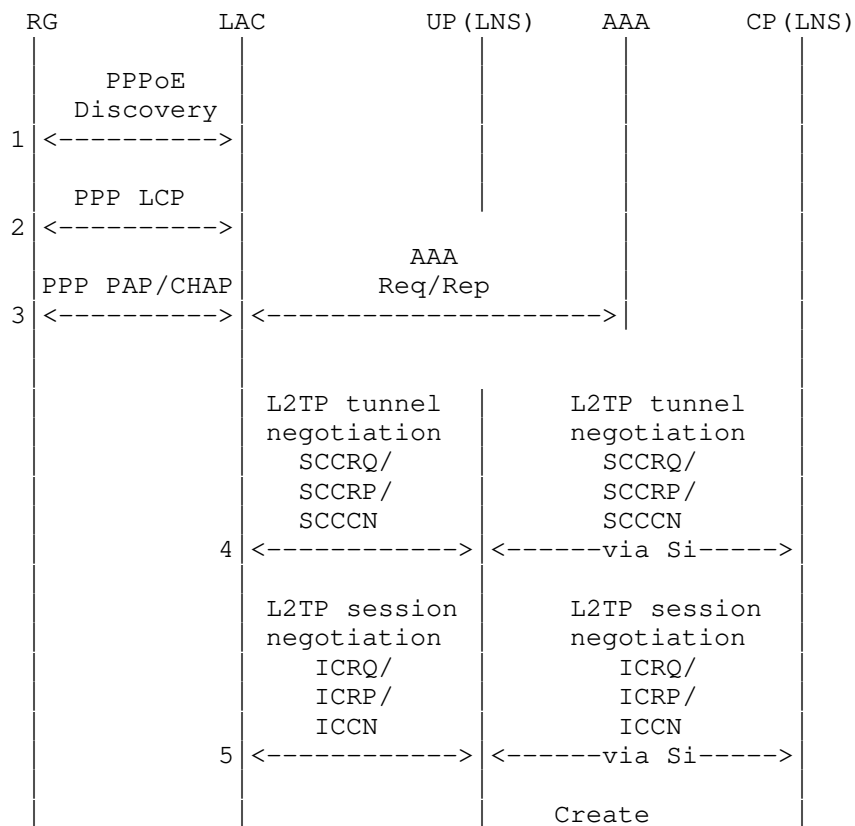
<Update_Response Message> ::= <Common Header>
                             <Update Response TLV>
                             [<Subscriber CGN Port Range TLV>]
  
```

After that, the LNS-CP will trigger an AAA authentication. If the authentication result is positive, a PPP IPCP process will follow, then the CP will update the session with the following message exchanges:

```
<Update_Request Message> ::= <Common Header>
    <L2TP-LNS Subscriber TLV>
    <Basic Subscriber TLV>
    <PPP Subscriber TLV>
    <IPv4 Subscriber TLV>
    <IPv4 Routing TLV>
    <L2TP-LNS Tunnel TLV>
    [<Subscriber Policy TLV>]
```

```
<Update_Response Message> ::= <Common Header>
    <Update Response TLV>
    [<Subscriber CGN Port Range TLV>]
```

#### 5.4.3 L2TP LNS IPv6 Access



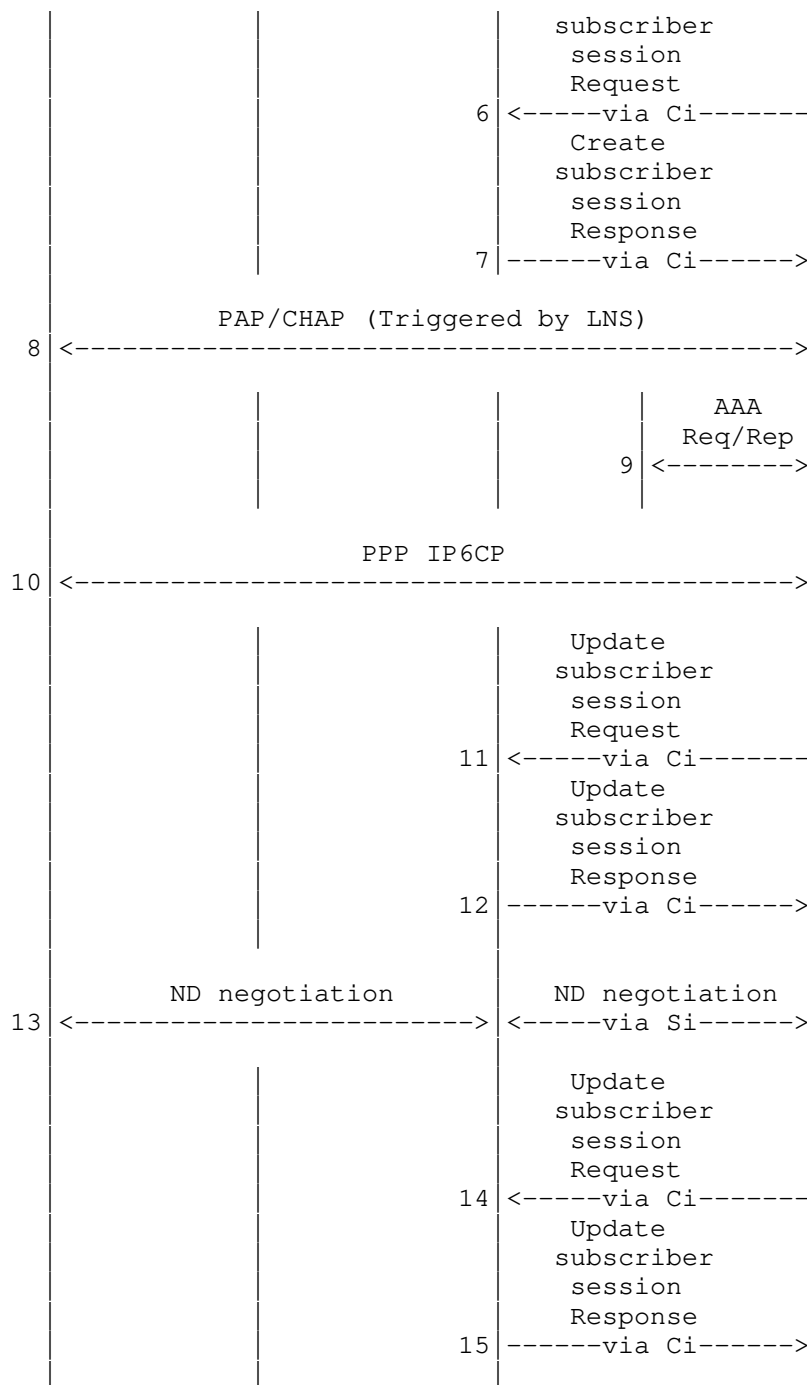


Figure 26: L2TP-LNS IPv6 Access

Steps 1-12 are the same as L2TP and LNS IPv4 Access. Steps 1-5 finish the normal L2TP dial-up process. When the L2TP session and tunnel negotiations are finished, the LNS-CP will create an L2TP LNS subscriber session on the LNS-UP. The format of the messages is as follows:

```
<Update_Request Message> ::= <Common Header>
                               <L2TP-LNS Subscriber TLV>
                               <Basic Subscriber TLV>
                               <PPP Subscriber TLV>
                               <IPv6 Subscriber TLV>
                               <IPv6 Routing TLV>
                               <L2TP-LNS Tunnel TLV>
                               [<Subscriber Policy TLV>]
```

```
<Update_Response Message> ::= <Common Header>
                               <Update Response TLV>
```

After that, the LNS-CP will trigger a AAA authentication. If the authentication result is positive, a PPP IP6CP process will follow, then the CP will update the session with the following message exchanges:

```
<Update_Request Message> ::= <Common Header>
                               <L2TP-LNS Subscriber TLV>
                               <Basic Subscriber TLV>
                               <PPP Subscriber TLV>
                               <IPv6 Subscriber TLV>
                               <IPv6 Routing TLV>
                               <L2TP-LNS Tunnel TLV>
                               [<Subscriber Policy TLV>]
```

```
<Update_Response Message> ::= <Common Header>
                               <Update Response TLV>
```

Then, an ND negotiation will be triggered by the RG. After the ND negotiation, the CP will update the session with the following message exchanges:

```
<Update_Request Message> ::= <Common Header>
                               <L2TP-LAC Subscriber TLV>
                               <Basic Subscriber TLV>
                               <PPP Subscriber TLV>
                               <IPv6 Subscriber TLV>
                               <IPv6 Routing TLV>
                               <L2TP-LNS Tunnel TLV>
                               [<Subscriber Policy TLV>]
```

```
<Update_Response Message> ::= <Common Header>
                               <Update Response TLV>
```

## 5.5 CGN (Carrier Grade NAT)

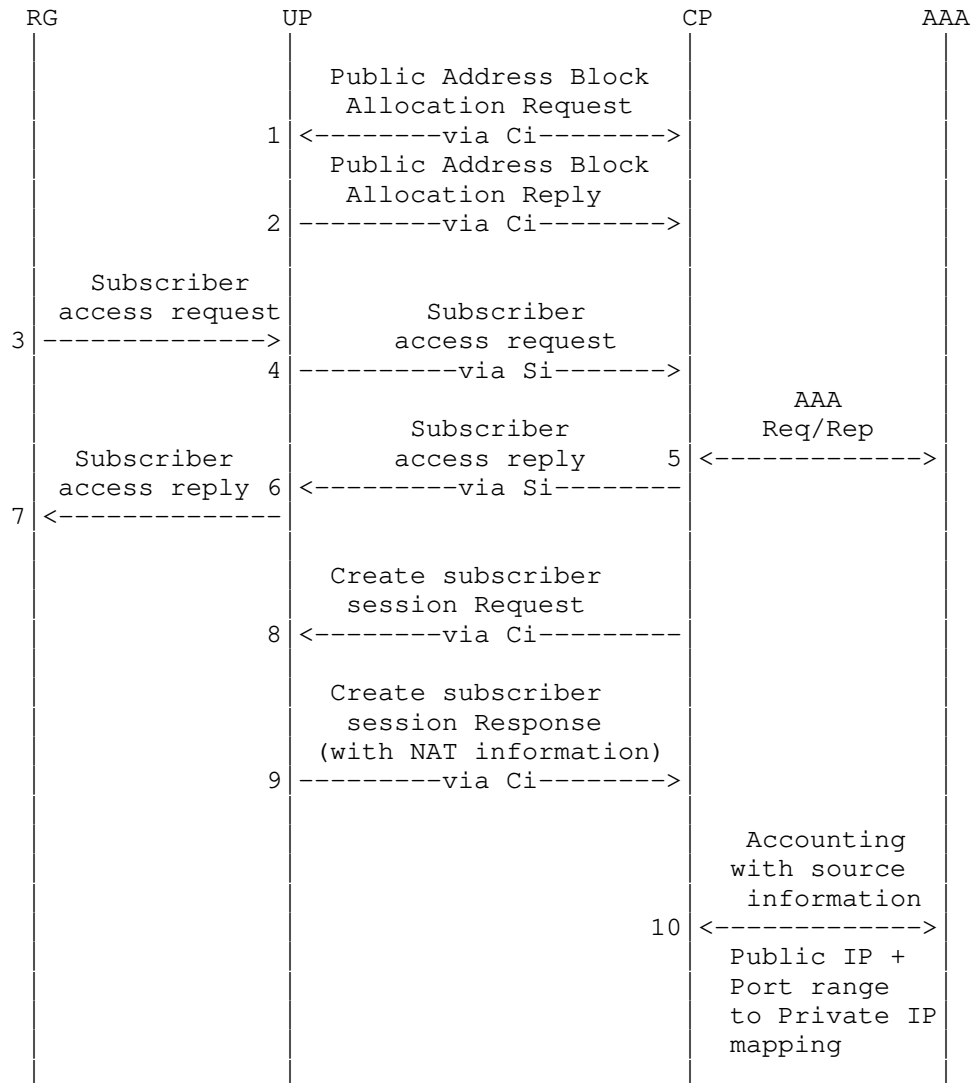


Figure 27: CGN Access

The first steps allocate one or more CGN address blocks to the UP (steps 1-2). This is achieved by the following message exchanges between CP and UP.

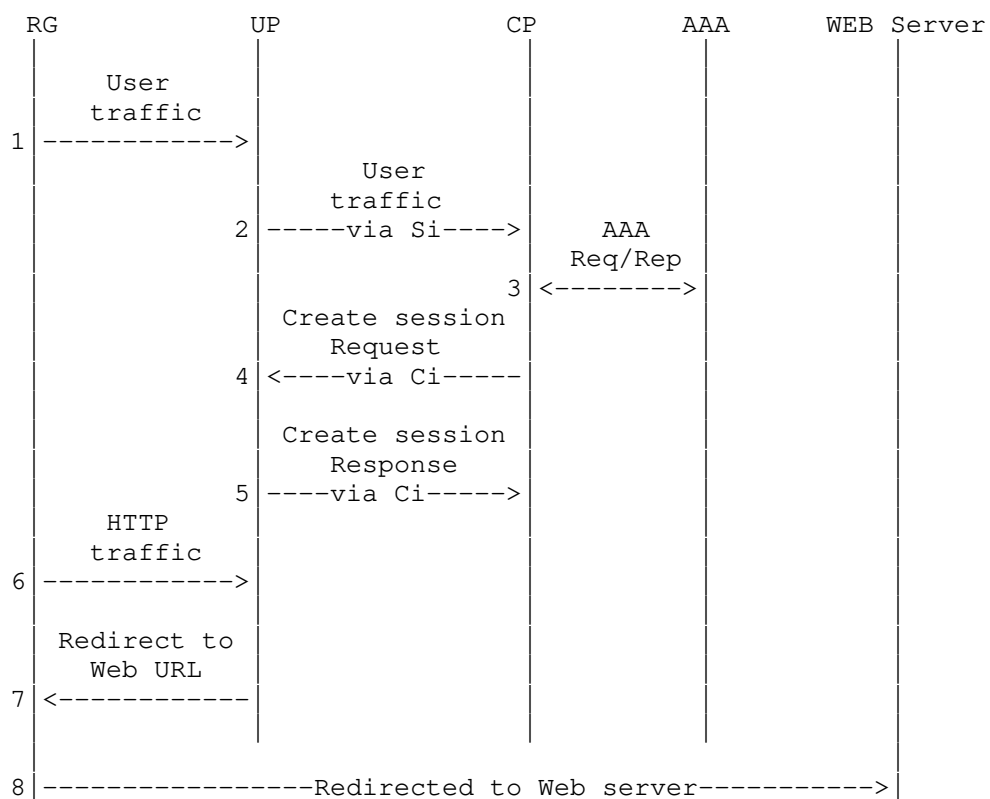
[illegible][illegible]

Steps 3-9 show the general dial-up process in the case of CGN mode. The specific processes (e.g., IPoE, PPPoE, L2TP, etc.) are defined in above sections.

If a subscriber is a CGN subscriber, once the subscriber session is created/updated, the UP will report the NAT information to the CP. This is achieved by carrying the "Subscriber CGN Port Range TLV" in the Update\_Response message.

## 5.6 L3 Leased Line Access

### 5.6.1 Web Authentication





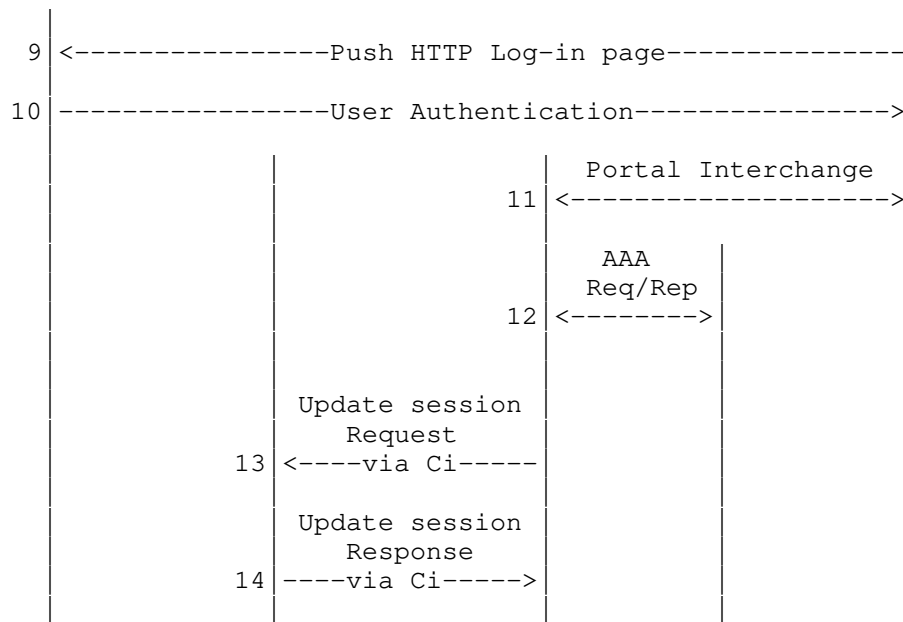


Figure 28: Web Authentication based L3 Leased Line Access

In this case, IP traffic from the RG will trigger the CP to authenticate the RG by checking the source IP and the exchanges with the AAA server. Once the RG passed the authentication, the CP will create a corresponding subscriber session on the UP through the following message exchanges:

## IPv4 Case:

```

<Update_Request Message> ::= <Common Header>
    <Basic Subscriber TLV>
    <IPv4 Subscriber TLV>
    <IPv4 Routing TLV>
    [<Subscriber Policy TLV>]
  
```

```

<Update_Response Message> ::= <Common Header>
    <Update Response TLV>
    [<Subscriber CGN Port Range TLV>]
  
```

## IPv6 Case:

```

<Update_Request Message> ::= <Common Header>
    <Basic Subscriber TLV>
    <IPv6 Subscriber TLV>
    <IPv6 Routing TLV>
    [<Subscriber Policy TLV>]
  
```

```

<Update_Response Message> ::= <Common Header>
    <Update Response TLV>
  
```

Then, the HTTP traffic from the RG will be redirected to a WEB server to finish the WEB authentication. Once the WEB authentication is passed, the CP will trigger another AAA authentication. After the AAA authentication, the CP will update the session with the following message exchanges:

IPv4 Case:

```
<Update_Request Message> ::= <Common Header>
                               <Basic Subscriber TLV>
                               <IPv4 Subscriber TLV>
                               <IPv4 Routing TLV>
                               [<Subscriber Policy TLV>]
```

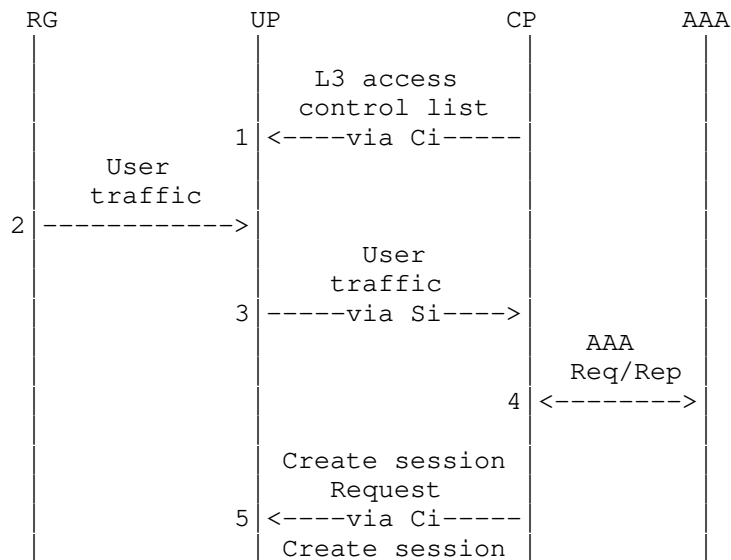
```
<Update_Response Message> ::= <Common Header>
                               <Update Response TLV>
                               [<Subscriber CGN Port Range TLV>]
```

IPv6 Case:

```
<Update_Request Message> ::= <Common Header>
                               <Basic Subscriber TLV>
                               <IPv6 Subscriber TLV>
                               <IPv6 Routing TLV>
                               [<Subscriber Policy TLV>]
```

```
<Update_Response Message> ::= <Common Header>
                               <Update Response TLV>
```

### 5.6.2 User Traffic Trigger



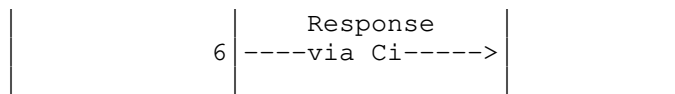


Figure 29: User Traffic Triggered L3 Leased Line Access

In this user traffic triggered case, the CP must install an access control list on the UP, which is used by the UP to determine whether an RG is legal or not. If the traffic is from a legal RG, it will be redirected to the CP through the Si. The CP will trigger a AAA interchange with the AAA server. After that, the CP will create a corresponding subscriber session on the UP with the following message exchanges:

IPv4 Case:

```

<Update_Request Message> ::= <Common Header>
    <Basic Subscriber TLV>
    <IPv4 Subscriber TLV>
    <IPv4 Routing TLV>
    [<Subscriber Policy TLV>]
  
```

```

<Update_Response Message> ::= <Common Header>
    <Update Response TLV>
    [<Subscriber CGN Port Range TLV>]
  
```

IPv6 Case:

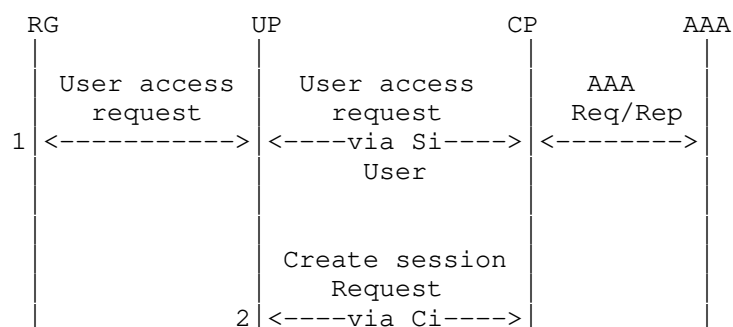
```

<Update_Request Message> ::= <Common Header>
    <Basic Subscriber TLV>
    <IPv6 Subscriber TLV>
    <IPv6 Routing TLV>
    [<Subscriber Policy TLV>]
  
```

```

<Update_Response Message> ::= <Common Header>
    <Update Response TLV>
  
```

## 5.7 Multicast Access



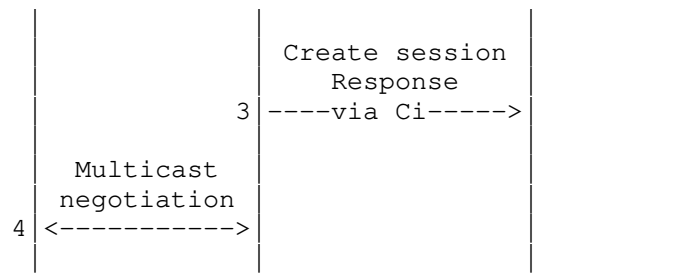


Figure 30: Multicast Access

Multicast access starts with an user access request from the RG. The request will be redirected to the CP by the Si. A follow-up AAA interchange between the CP and the AAA server will be triggered. After the authentication, the CP will create a multicast subscriber session on the UP through the following messages:

IPv4 Case:

```

<Update_Request Message> ::= <Common Header>
    <Multicast Group Information TLV>
    <Basic Subscriber TLV>
    <IPv4 Subscriber TLV>
    <IPv4 Routing TLV>
    [<Subscriber Policy TLV>]
  
```

```

<Update_Response Message> ::= <Common Header>
    <Update Response TLV>
    [<Subscriber CGN Port Range TLV>]
  
```

IPv6 Case:

```

<Update_Request Message> ::= <Common Header>
    <Multicast Group Information TLV>
    <Basic Subscriber TLV>
    <IPv6 Subscriber TLV>
    <IPv6 Routing TLV>
    [<Subscriber Policy TLV>]
  
```

```

<Update_Response Message> ::= <Common Header>
    <Update Response TLV>
  
```

## 6. S-CUSP Message Formats

An S-CUSP message consists of a common header followed by a variable-length body consisting entirely of TLVs. Receiving an S-CUSP message with an unknown message type or missing mandatory TLV MUST trigger an Error message (see Section 6.7) or a response message with an Error Information TLV (see Section 7.6).

Conversely, if a TLV is optional, the TLV may or may not be present. Optional TLVs are indicated in the message formats shown in this document by being enclosed in square brackets.

This section specifies the format of the common S-CUSP message header and lists the defined messages.

Network byte order is used for all multi-byte fields.

### 6.1 Common Message Header

S-CUSP Common Message Header:

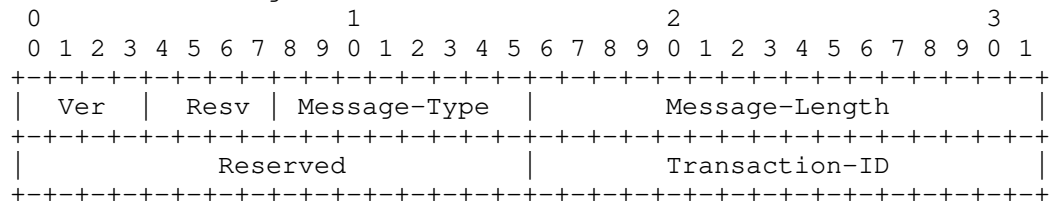


Figure 6.1: S-CUSP Message Common Header

- o Ver (4 bits): The major version of the protocol. This document specifies version 1. Different major versions of the protocol may have significantly different message structure and format except that the Ver field will always be in the same place at the beginning of each message. A successful S-CUSP session depends on the CP and the UP both using the same major version of the protocol.
- o Resv (4 bits): Reserved. MUST be sent as zero and ignored on receipt.
- o Message-Type (8 bits): The set of message types specified in this document is listed in Section 9.1.
- o Message-Length (16 bits): Total length of the S-CUSP message including the common header, expressed in number of bytes as an unsigned integer.

- o Transaction ID (16 bits): This field is used to identify requests. It is echoed back in any corresponding ACK / response / Error message. It is RECOMMENDED that a monotonically increasing value be used in successive message and that value wrap back to zero after 0xFFFF. The contents of this field is an opaque value that the receiver MUST NOT use for any purpose except to echo back in a corresponding response and, optionally, for logging.

## 6.2 Control Messages

This document defines the following control messages:

Type	Name	Notes and TLVs that can be carried
-----	----	-----
1	Hello	Hello TLV, Keep-Alive TLV.
2	Keepalive	A common header with the Keepalive message type.
3	Sync_Request	Synchronization request.
4	Sync_Begin	Synchronization starts.
5	Sync_Data	Synchronization data: TLVs specified in Section 5.
6	Sync_End	End synchronization.
7	Update_Request	TLVs specified in Sections 7.6-7.9.
8	Update_Response	TLVs specified in Sections 7.6-7.9.

### 6.2.1 Hello Message

Hello message is used for S-CUSP session establishment and version negotiation. The detail of S-CUSP session establishment and version negotiation can be found in Section 4.1.1.

The format of Hello message is as follows:

```
<Hello Message> ::= <Common Header>
                    <Hello TLV>
                    <Keepalive TLV>
                    [<Error Information TLV>]
```

The return code and negotiation result will be carried in the Error Information TLV. They are listed as follows:

- 0: Success, version negotiation success.
- 1: Failure, malformed message received.

2: One or more of the TLVs was not understood.

1001: The version negotiation fails. The S-CUSP session establishment phase fails.

1002: The keepalive negotiation fails. The S-CUSP session establishment phase fails.

1003: The establishment timer expires. session establishment phase fails.

#### 6.2.2 Keepalive Message

The Keepalive message is periodically sent by each end of an S-CUSP session. It is used to detect whether the peer end is still alive. The Keepalive procedures are defined Section 4.1.2.

The format of the Keepalive message is as follows:

<Keepalive Message> ::= <Common Header>

#### 6.2.3 Sync\_Request Message

The Sync\_Request message is used to request synchronization from an S-CUSP peer. Both CP and UP can request their peer to synchronize data.

The format of the Sync\_Request message is as follows:

<Sync\_Request Message> ::= <Common Header>

A Sync\_Request message may result in a Sync\_Begin message from its peer. The Sync\_Begin message is defined in Section 6.2.4.

#### 6.2.4 Sync\_Begin Message

The Sync\_Begin message is a reply to a Sync\_Request message. It is used to notify the synchronization requester whether the synchronization can be started.

The format of Sync\_Begin message is as follows:

<Sync\_Begin Message> ::= <Common Header>  
                                  <Error Information TLV>

The return codes are carried in the Error Information TLV. The codes are listed below:

- 0: Success, be ready to synchronize.
- 1: Failure, malformed message received.
- 2: One or more of the TLVs was not understood.
- 2001: Synch-NoReady. The data to be synchronized is not ready.
- 2002: Synch-Unsupport. The data synchronization is not supported.

#### 6.2.5 Sync\_Data Message

The Sync\_Data message is used to send data being synchronized between the CP and UP. The Sync\_Data message has the same function and format as the Update\_Request message. The difference is that there is no ACK for a Sync\_Data message. An error caused by the Sync\_Data message will result in a Sync\_End message.

There are two scenarios:

Synchronization from UP to CP: Synchronize the resource data to CP.

```
<Sync_Data Message> ::= <Common Header>
                        [<Resource Reporting TLVs>]
```

Synchronization from CP to UP: Synchronize all subscriber sessions to UP. As for which TLVs should be carried, it depends on the specific session data to be synchronized. This is equivalent to create the specific session. Refer to Section 5 to see more details.

```
<Sync_Data Message> ::= <Common Header>
                        [<User Routing TLVs>]
                        [<User Information TLVs>]
                        [<L2TP Subscriber TLVs>]
                        [<Subscriber CGN Port Range TLV>]
                        [<Subscriber Policy TLV>]
```

#### 6.2.6 Sync\_End Message

The Sync\_End message is used to indicate the end of a synchronization process. The format of a Sync\_End message is as follows:



```
<Sync_End Message> ::= <Common Header>
                        <Error Information TLV>
```

The return/error codes are listed as follows:

- 0: Success, synchronization finished.
- 1: Failure, malformed message received.
- 2: One or more of the TLVs was not understood.

#### 6.2.7 Update\_Request Message

The Update\_Request message is a multi-task message, it can be used to create, update, and delete subscriber sessions on a UP.

For session operations, the specific operation is controlled by the "Oper" field of the carried TLVs. As defined in Section 7.1, the "Oper" can be set to either "update" or "delete" when a TLV is carried in an Update\_Request message.

When the "Oper" set to update, it means to create or update a subscriber session, if the "Oper" set to delete, it indicates to delete a corresponding session on an UP.

The format of Update\_Request message is as follows:

```
<Update_Request Message> ::= <Common Header>
                              [<User Routing TLVs>]
                              [<User Information TLVs>]
                              [<L2TP Subscriber TLVs>]
                              [<Subscriber CGN Port Range TLV>]
                              [<Subscriber Policy TLV>]
```

Each Update\_Request message will result in an Update\_Response message that is defined in Section 6.2.8.

#### 6.2.8 Update\_Response Message

The Update\_Response message is a response to an Update\_Request message. It is used to confirm the update request (or reject it in the case of an error). The format of an Update\_Response message is as follows:

```
<Update_Response Message> ::= <Common Header>
                               [<Subscriber CGN Port Range TLV>]
                               <Error Information TLV>
```

The return/error codes are carried in the Error Information TLV. They are listed as follows:

- 0: Success.
- 1: Failure, malformed message received.
- 2: One or more of the TLVs was not understood.
- 3001 (Pool-Mismatch): The corresponding address pool cannot be found.
- 3002 (Pool-Full): The address pool is fully allocated and no address segment is available.
- 3003 (Subnet-Mismatch): The address pool subnet cannot be found.
- 3004 (Subnet-Conflict): Subnets in the address pool have been classified into other clients.
- 4001 (Update-Fail-No-Res): The forwarding table fails to be delivered because the forwarding resources are insufficient.
- 4002 (QoS-Update-Success): The QoS policy takes effect.
- 4003 (QoS-Update-Sq-Fail): Failed to process the queue in the QoS policy.
- 4004 (QoS-Update-CAR-Fail): Processing of the CAR in the QoS policy fails.
- 4005 (Statistic-Fail-No-Res): Statistics processing failed due to insufficient statistics resources.

### 6.3 Event Message

The Event message is used to report subscriber session traffic statistics and detection information. The format of Event message is as follows:

```
<Event Message> ::= <Common Header>
                    [<User Traffic Statistics Report TLV>]
                    [<User Detection Result Report TLV>]
```

#### 6.4 Report Message

The Report message is used to report board and interface status on a UP. The format of Report message is as follows:

```
<Report Message> ::= <Common Header>
                        [<Board Status TLVs>]
                        [<Interface Status TLVs>]
```

#### 6.5 CGN Messages

This document defines the following resource allocation messages:

Type	Message Name	TLV that is carried
200	Addr_Allocation_Req	Address Allocation Request
201	Addr_Allocation_Ack	Address Allocation Response
202	Addr_Renew_Req	Address Renewal Request
203	Addr_Renew_Ack	Address Renewal Response
204	Addr_Release_Req	Address Release Request
205	Addr_Release_Ack	Address Release Response

##### 6.5.1 Addr\_Allocation\_Req Message

The Addr\_Allocation\_Req message is used to request CGN address allocation. The format of Addr\_Allocation\_Req message is as follows:

```
<Addr_Allocation_Req Message> ::= <Common Header>
                                   <Address Allocation Request TLV>
```

##### 6.5.2 Addr\_Allocation\_Ack Message

The Addr\_Allocation\_Ack message is a response to an Addr\_Allocation\_Req message. The format of Addr\_Allocation\_Ack message is as follows:

```
<Addr_Allocation_Ack Message> ::= <Common Header>
                                   <Address Allocation Response TLV>
```

### 6.5.3 Addr\_Renew\_Req Message

The Addr\_Renew\_Req message is used to request address renewal. The format of Addr\_Renew\_Req message is as follows:

```
<Addr_Renew_Req Message> ::= <Common Header>
                               <Address Renewal Request TLV>
```

### 6.5.4 Addr\_Renew\_Ack Message

The Addr\_Renew\_Ack message is a response to an Addr\_Renew\_Req message. The format of Addr\_Renew\_Ack message is as follows:

```
<Addr_Renew_Ack Message> ::= <Common Header>
                               <Address Renewal Response TLV>
```

### 6.5.5 Addr\_Release\_Req Message

The Addr\_Release\_Req message is used to request address release. The format of Addr\_Release\_Req message is as follows:

```
<Addr_Release_Req Message> ::= <Common Header>
                               <Address Release Request TLV>
```

### 6.5.6 Addr\_Release\_Ack Message

The Addr\_Release\_Ack message is a response to an Addr\_Release\_Req message. The format of Addr\_Release\_Ack message is as follows:

```
<Addr_Release_Ack Message> ::= <Common Header>
                               <Address Release Response TLV>
```

## 6.6 Vendor Message

The Vendor message is, in conjunction with the vendor TLV and vendor sub-TLV, can be used by vendors to extend the S-CUSP protocol. It's message type is 11. If the receiver does not recognize the message, an Error message will be returned to the sender.

The format of the Vendor message is as follows:

```
<Vendor Message> ::= <Common Header>
                        <Vendor TLV>
                        [<any other TLVs as specified by the vendor>]
```

## 6.7 Error Message

The Error message is defined to return some critical error information to the sender. If a receiver does not know the message type of a received message, it MUST return an Error message to the sender.

The format of the Error message is as below:

```
<Error Message> ::= <Common Header>
                    <Error Information TLV>
```

## 7. S-CUSP TLVs and Sub-TLVs

This section specifies the following:

- o the format of the TLVs that appear in S-CUSP messages,
- o the format of the sub-TLVs that appear within the values of some TLVs, and
- o the format of some basic data fields that appear within TLVs or sub-TLVs.

See Section 9 for a list of all defined TLVs and sub-TLVs.

### 7.1 Common TLV Header

S-CUSP messages consist of the common header specified in Section 6.1 followed by TLVs formatted as specified in this section.

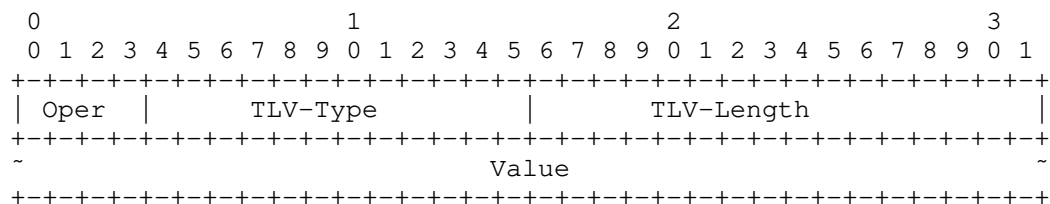


Figure 32: Common TLV Header

- o Oper (4 bits): For Message-Types that indicate an operation on a data set, the Oper field is interpreted as Update, Delete, or Reserved as specified in Section 9.3. For all other Message-Types, the Oper field MUST be sent as zero and ignored on receipt.
- o TLV-Type (12 bits): The Type of a TLV, that is the meaning and format of the Value part, are determined by the TLV-Type of the TLV. See Section 9.2.
- o TLV-Length (2 bytes): The length of the Value portion of the TLV in bytes as an unsigned integer.
- o Value (variable length): This is the value portion of the TLV whose size is given by TLV-Length. The value portion consists of fields, frequently using one of the basic data field types (see Section 7.2) and sub-TLVs (see Section 7.3).

## 7.2 Basic Data Fields

This section specifies the binary format of several standard basic data fields that are used within other data structures in this specification.

- o **STRING:** 0 to 255 octets. Will be encoded as a sub-TLV (see Section 7.3) to provide the length. The use of this data type in S-CUSP is to provide convenient labels for use by network operators in configuring and debugging their networks and interpreting S-CUSP messages. These labels will not normally be seen by subscribers. They are normally interpreted as ASCII [RFC20].
- o **MAC-Addr:** 6 octets. Ethernet MAC Address [RFC7042].
- o **IPv4-Address:** 8 octets. 4 octets of the IPv4 address value followed by a 4 octet address mask in the format XXX.XXX.XXX.XXX.
- o **IPv6-Address:** 20 octets. 16 octets of IPv6 address followed by a 4 octet integer  $n$  in the range of 0 to 128 which gives the address mask as the one's complement of  $2^{(128-n)} - 1$ .
- o **VLAN ID:** 2 octets. As follows [802.1Q]:

```

      0                               1
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5
+---+---+---+---+---+---+---+---+---+
| PRI |D|           VLAN-ID           |
+---+---+---+---+---+---+---+---+---+

```

**PRI:** Priority. Default value 7.

**D:** Drop Eligibility Indicator (DEI). Default value 0.

**VLAN-ID:** Unsigned integer in the range 1-4094. (0 and 4095 are not valid VLAN IDs [802.1Q].)

### 7.3 Sub-TLV Format and Sub-TLVs

In some cases, the Value portion of a TLV, as specified above, can contain one or more Sub-TLVs formatted as follows:

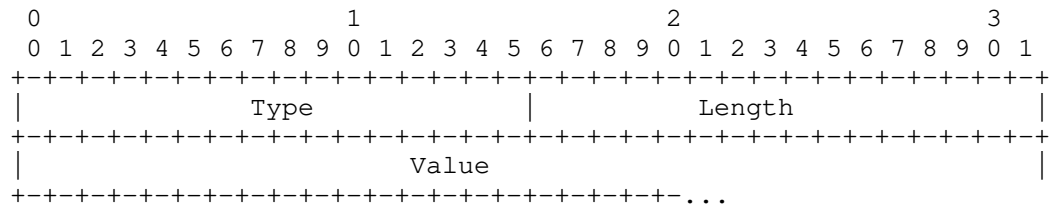


Figure 33: Sub-TLV Header

- o **Type (2 bytes):** The Type of a Sub-TLV, that is the meaning and format of the Value part, are determined by the Type of the TLV. Sub-TLV Types numbers have the same meaning regardless of the TLV Type of the TLV within which the sub-TLV occurs. See Section 9.4.
- o **Length (2 bytes):** The length of the Value portion of the sub-TLV in bytes as an unsigned integer.
- o **Value (variable length):** This is the value portion of the sub-TLV whose size is given by Length.

The sub-TLVs currently specified are defined in the following subsections.

#### 7.3.1 Name sub-TLVs

This document defines the following name sub-TLVs that are used to carry the name of the corresponding object. The length of each of these sub-TLV is variable from 1 to 255 octets. The value is of type STRING padded with zeros octets to 4-octet alignment.

Type	Sub-TLV Name	Meaning
1	VRF-Name	The name of a VRF
2	Ingress-QoS-Profile	The name of an ingress QoS profile
3	Egress-QoS-Profile	The name of an egress QoS profile
4	User-ACL-Policy	The name of an ACL policy
5	Multicast-ProfileV4	The name of an IPv4 multicast profile
6	Multicast-ProfileV6	The name of an IPv6 multicast profile
7	NAT-Instance	The name of a NAT instance
8	Pool-Name	The name of an address pool



### 7.3.2 Ingress-CAR sub-TLV

The Ingress-CAR sub-TLV indicates the authorized upstream Committed Access Rate (CAR) parameters. The sub-TLV type of the Ingress-CAR sub-TLV is 9 and the sub-TLV length is 16. The format is as shown in Figure 34.

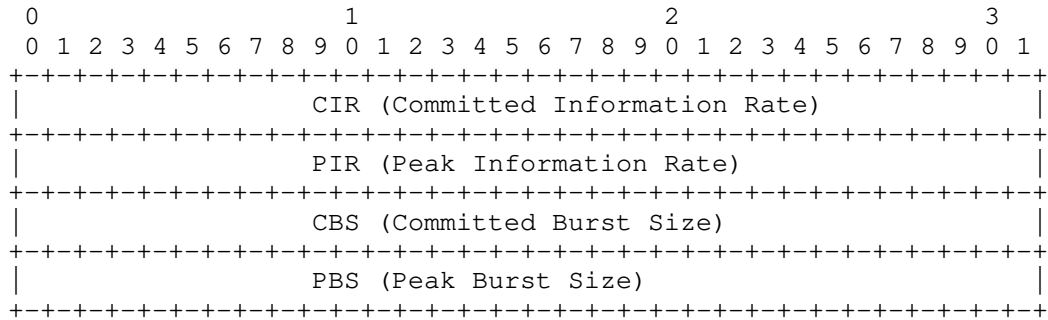


Figure 34: Ingress-CAR sub-TLV

Where:

CIR (4 bytes): Guaranteed rate in bits/second.

PIR (4 bytes): Burst rate in bits/second.

CBS (4 bytes): The token bucket in bytes.

PBS (4 bytes): Burst token bucket in bytes.

These fields are unsigned integers. More details about CIR, PIR, CBS, and PBS can be found in [RFC2698].

### 7.3.3 Egress-CAR sub-TLV

The Egress-CAR sub-TLV indicates the authorized downstream Committed Access Rate (CAR) parameters. The sub-TLV type of the Egress-CAR sub-TLV is 10 and its sub-TLV length is 16 octets. The format of the value part is as defined below.

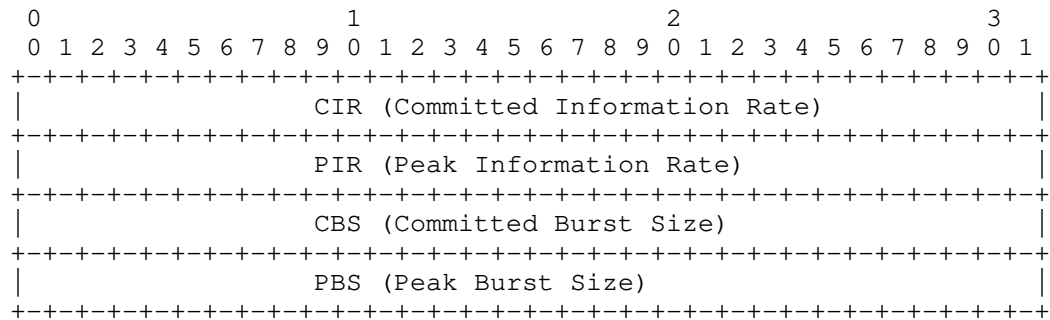


Figure 35: Egress-CAR sub-TLV

Where:

CIR (4 bytes): Guaranteed rate in bits/second.

PIR (4 bytes): Burst rate in bits/second.

CBS (4 bytes): The token bucket in bytes.

PBS (4 bytes): Burst token bucket in bytes.

These fields are unsigned integers. More details about CIR, PIR, CBS, and PBS can be found in [RFC2698].

#### 7.3.4 If-Desc sub-TLV

The If-Desc sub-TLV is defined to designate an interface. It is an optional sub-TLV that may be carried in those TLVs that have an "if-index" or "out-if-index" field. The If-Desc sub-TLV is used as a local unique identifier within a BNG.

The sub-TLV type is 11 and the sub-TLV length is 12 octets. The format depends on the If-Type. The format of the value part is as follows:

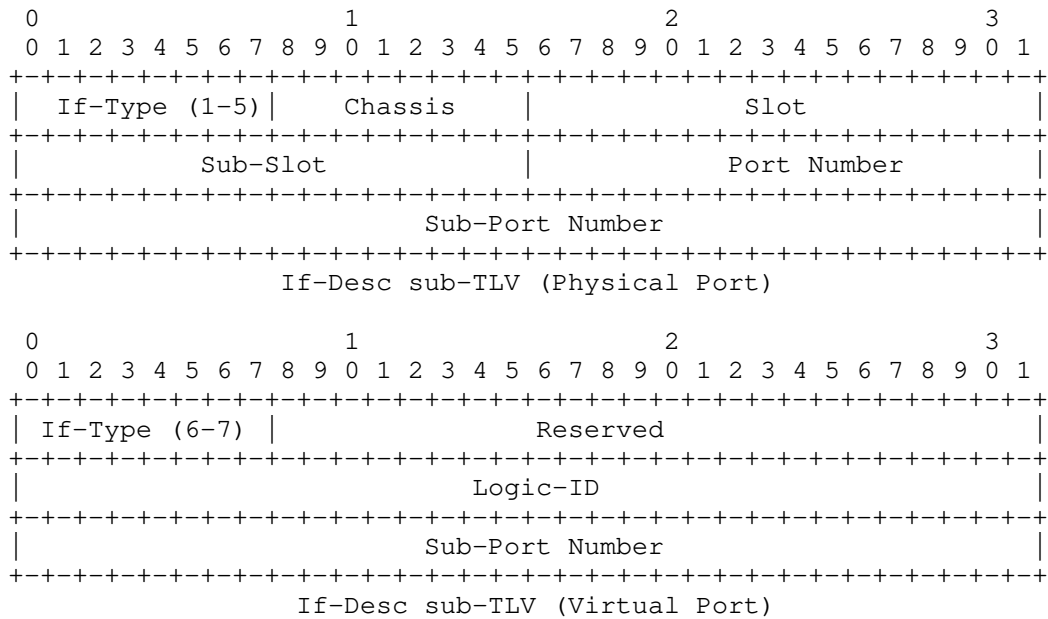


Figure 36: If-Desc sub-TLV Formats

Where:

If-Type: 8 bits in length, indicates the type of an interface. Following types are defined in this document:

- 0: Reserved
- 1: Fast Ethernet (FE)
- 2: GE
- 3: 10GE
- 4: 100GE
- 5: Eth-Trunk
- 6: Tunnel
- 7: VE
- 8-255: Reserved.

Chassis (8 bits): Identifies the chassis that the interface belongs to.

Slot (16 bits): Identifies the slot that the interface belongs to.

Sub-slot (16 bits): Identifies the sub-slot the interface belongs to.

Port Number (16 bits): An identifier of a physical port/interface (e.g., If-Type: 1-5). It is locally significant within the slot/sub-slot.

Sub-port Number (32 bits): An identifier of the sub-port. Locally significant within its "parent" port (physical or virtual).

Logic-ID (32 bits): An identifier of a virtual interface (e.g., If-Type: 6-7)

### 7.3.5 IPv6 Address List sub-TLV

The IPv6 Address List sub-TLV is used to convey one or more IPv6 addresses. It is carried in the IPv6 Subscriber TLV. The sub-TLV type is 12, the sub-TLV length is variable.

The format of IPv6 Addresses sub-TLV is as follows:

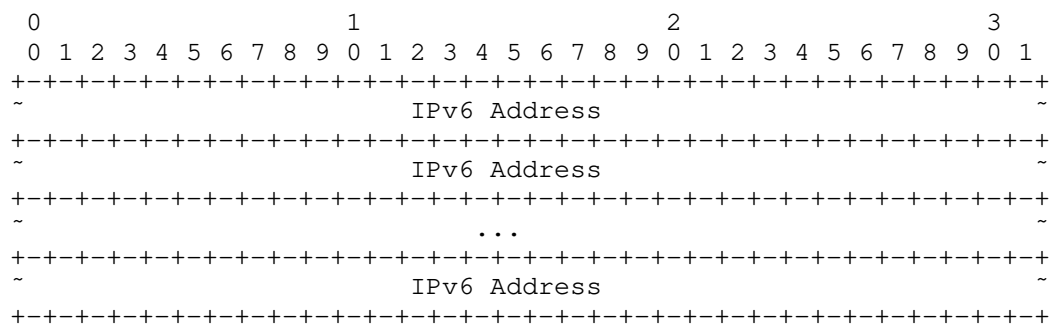


Figure 37: IPv6 Address List sub-TLV

Where:

IP Address (IPv6-Address): Each IP Address is an IP-Address type, carries an IPv6 address.

### 7.3.6 Vendor sub-TLV

The Vendor sub-TLV is intended to be used inside the value portion of the Vendor TLV (Section 7.13). It provides a Sub-Type that effectively extends the sub-TLV type in the sub-TLV header and provides for versioning of vendor sub-TLVs.

The value part of the Vendor sub-TLV is formatted as follows:

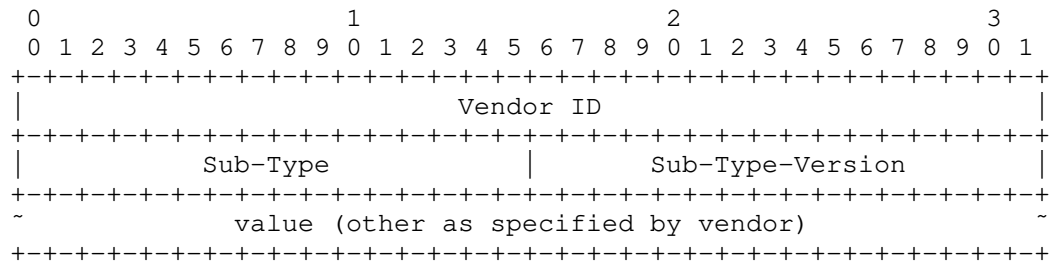


Figure 38: Vendor sub-TLV

Where:

The sub-TLV type: 13.

The sub-TLV length: variable.

Vendor-ID (4 bytes): Vendor ID as defined in RADIUS [RFC2865].

Sub-Type (2 bytes): Used by the Vendor to distinguish multiple different sub-TLVs.

Sub-Type-Version (2 bytes): Used by the Vendor to distinguish different versions of a Vendor Defined sub-TLV sub-Type.

value: as specified by the vendor.

Since Vendor code will be handling the sub-TLV after the Vendor ID field is recognized, the remainder of the sub-TLV can be organized however the vendor wants. But it is desirable for a vendor to be able to define multiple different vendor sub-TLVs and to keep track of different versions of its vendor defined sub-TLVs. Thus, it is RECOMMENDED that the vendor assign a Sub-Type value for each of that vendor's sub-TLVs that is different from other Sub-Type values that vendor has used. Also, when modifying a vendor defined sub-TLV in a way potentially incompatible with a previous definition, the vendor SHOULD increase the value it is using in the Sub-Type-Version field.

#### 7.4 The Hello TLV

The Hello TLV is defined to be carried in the Hello message for version and capabilities negotiation. It indicates the S-CUSP sub-version and capabilities supported. The format of Hello TLV is as follows:

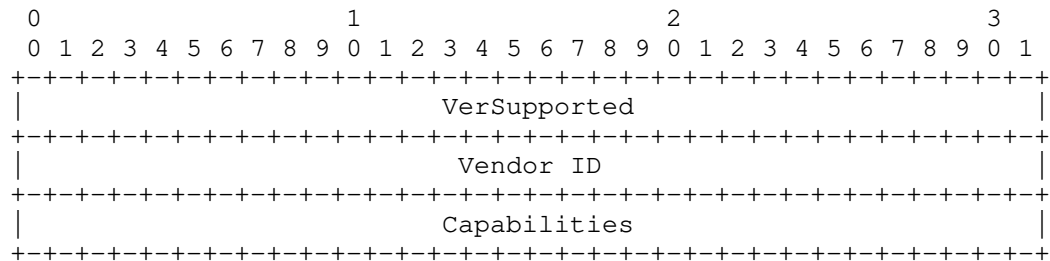


Figure 39: Hello TLV

Where:

The TLV type is 100.

The TLV length is 12 octets.

The value field consists of three parts:

**VerSupported:** 32 bits in length. This is a bit map of the Sub-Versions of the S-CUSP protocol that the sender supports. This document specifies Sub-Version zero of Major Version 1, that is, Version 1.0. The VerSupported field **MUST** be non-zero. The VerSupported bits are numbered from 0 as the most significant bit. Bit 0 indicates support of Sub-Version zero, bit 1 indicates support of Sub-Version one, etc.

**Vendor-ID:** 4 bytes in length. Vendor ID, as defined in RADIUS [RFC2865].

**Capabilities:** 32 bits in length. Flags that indicate the support of particular capabilities by the sender of the Hello. No Capabilities are defined in this document and so implementations will set this field to zero. The Capabilities field of the Hello TLV **MUST** be checked before any other TLVs in the Hello because capabilities defined in the future might extend existing TLVs or permit new TLVs.

After the exchange of Hello messages, the CP and UP each perform a logical AND of the Sub-Version supported by the CP and the UP and separately perform a logical AND of the Capabilities bits fields for the CP and the UP.

For example, if one side supports Sub-Versions 1, 3, 4, and 5 (VerSupported = 0x5C000000) and the other side supports 2, 3, and 4 (VerSupported = 0x38000000) then 3 and 4 are the Sub-Versions in common and 4 is the highest Sub-Version supported by both sides. So Sub-Version 4 is used for the session that has been negotiated.

The result of the logical AND of the Capabilities bits will show what additional capabilities both sides support. If this result is zero, there are no such capabilities so none can be used during the session. If this result is non-zero, it shows the additional capabilities that can be used during the session. The CP and the UP MUST NOT use a capability unless both advertise support.

## 7.5 The Keep Alive 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								
Keepalive										DeadTimer										Reserved																			

Figure 40: Keep Alive TLV

The TLV type: 102.

The value of length: 4 octets.

**Keepalive (8 bits):** Indicates the maximum period of time (in seconds) between two consecutive S-CUSP messages sent by the sender of the message containing this TLV as an unsigned integer. The minimum value for the Keepalive is 1 second. When set to 0, once the session is established, no further Keepalive messages are sent to the remote peer. A RECOMMENDED value for the Keepalive frequency is 30 seconds.

**DeadTimer** (8 bits in length): Specifies the amount of time as an unsigned integer number of seconds after the expiration of which

the S-CUSP peer can declare the session with the sender of the Hello message to be down if no S-CUSP message has been received. The DeadTimer SHOULD be set to 0 and MUST be ignored if the Keepalive is set to 0. A RECOMMENDED value for the DeadTimer is 4 times the value of the Keepalive.

The Reserved bits MUST be sent as zero and ignored on receipt.

## 7.6 The Error Information TLV

The Error Information TLV is a common TLV that can be used in many Response (e.g., Update\_Response message) and ACK messages (e.g., Addr\_Allocation\_Ack message, etc.). It is used to convey the information about an error in the received S-CUSP message. The format of the Error Information TLV is as follows:

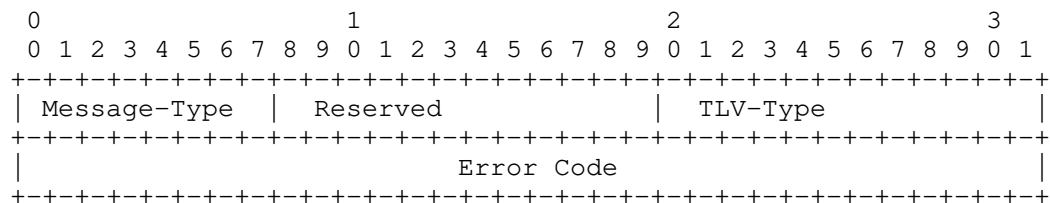


Figure 41: Error Information TLV

Where:

The TLV type: 101.

The value of length: 8 octets.

Message-Type(1 byte): This parameter is the message type of the message containing an error.

Reserved (1 byte): MUST be sent as zero and ignored on receipt.

TLV-Type (2 bytes): Indicates which TLV caused the error.

Error Code: 4 bytes in length. Indicate the specific Error Code (see Section 9.5).

## 7.7 BAS Function TLV

The BAS Function TLV is used by a CP to control the access mode, authentication methods, and other related functions of an interface



on a UP.

The format of the BAS Function TLV value part is as follows:

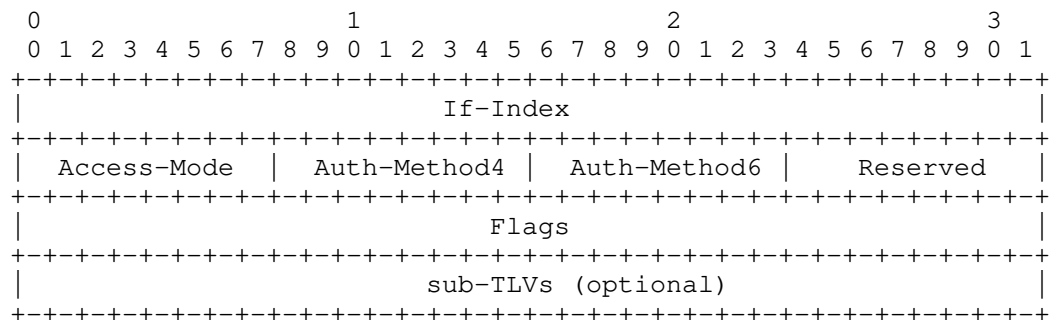


Figure 42: BAS Function TLV

Where:

The TLV type: 1.

The value of length: variable.

If-Index: 4 bytes in length, a unique identifier of an interface of a BNG.

Access-Mode: 1 byte in length, indicates the access mode of the interface. This document defines following values:

- 0: Layer 2 subscriber;
- 1: Layer 3 subscriber;
- 2: Layer 2 leased line;
- 3: Layer 3 leased line;
- 4-255: Reserved.

Auth-Method4: 1 byte in length, for IPv4 scenario, it indicates the authentication on this interface; this field is defined as a bitmap, this document defines following values (other bits are reserved and MUST be sent as zero and ignored on receipt):

- 0x1: PPPoE authentication;
- 0x2: DOT1X authentication;
- 0x4: Web authentication;
- 0x8: Web fast authentication;
- 0x10: Binding authentication.

Auth-Method6: 1 byte in length, for IPv6 scenario, it indicates the authentication on this interface; this field is defined as a bitmap, this document defines following values (other bits are

reserved and MUST be sent as zero and ignored on receipt):

0x1: PPPoE authentication;  
 0x2: DOT1X authentication;  
 0x4: Web authentication;  
 0x8: Web fast authentication;  
 0x10: Binding authentication;

sub-TLVs:

The IF-Desc sub-TLV can be carried.  
 If-Desc sub-TLV: carries the interface information.

The flags field is defined as follows:

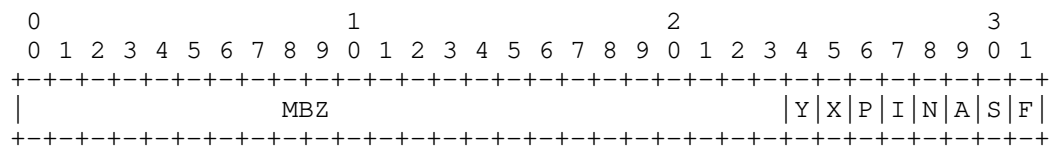


Figure 43: Interface Flags

Where:

F (IPv4 Trigger) bit: Indicates whether IPv4 packets can trigger a subscriber to go online. 1: enabled, 0: disabled.

S (IPv6 Trigger) bit: Indicates whether IPv6 packets can trigger a subscriber to go online. 1: enabled, 0: disabled.

A (ARP Trigger) bit: Indicates whether ARP packets can trigger a subscriber to go online. 1: enabled, 0: disabled.

N (ND Trigger) bit: Indicates whether ND packets can trigger a subscriber to go online. 1: enabled, 0: disabled.

I (IPoE-Flow-Check): Used for UP detection. IPoE 1: Enable traffic detection. 0: Disable traffic detection.

P (PPP-Flow-Check) bit: Used for UP detection. PPP 1: Enable traffic detection. 0: Disable traffic detection.

X (ARP-Proxy) bit: 1: The interface is enabled with ARP proxy and can process ARP requests across different Port+VLANs. 0: The ARP proxy is not enabled on the interface, and only the ARP requests of the same Port+VLAN are processed.

Y (ND-Proxy) bit: 1: The interface is enabled with ND proxy and can process ND requests across different Port+VLANs. 0: The ND proxy is not enabled on the interface, and only the ND requests of the same Port+VLAN are processed.

MBZ: Reserved bits that MUST be sent as zero and ignored on receipt.

## 7.8 Routing TLVs

Normally, after an S-CUSP session is established between a UP and a CP, the CP will allocate one or more blocks of IP addresses to the UP. Those IP addresses will be allocated to subscribers who will dial-up to the UP. In order to make sure that other nodes within the network learn how to reach those IP addresses, the CP needs to install one or more routes that can reach those IP addresses on the UP and notify the UP to advertise the routes to the network.

The Routing TLVs are used by a CP to notify a UP of the network routing. They can be carried in the Update\_Request message and Sync\_Data message.

### 7.8.1 IPv4 Routing TLV

The IPv4 Routing TLV is used to carry IPv4 network routing related information.

The format of the TLV value part is as below:

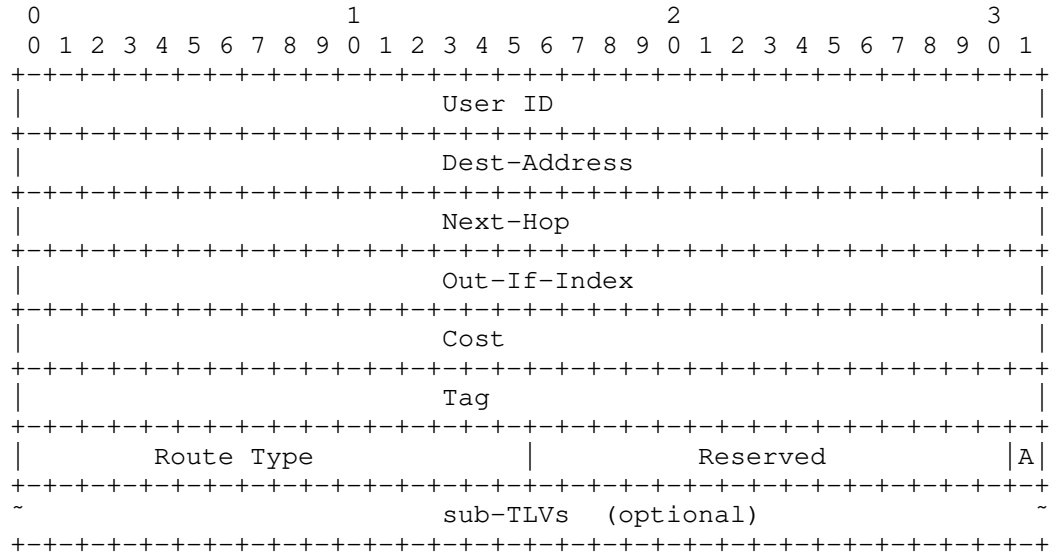


Figure 44: IPv4 Routing TLV

Where:

The TLV Type: 7

The TLV Length: Variable

User-ID: 4 bytes in length. Carries the user identifier. This field is filled with all Fs when a non-user route is delivered to the UP.

Dest-Address (IPv4-Address type): Identifies the destination address.

Next-Hop: (IPv4-Address type): Identifies the next hop address.

Out-If-Index (4 bytes): Indicates the interface index.

Cost (4 bytes): The cost value of the route.

Tag (4 bytes): The tag value of the route.

Route-Type (2 bytes): Indicates the route type. The options are as follows:

- 0: User host route
- 1: Radius authorization FrameRoute
- 2: Network segment route
- 3: Gateway route
- 4: Radius authorized IP route
- 5: L2TP LNS side user route

Advertise-Flag: 1 bit. Indicates whether the route should be advertised by the UP. Following flags are defined:

- 0: Not advertised,
- 1: advertised.

sub-TLVs: The VRF-Name and/or If-Desc sub-TLVs can be carried.

VRF-Name sub-TLV: indicates which VRF the route belongs to.

If-Desc sub-TLV: carries the interface information.

The Reserved field MUST be sent as zero and ignored on receipt.

### 7.8.2 IPv6 Routing TLV

The IPv6 Routing TLV is used to carry IPv6 network routing information.

The format of this TLV is as follows:

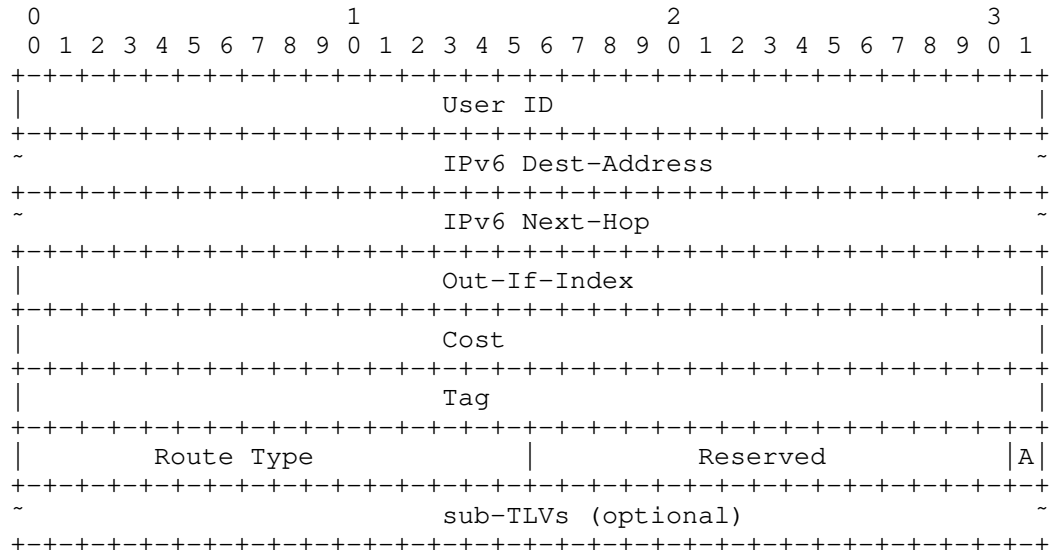


Figure 45: IPv6 Routing TLV

Where:

The TLV Type: 7

The TLV Length: Variable

User-ID: 4 bytes in length. Carries the user identifier. This field is filled with all Fs when a non-user route is delivered to the UP.

IPv6 Dest-Address (IPv6-Address type): Identifies the destination address.

IPv6 Next-Hop: (IPv6-Address type): Identifies the next hop address.

Out-If-Index (4 bytes): Indicates the interface index.

Cost (4 bytes): The cost value of the route.

Tag (4 bytes): The tag value of the route.

Route-Type: (2 bytes): Indicates the route type. The options are as follows:

- 0: User host route
- 1: Radius authorization FrameRoute
- 2: Network segment route
- 3: Gateway route
- 4: Radius authorized IP route
- 5: L2TP LNS side user route

Advertise-Flag: 1 bit. Indicates whether the route should be advertised by the UP. Following flags are defined:

- 0: Not advertised,
- 1: advertised.

sub-TLVs: If-Desc and VRF-Name sub-TLVs can be carried.

VRF-Name sub-TLV: Indicates the VRF to which the subscriber belongs.

If-Desc sub TLV: carries the interface information.

The Reserved field MUST be sent as zero and ignored on receipt.

## 7.9 Subscriber TLVs

The Subscriber TLVs are defined for a CP to send the basic information about a user to a UP.

### 7.9.1 Basic Subscriber TLV

The Basic Subscriber TLV is used to carry the basic common information for all kinds of access subscribers. It is carried in an Update\_Request message.

The format of the Basic Subscriber TLV value part is as follows:

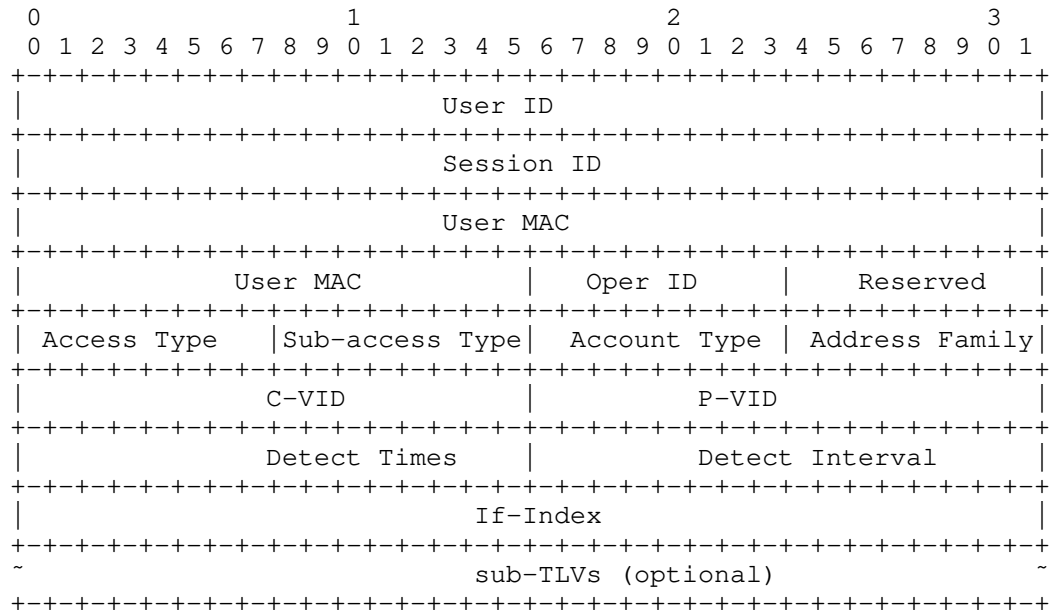


Figure 46: Basic Subscriber TLV

Where:

The TLV Type: 2.

The TLV Length: variable in length.

User-ID (4 bytes): The identifier of a subscriber.

Session-ID (4 bytes): Session ID of a PPPoE subscriber. Zero means non-PPPoE subscriber.

User-Mac (MAC-Addr type): The MAC Address of a subscriber.

Oper-ID (1 byte): Indicates the ID of an operation performed by a user. This field is carried in the response from the UP.

Reserved (1 byte): MUST be sent as zero and ignored on receipt.

Access-Type (1 byte):

- 1: PPP access (PPP [RFC1661])
- 2: PPP over Ethernet over ATM access (PPPoEoA)
- 3: PPP over ATM access (PPPoA [RFC3336])
- 4: PPP over Ethernet access (PPPoE [RFC2516])
- 5: PPPoE over VLAN access (PPPoEoVLAN [RFC2516])
- 6: PPP over LNS access (PPPoLNS)
- 7: IP over Ethernet DHCP access (IPoE\_DHCP)
- 8: IP over Ethernet EAP authentication access (IPoE\_EAP)
- 9: IP over Ethernet Layer 3 access (IPoE\_L3)
- 10: IP over Ethernet Layer 2 Static access (IPoE\_L2\_STATIC)
- 11: Layer 2 Leased Line access (L2\_Leased\_Line)
- 12: Layer 2 VPN Leased Line access (L2VPN\_Leased\_Line)
- 13: Layer 3 Leased Line access (L3\_Leased\_Line)
- 14: Layer 2 Leased line Sub-User access  
(L2\_Leased\_Line\_SUB\_USER)
- 15: L2TP LAC tunnel access (L2TP\_LAC)
- 16: L2TP LNS tunnel access (L2TP\_LNS)

Sub-Access-Type (1 byte): Indicates whether PPP termination or PPP relay is used.

- 0: Reserved
- 1: PPP Relay (for LAC)
- 2: PPP termination (for LNS)

Account-Type(1 byte):

- 0: Collects statistics on IPv4 and IPv6 traffic of terminals independently.
- 1: Collects statistics on IPv4 and IPv6 traffic of terminals.

Address Family (1 byte)

- 1: IPv4
- 2: IPv6
- 3: dual stack

C-VID (VLAN-ID): Indicates the inner VLAN ID. The value 0 indicates that the VLAN ID is invalid. The default value of PRI is 7, the value of DEI is 0, and the value of VID is 1~4094. The PRI value can also be obtained by parsing terminal packets.

P-VID (VLAN-ID): Indicates the outer VLAN ID. The value 0 indicates that the VLAN ID is invalid. The format is the same as that for C-VID.

Detect-Times (2 bytes): Number of detection timeout times. The value 0 indicates that no detection is performed.

Detect-Interval (2 bytes): Detection interval in seconds.



If-Index (4 bytes): Interface index.

Sub-TLVs: VRF-Name sub-TLV and If-Desc sub-TLV can be carried.

VRF-Name sub-TLV: Indicates the VRF to which the subscriber belongs.

If-Desc sub-TLV: carries the interface information.

The Reserved field MUST be sent as zero and ignored on receipt.

### 7.9.2 PPP Subscriber TLV

The PPP Subscriber TLV is defined to carry PPP information of a User from a CP to a UP. It will be carried in an Update\_Request message when PPPoE or L2TP access is used.

The format of the TLV value part is as follows:

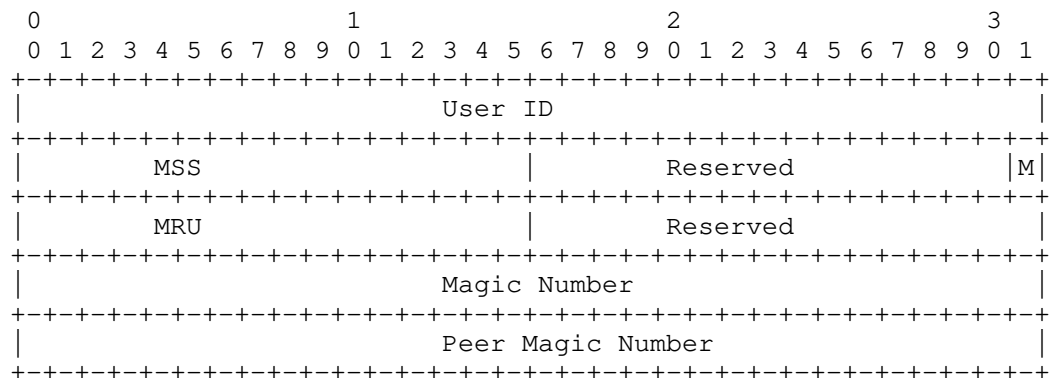


Figure 47: PPP Subscriber TLV

Where:

The TLV type: 3.

The TLV length: 12 octets.

User-ID (4 bytes): The identifier of a subscriber.

MSS-Value (2 bytes): Indicates the MSS value (in bytes).

MSS-Enable (M) (1 bit): Indicates whether the MSS is enabled.

0: Disabled.

1: Enabled.

MRU (2 bytes): PPPoE local MRU (in bytes).

Magic-Number (4 bytes): Local magic number in PPP negotiation packets.

Peer-Magic-Number (4 bytes): Remote peer magic number.

The Reserved fields MUST be sent as zero and ignored on receipt.

### 7.9.3 IPv4 Subscriber TLV

The IPv4 Subscriber TLV is defined to carry IPv4 related information for a BNG user. It will be carried in an Update\_Request message when IPv4 IPE, or PPPoE access is used.

The format of the TLV value part is as follows:

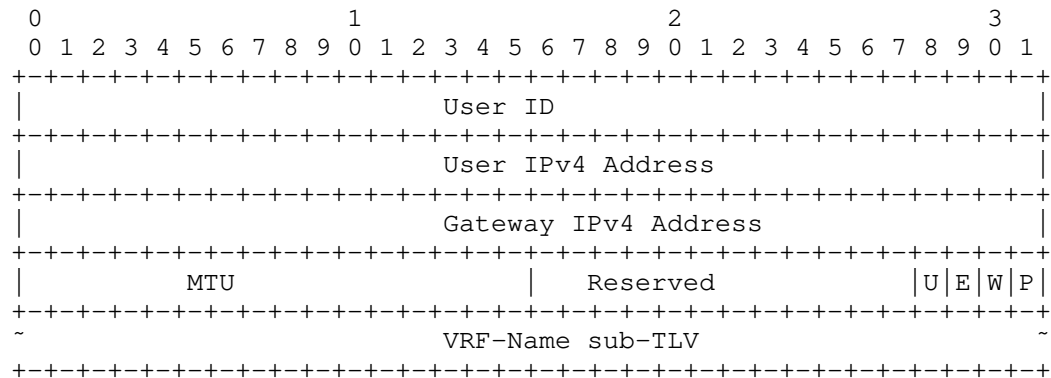


Figure 48: IPv4 Subscriber TLV

Where:

The TLV type: 4.

The TLV length: variable.

User-ID (4 bytes): The identifier of a subscriber.

User-IPv4 (IPv4-Address): The IPv4 address of the subscriber.

Gateway-IPv4 (IPv4-Address): The gateway address of the subscriber.

Portal Force (P) (1 bit ): Push advertisement, 0: off, 1: on.

Web-Force (W) (1 bit): Push IPv4 Web. 0: off, 1: on.

Echo-Enable (E) (1 bit): UP returns ARP Req or PPP Echo. 0: off, 1: on.

IPv4-URPF (U) (1 bit): User Unicast Reverse Path Forwarding (URPF) flag. 0: off, 1: on.

MTU 2 bytes MTU value. The default value is 1500.

VRF-Name Sub-TLV: Indicates the subscriber belongs to which VRF.

The Reserved field MUST be sent as zero and ignored on receipt.

#### 7.9.4 IPv6 Subscriber TLV

The IPv6 Subscriber TLV is defined to carry IPv6 related information for a BNG user. It will be carried in an Update\_Request message when IPv6 IPE, or PPPoE access is used.

The format of the TLV value part is as follows:

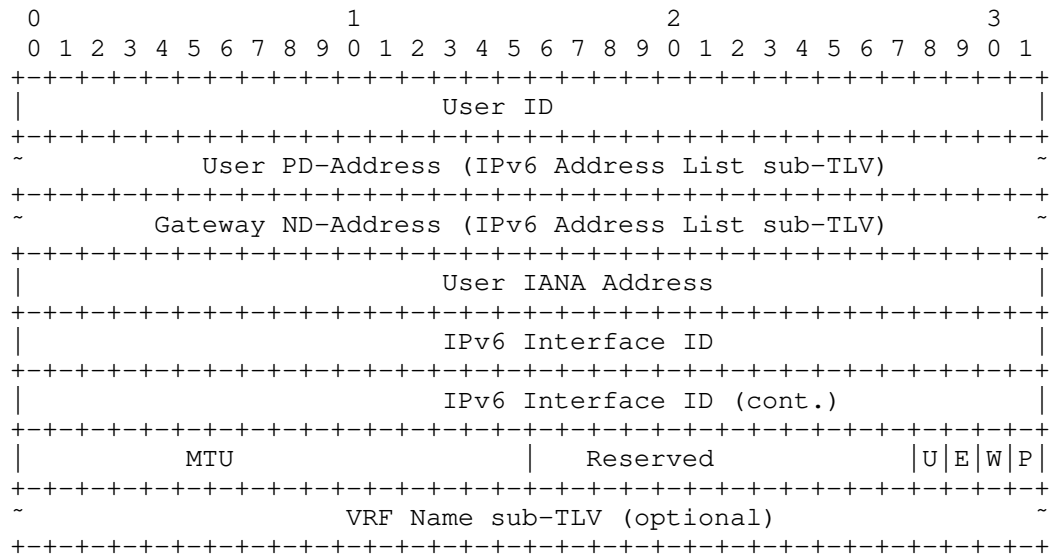


Figure 49: IPv6 Subscriber TLV

Where:

The TLV type: 5.

The TLV length: variable.

User-ID (4 bytes): The identifier of a subscriber.

User PD-Addresses (IPv6 Address List): Carries a list of Prefix Delegation (PD) addresses of the subscriber.

User ND-Addresses (IPv6 Address List): Carries a list of Neighbor Discovery (ND) addresses of the subscriber.

User IANA-Address (IPv6-Address): The IANA address of the subscriber.

IPv6 Interface ID (8 bytes): The identifier of an IPv6 interface.

Portal Force 1 bit (P): Push advertisement, 0: off, 1: on.

Web-Force 1 bit (W): Push IPv6 Web, 0: off, 1: on.

Echo-Enable 1 bit (E): The UP returns ARP Req or PPP Echo. 0: off; 1: on.

IPv6-URPF 1 bit (U): User Reverse Path Forwarding (URPF) flag, 0: off; 1: on.

MTU (2 bytes): The MTU value. The default value is 1500.

VRF-Name Sub-TLV: Indicates the VRF to which the subscriber belongs.

The Reserved field MUST be sent as zero and ignored on receipt.

#### 7.9.5 IPv4 Static Subscriber Detect TLV

The IPv4 Static Subscriber Detect TLV is defined to carry IPv4 related information for a static access subscriber. It will be carried in an Update\_Request message when IPv4 static access on a UP needs to be enabled.

The format of the TLV value part is as follows:

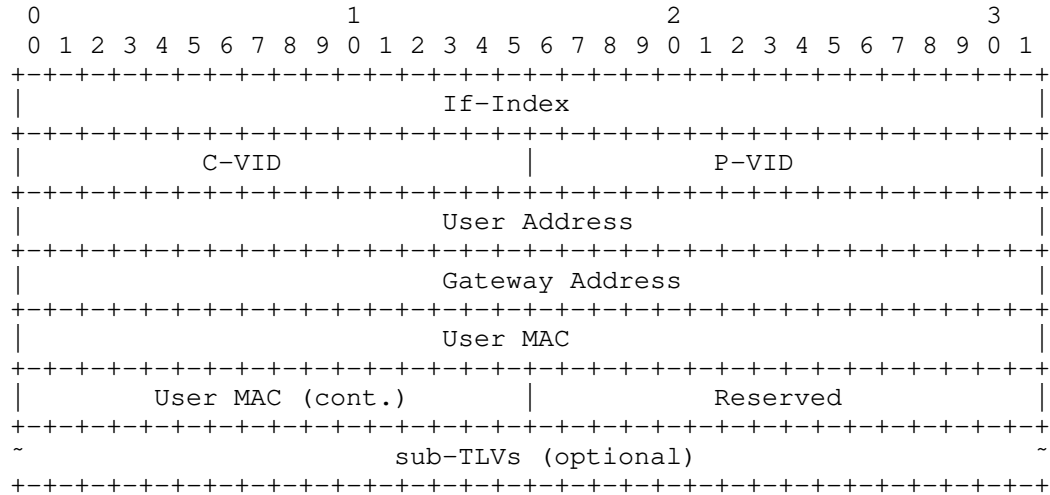


Figure 50: IPv4 Static Subscriber TLV

Where:

The TLV type: 6.

The TLV length: variable.

If-Index (4 bytes): The interface index of the interface from which the subscriber will dial-up.

C-VID (VLAN-ID): Indicates the inner VLAN ID. The value 0 indicates that the VLAN ID is invalid. A valid value is 1~4094.

P-VID (VLAN-ID): Indicates the outer VLAN ID. The value 0 indicates that the VLAN ID is invalid. The format is the same as that of the C-VID. A valid value is 1~4094. For a single-layer VLAN, set this parameter to PeVid.

User Address (IPv4-Addr): The user's IPv4 address.

Gateway Address (IPv4-Addr): The gateway's IPv4 Address.

User-MAC (MAC-Addr type): The MAC address of the subscriber.

Sub-TLVs: VRF-Name and If-Desc sub-TLVs may be carried.

VRF-Name sub-TLV: Indicates the VEF to which the subscriber belongs.

If-Desc sub-TLV: Carries the interface information.

The Reserved field MUST be sent as zero and ignored on receipt.

#### 7.9.6 IPv6 Static Subscriber Detect TLV

The IPv6 Static Subscriber Detect TLV is defined to carry IPv6 related information for a static access subscriber. It will be carried in an Update\_Request message when needed to enable IPv6 static subscriber detection on a UP.

The format of the TLV value part is as follows:

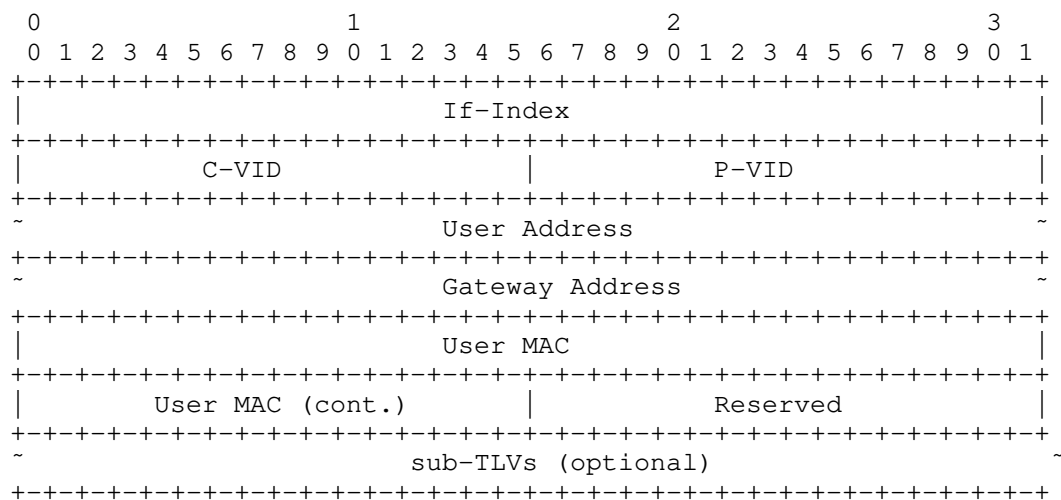


Figure 51: IPv6 Static Subscriber Detect TLV

Where:

The TLV type: 6.

The TLV length: variable.

If-Index (4 bytes): The interface index of the interface from which the subscriber will dial-up.

C-VID (VLAN-ID): Indicates the inner VLAN ID. The value 0 indicates that the VLAN ID is invalid. A valid value is 1~4094.

P-VID (VLAN-ID): Indicates the outer VLAN ID. The value 0 indicates that the VLAN ID is invalid. The format is the same as that the of C-VID. A valid value is 1~4094. For a single-layer VLAN, set this parameter to PeVid.

User Address (IPv6-Address type): The subscriber's IPv6 address.

Gateway Address (IPv6-Address type): The gateway's IPv6 Address.

User-MAC (MAC-Addr type): The MAC address of the subscriber.

sub-TLVs: VRF-Name and If-Desc sub-TLVs may be carried

VRF-Name Sub-TLV: Indicates the VRF to which the subscriber belongs.

If-Desc sub-TLV: Carries the interface information.

The Reserved field MUST be sent as zero and ignored on receipt.

#### 7.9.7 L2TP-LAC Subscriber TLV

The L2TP-LAC Subscriber TLV is defined to carry the related information for a L2TP LAC access subscriber. It will be carried in an Update\_Request message when L2TP LAC access is used.

The format of the TLV value part is as follows:

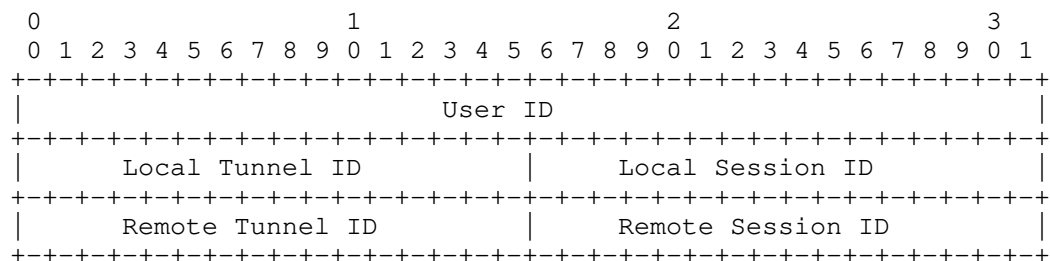


Figure 52: L2TP-LAC Subscriber TLV

Where:

The TLV type: 11.

The TLV length: 12 octets.

User-ID (4 bytes): The identifier of a user/subscriber.

Local-Tunnel-ID (2 bytes): The local ID of the L2TP tunnel.

Local-Session-ID (2 bytes): The local session ID with the L2TP tunnel.

Remote-Tunnel-ID (2 bytes): The remote ID of the L2TP tunnel.

Remote-Session-ID (2 bytes): The remote session ID of the L2TP tunnel.

#### 7.9.8 L2TP-LNS Subscriber TLV

The L2TP-LNS Subscriber TLV is defined to carry the related information for a L2TP LNS access subscriber. It will be carried in an Update\_Request message when L2TP LNS access is used.

The format of the TLV value part is as follows:

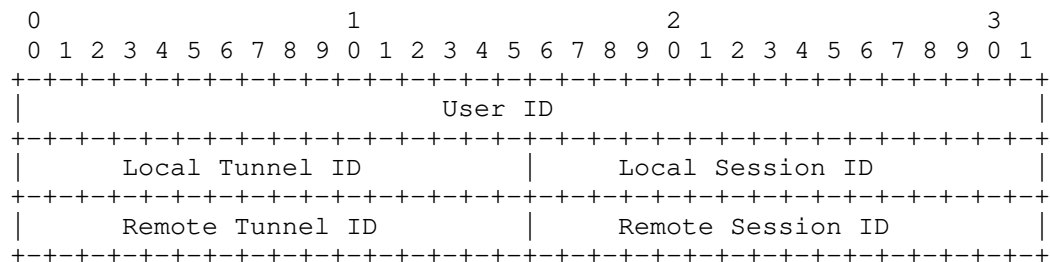


Figure 53: L2TP-LNS Subscriber TLV

Where:

The TLV type: 12.

The TLV length: 12 octets.

User-ID (4 bytes): The identifier of a user/subscriber.

Local-Tunnel-ID (2 bytes): The local ID of the L2TP tunnel.

Local-Session-ID (2 bytes): The local session ID with the L2TP tunnel.

Remote-Tunnel-ID (2 bytes): The remote ID of the L2TP tunnel.

Remote-Session-ID (2 bytes): The remote session ID of the L2TP tunnel.

#### 7.9.9 L2TP-LAC Tunnel TLV

The L2TP-LAC Tunnel TLV is defined to carry the L2TP LAC tunnel related information. It will be carried in the Update\_Request message when L2TP LAC access is used.



The format of the TLV value part is as follows:

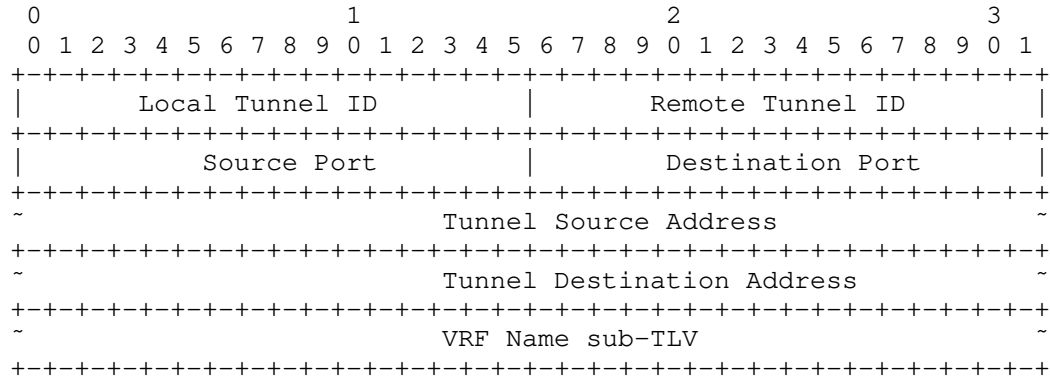


Figure 54: L2TP-LAC Tunnel TLV

Where:

The TLV type: 13.

The TLV length: variable.

Local-Tunnel-ID (2 bytes): The local ID of the L2TP tunnel.

Remote-Tunnel-ID (2 bytes): The remote ID of the L2TP tunnel.

Source-Port (2 bytes): The source UDP port number of an L2TP subscriber.

Dest-Port (2 bytes): The destination UDP port number of an L2TP subscriber.

Source-IP (IPv4/v6): The source IP address of the tunnel.

Dest-IP (IPv4/v6): The destination IP address of the tunnel.

VRF-Name Sub-TLV: The VRF name to which the L2TP subscriber tunnel belongs.

#### 7.9.10 L2TP-LNS Tunnel TLV

The L2TP-LNS Tunnel TLV is defined to carry the L2TP LNS tunnel related information. It will be carried in the Update\_Request message when L2TP LNS access is used.

The format of the TLV value part is as follows:

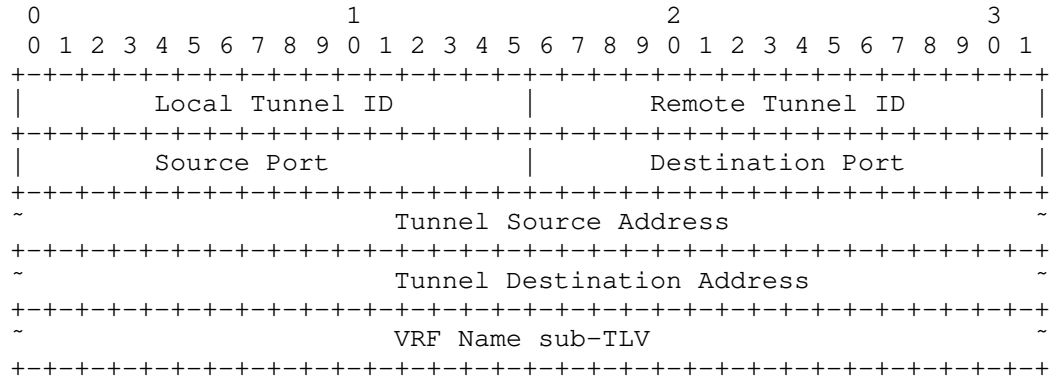


Figure 55: L2TP-LNS Tunnel TLV

Where:

The TLV type: 14.

The TLV length: variable.

Local-Tunnel-ID (2 bytes): The local ID of the L2TP tunnel.

Remote-Tunnel-ID (2 bytes): The remote ID of the L2TP tunnel.

Source-Port (2 bytes): The source UDP port number of an L2TP subscriber.

Dest-Port (2 bytes): The destination UDP port number of an L2TP subscriber.

Source-IP (IPv4/v6): The source IP address of the tunnel.

Dest-IP (IPv4/v6): The destination IP address of the tunnel.

VRF-Name Sub-TLV: The VRF name to which the L2TP subscriber tunnel belongs.

#### 7.9.11 Update Response TLV

The Update Response TLV is used to return the operation result of an update request. It is carried in the Update\_Response message as a response to the Update\_Request message.

The format of Update Response TLV is as follows:

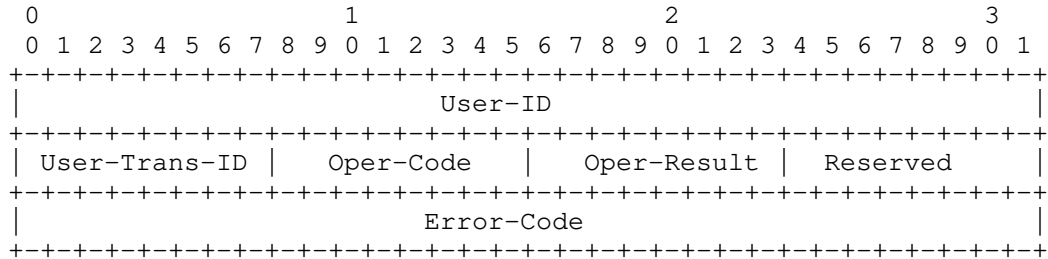


Figure 56: Update Response TLV

Where:

The TLV type: 302.

The TLV length: 12.

User-ID (4 bytes): A unique identifier of an user/subscriber.

User-Trans-ID (1 byte): In the case of dual-stack access or when modifying a session, User-Trans-ID is used to identify a user operation transaction. It is used by the CP to correlate a response to a specific request.

Oper-Code (1 byte): Operation code. 1: update, 2: delete.

Oper-Result (1 byte): Operation Result. 0: Success, Others: Failure.

Error-Code (4 bytes): Operation failure cause code. for details, see Section 9.5.

The Reserved field MUST be sent as zero and ignored on receipt.

#### 7.9.12 Subscriber Policy TLV

The Subscriber Policy TLV is used to carry the policies that will be applied to a subscriber. It is carried in the Update\_Request message.

The format of the TLV value part is as follows:

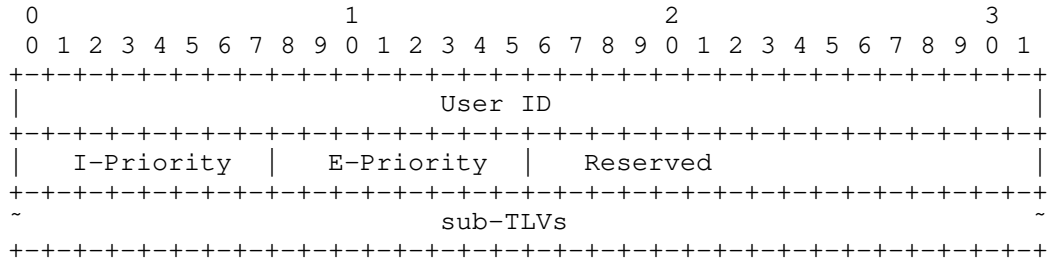


Figure 57: User QoS Policy Information TLV

Where:

The TLV type: 6.

The TLV length: variable.

User-ID (4 bytes): The identifier of a user/subscriber.

Ingress-Priority (1 byte): Indicates the upstream priority. The value range is 0~7.

Egress-Priority (1 byte): Indicates the downstream priority. The value range is 0~7.

sub-TLVs: The sub-TLVs that are present can occur in any order.

Ingress-CAR sub-TLV: Upstream CAR.

Egress-CAR sub-TLV: Downstream CAR.

Ingress-QoS-Profile sub-TLV: Indicates the name of the QoS-Profile profile in the upstream direction.

Egress-QoS-Profile Sub-TLV: Indicates the name of the QoS-Profile profile in the downstream direction.

User-ACL-Policy Sub-TLV: All ACL user policies, including v4ACLIN, v4ACLOUT, v6ACLIN, v6ACLOUT, v4WEBACL, v6WEBACL, v4SpecialACL, and v6SpecialACL.

Multicast-Profile4 Sub-TLV: IPv4 multicast policy name.

Multicast-Profile6 Sub-TLV: IPv6 multicast policy name.

NAT-Instance Sub-TLV: Indicates the instance ID of a NAT user.

The Reserved field MUST be sent as zero and ignored on receipt.

#### 7.9.13 Subscriber CGN Port Range TLV

The Subscriber CGN Port Range TLV is used to carry the NAT public address and port range. It will be carried in the Update\_Response message when CGN is used.

The format of this TLV is as follows:

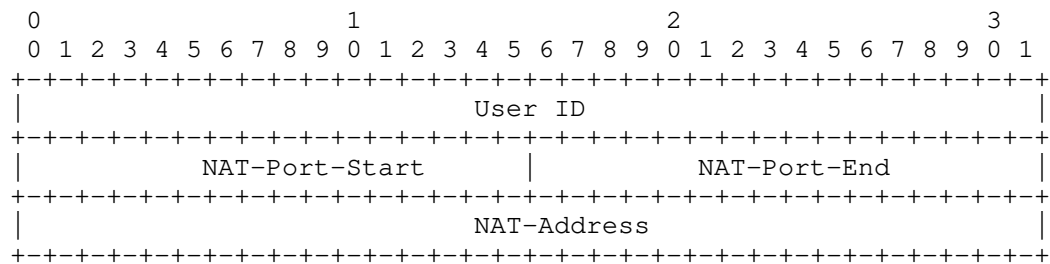


Figure 58: Subscriber CGN Port Range TLV

Where:

The TLV type: 15.

The TLV length: 12 octets.

User-ID (4 bytes): The identifier of a user/subscriber.

NAT-Port-Start (2 bytes): The start port number.

NAT-Port-End (2 bytes): The end port number.

NAT-Address (4 bytes): The NAT public network address.

#### 7.10 Device Status TLVs

The TLVs in this section are for reporting Interface and Board level information from the UP to the CP.

### 7.10.1 Interface Status TLV

The Interface Status TLV is used to carry the status information of an interface on a UP. It is carried in a Report message.

The format of the value part of this TLV is as follows:

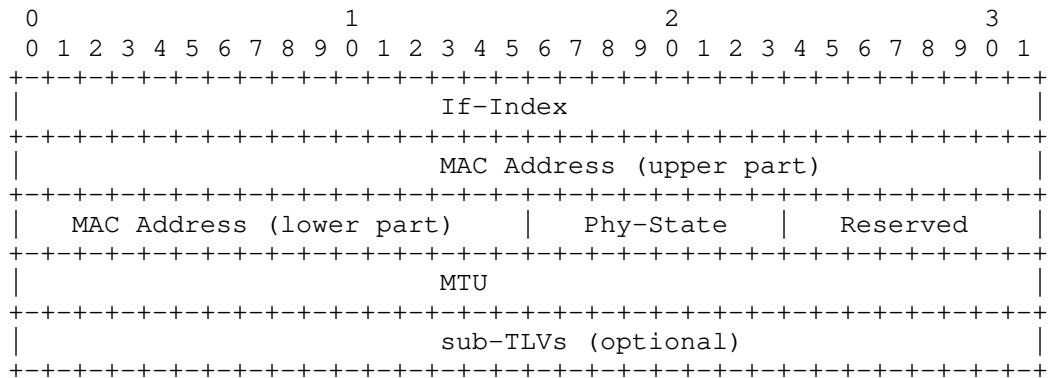


Figure 59: Interface Status TLV

Where:

The TLV type: 200.

The TLV length: variable.

If-Index (4 bytes): Indicates the interface index.

MAC-Address (MAC-Addr type): Interface MAC address.

Phy-State (1 byte): Physical status of the interface. 0: down, 1: Up

MTU (4 bytes): Interface MTU value.

sub-TLVs: The If-Desc and VRF-Name sub-TLVs can be carried.

The Reserved field MUST be sent as zero and ignored on receipt.

### 7.10.2 Board Status TLV

The Board Status TLV is used to carry the status information of a board on an UP. It is carried in a Report message.

The format of Board Status TLV is as follows:

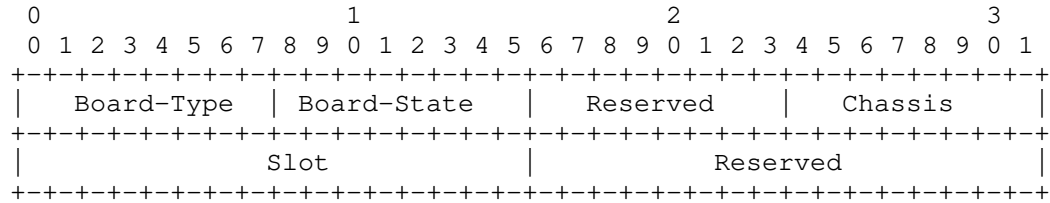


Figure 60: Interface Resource TLV

Where:

The TLV type: 201.

The TLV length: 8 octets.

Chassis (1 byte): The chassis number of the Board.

Slot (1 byte): The slot number of the Board.

Sub-Slot (1 byte): The sub-slot number of the Board.

Board-Type (1 byte):

- 1: CGN Service Process Unit (SPU) board.
- 2: Line Process Unit (LPU) Board.

Board-State (1 byte):

- 0: Normal.
- 1: Abnormal.

The reserved field MUST be sent as zero and ignored on receipt.

## 7.11 CGN TLVs

### 7.11.1 Address Allocation Request TLV

The Address Allocation Request TLV is used to request address allocation from CP. An address Pool-Name sub-TLV is carried to indicate from which address pool to allocate addresses. The Address Allocation Request TLV is carried in the Addr\_Allocation\_Req message.

The format of the value part of this TLV is as follows:

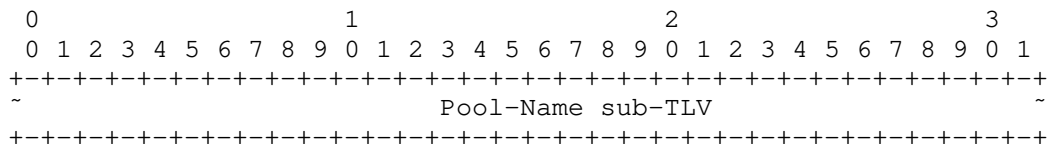


Figure 61: Address Allocation Request TLV

Where:

The TLV type: 400.

The TLV length: variable.

Pool-Name sub-TLV: Indicates from which Address pool to allocate address.

### 7.11.2 Address Allocation Response TLV

The Address Allocation Response TLV is used to return the address allocation result, it is carried the Addr\_Allocation\_Ack message.

The value part of the Address Allocation Response TLV is formatted as follows:

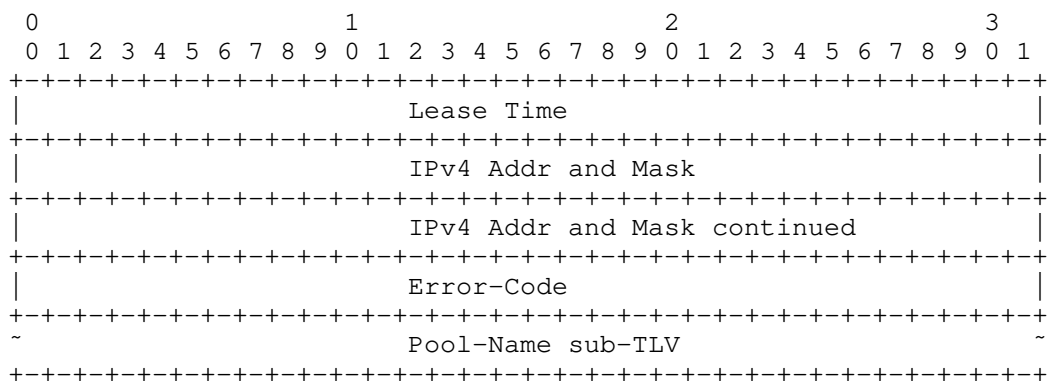


Figure 62: Address Assignment Response TLV

Where:

The TLV type: 401.

The TLV length: variable.



Lease Time (4 bytes): Duration of address lease in seconds.

IPv4 Addr and Mask (IPv4-Address type): The allocated IPv4 address.

Error-Code (4 bytes): Indicates success or an error.

0: Success.

1: Failure.

3001 (Pool-Mismatch): The corresponding address pool cannot be found.

3002 (Pool-Full): The address pool is fully allocated and no address segment is available.

Pool-Name sub-TLV: A Pool-Name sub-TLV to indicate from which Address pool the address is allocated.

### 7.11.3 Address Renewal Request TLV

The Address Renewal Request TLV is used to request address renewal from the CP. It is carried the Addr\_Renew\_Req message.

The format of this TLV value is as follows:

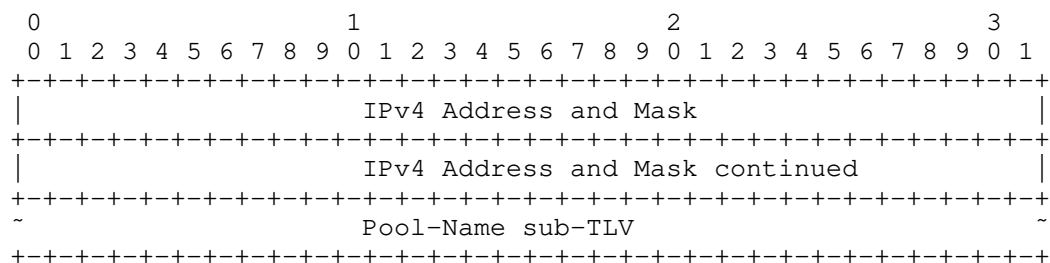


Figure 63: Address Renewal Request TLV

Where:

The TLV type: 402.

The TLV length: variable.

IPv4 Addr and Mask (IPv4-Addr): The IPv4 address to be renewed.

Pool Name sub-TLV: A Pool-Name sub-TLV to indicate from which

Address pool to renew the address.

#### 7.11.4 The Address Renewal Response TLV

The Address Renewal Response TLV is used to return the address renewal result. It is carried in the Addr\_Renew\_Ack message.

The format of this TLV value is as follows:

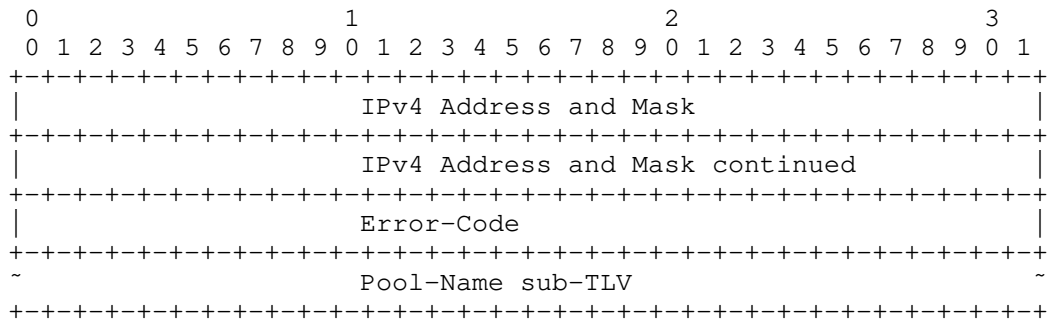


Figure 64: Address Renewal Response TLV

Where:

The TLV type: 403.

The TLV length: variable.

Client-IP (IPv4-Address type): The renewed IPv4 address.

Error Code (4 bytes): Indicates success or an error:

0: Renew success.

1: Renew failed.

3001 (Pool-Mismatch): The corresponding address pool cannot be found.

3002 (Pool-Full): The address pool is fully allocated and no address segment is available.

3003 (Subnet-Mismatch): The address pool subnet cannot be found.

3004 (Subnet-Conflict): Subnets in the address pool have been assigned to other clients.

Pool Name sub-TLV: A Pool-Name Sub-TLV to indicate from which Address pool to renew the address.

#### 7.11.5 Address Release Request TLV

The Address Release Request TLV is used to release an IPv4 address. It is carried in the Addr\_Release\_Req message.

The value part of this TLV is formatted as follows:

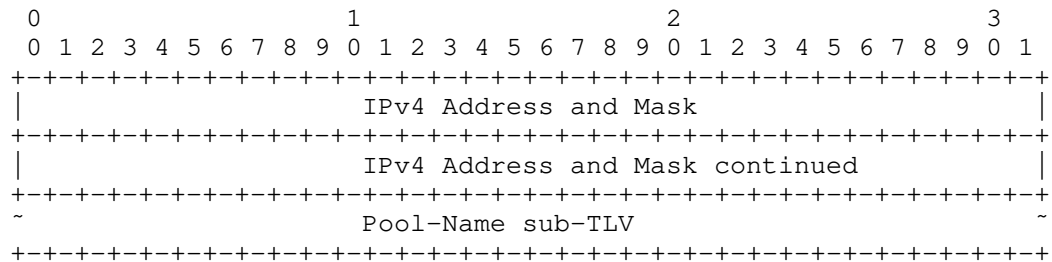


Figure 65: Address Release Request TLV

Where:

The TLV type: 404.

The TLV length: variable.

IPv4 Address and Mask (IPv4-Address type): The IPv4 address be released.

Pool-Name sub-TLV: A Pool-Name Sub-TLV to indicate from which Address pool to release the address.

#### 7.11.6 The Address Release Response TLV

The Address Release Response TLV is used to return the address release result. It is carried in the Addr\_Release\_Ack message.

The format of this TLV is as follows:

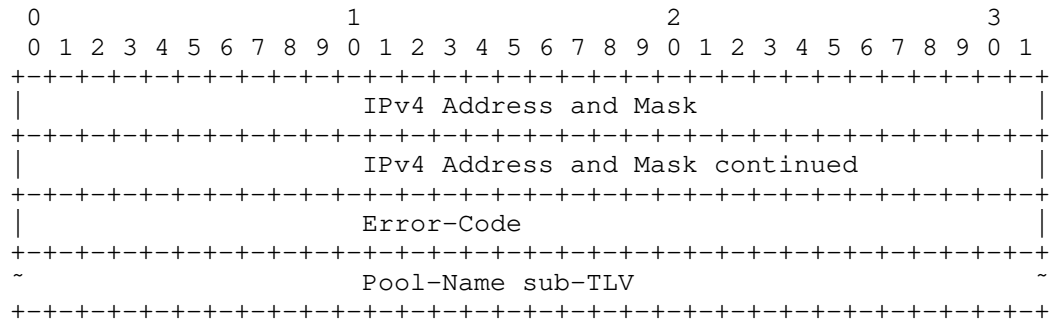


Figure 66: Address Renewal Response TLV

Where:

The TLV type: 405.

The TLV length: variable.

Client-IP (IPv4-Address type): The released IPv4 address.

Error-Code (4 bytes): Indicates success or an error.

0: Address release success.

1: Address release failed.

3001 (Pool-Mismatch): The corresponding address pool cannot be found.

3003 (Subnet-Mismatch): The address cannot be found.

3004 (Subnet-Conflict): The address has been allocated to another subscriber.

Pool-Name sub-TLV: A Pool-Name Sub-TLV to indicate from which address pool to release the address.

## 7.12 Event TLVs

## 7.12.1. Subscriber Traffic Statistics TLV

The Subscriber Traffic Statistics TLV is used to return the traffic statistics of a user/subscriber. The format of this TLV is as follows:

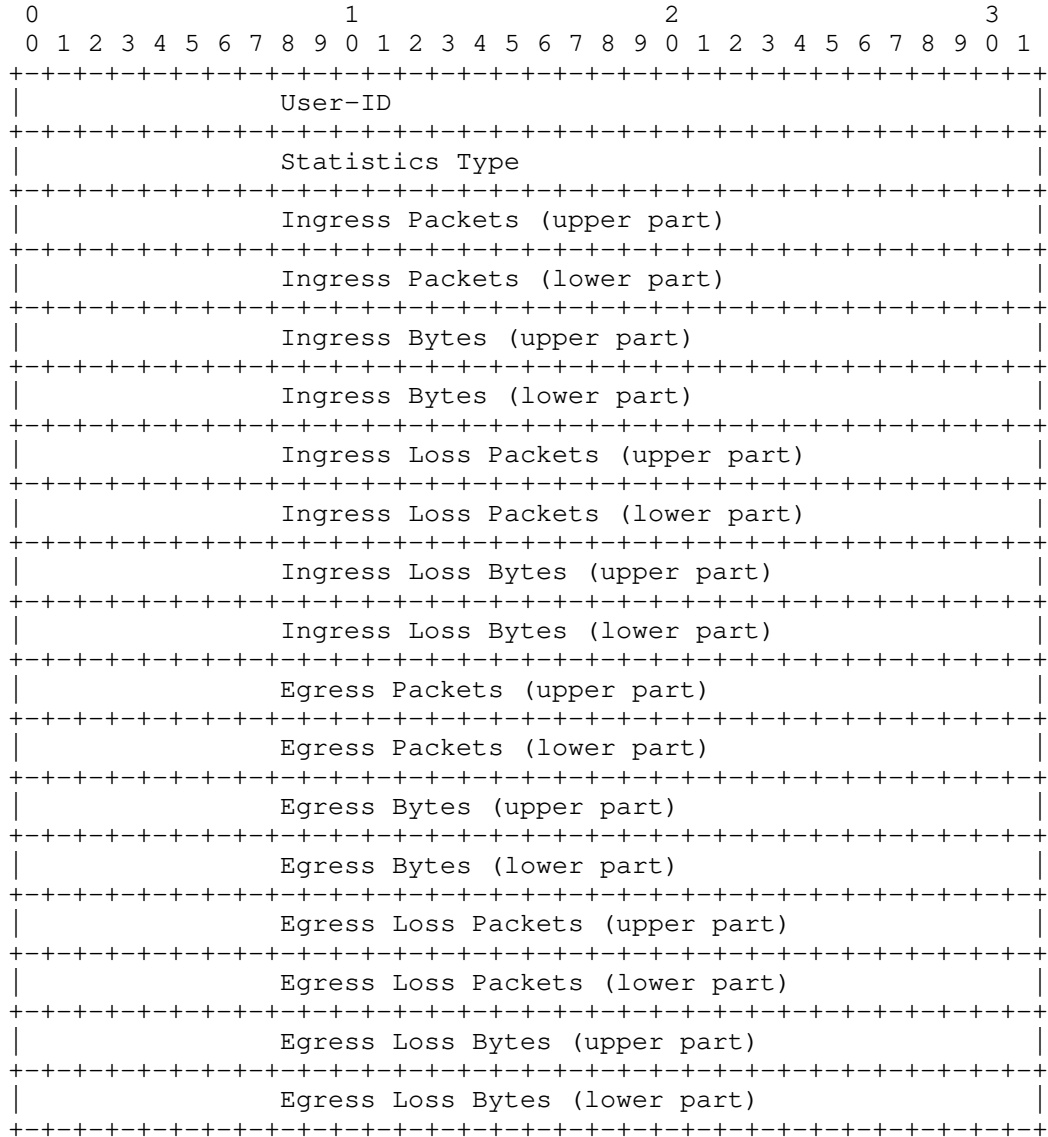


Figure 67: Subscriber Traffic Statistics TLV

Where:

The TLV type: 300.

The TLV length: 72 octets.

User-ID (4 bytes): The user/subscriber identifier.

Statistics-Type (4 bytes): Traffic type. It can be one of the following options:

- 0: IPv4 traffic.
- 1: IPv6 traffic.
- 2: Dual stack traffic.

Ingress Packets (8 bytes): The number of the packets in upstream direction.

Ingress Bytes (8 bytes): The bytes of the upstream traffic.

Ingress Loss Packets (8 bytes): The number of the lost packets in upstream direction.

Ingress Loss Bytes (8 bytes): The bytes of the lost upstream packets.

Egress Packets (8 bytes): The number of the packets in downstream direction.

Egress Bytes (8 bytes): The bytes of the downstream traffic.

Egress Loss Packets (8 bytes): The number of the lost packets in downstream direction.

Egress Loss Bytes (8 bytes): The bytes of the lost downstream packets.

#### 7.12.2 Subscriber Detection Result TLV

The Subscriber Detection Result TLV is used to return the detection result of a subscriber. Subscriber detection is a function to detect whether a subscriber is online or not. The result can be used by the CP to determine how to deal with the subscriber session. (e.g., delete the session if detection failed).

The format of this TLV value part is as follows:

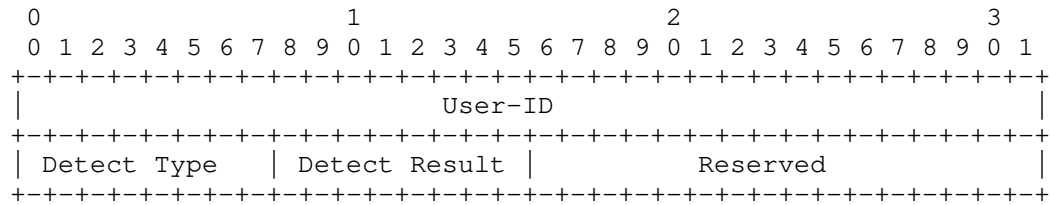


Figure 68: Subscriber Detection Result TLV

Where:

The TLV type: 301.

The TLV length: 8 octets.

User-ID (4 bytes): A user/subscriber identifier.

Detect-Type (1 byte):

0: IPv4 detection.

1: IPv6 detection.

2: PPP detection.

Detect-Result (1 bytes):

0: Indicates that the detection is successful.

1: Detection failure. The UP needs report only when the detection fails.

The Reserved field MUST be sent as zero and ignored on receipt.

### 7.13 Vendor TLV

The Vendor ID TLV occurs as the first TLV in the Vendor message (Section 6.6). It provides a Sub-Type that effectively extends the message type in the message header, provides for versioning of vendor TLVs, and can accommodate sub-TLVs.

The value part of the Vendor TLV is formatted as follows:

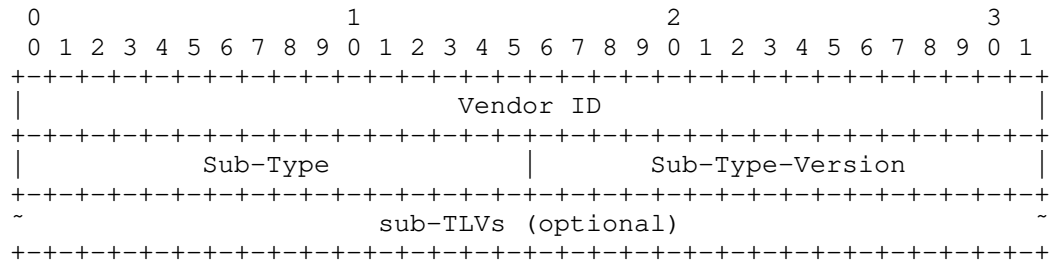


Figure 69: Vendor TLV

Where:

The TLV type: 1024.

The TLV length: variable.

Vendor-ID (4 bytes): Vendor ID as defined in RADIUS [RFC2865].

Sub-Type (2 bytes): Used by the Vendor to distinguish multiple different vendor messages.

Sub-Type-Version (2 bytes): Used by the Vendor to distinguish different versions of a Vendor Defined message sub-type.

Sub-TLVs (variable): Sub-TLVs as specified by the vendor.

Since Vendor code will be handling the TLV after the Vendor ID field is recognized, the remainder of the TLV value can be organized however the vendor wants. But it is desirable for a vendor to be able to define multiple different vendor messages and to keep track of different versions of its vendor defined messages. Thus, it is RECOMMENDED that the vendor assign a Sub-Type value for each vendor message that it defines different from other Sub-Type values that vendor has used. Also, when modifying a vendor defined message in a way potentially incompatible with a previous definition, the vendor SHOULD increase the value it is using in the Sub-Type-Version field.



## 8. Implementation Status

RFC Editor: Please remove this section before publication.

This section discusses the status of implementations that have provided information and the testing of this protocol at the time of posting of this Internet-Draft, and is based on the proposal in [RFC7942] ("Improving Awareness of Running Code: The Implementation Status Section"). The description of implementations in this section is intended to assist in processing drafts to RFCs.

Please note that the listing of any individual implementation or test results here does not imply endorsement by the RFC Editor or the IETF. Furthermore, no effort has been spent to verify the information presented here that was supplied by contributors. This is not intended as, and must not be construed to be, a catalog of available implementations or their testing or features. Readers are advised to note that other implementations may exist.

According to [RFC7942], "this will allow reviewers ... to assign due consideration to documents that have the benefit of running code, which may serve as evidence of valuable experimentation and feedback that have made the implemented protocols more mature."

### 8.1 Implementations

Information on three S-CUSP implementations appears below.

#### 8.1.1 Huawei Technologies

Name: Cloud-based BNG.

Maturity: Production.

Coverage: According to S-CUSP protocol.

Contact information:

Zhouyi Yu: yuzhouyi@huawei.com

Date: 2018-11-01

### 8.1.2 ZTE

Name: ZXR10 V6000 vBRAS

Maturity: Production

Coverage: According to S-CUSP protocol.

Contact information:

Yong Chen: 10056167@zte.com.cn

Huaibin Wang: 10008729@zte.com.cn

Date: 2018-12-01

### 8.1.3 H3C

Name: CUSP protocol for BRAS Control Plane and User Plan Separation

Maturity: Research

Coverage: According to S-CUSP protocol

Contact information: mengdan@h3c.com; liuhanlei@h3c.com

Date: 2019-1-30

## 8.2 Hackathon

Successful use of the protocol at the IETF-102 Hackathon, Montreal, Quebec, in 2018.

Hackathon Project: Control Plane and User Plane Separation BNG control channel Protocol (CUSP)

Champions: Zhenqiang Li, Michael Wang

Report: See

[github.com/IETF-Hackathon/ietf102-project-presentations/blob/master/IETF102-hackathon-presentation-CUSP.pptx](https://github.com/IETF-Hackathon/ietf102-project-presentations/blob/master/IETF102-hackathon-presentation-CUSP.pptx)

### 8.3 EANTC Testing

EANTC (European Advanced Networking Test Center ([www.eantc.de](http://www.eantc.de))) tested the Huawei implementation. Their summary was as follows:  
"EANTC tested advanced aspects of the Cloud-based Broadband Network Gateway (vBNG) with a focus on performance, scalability and high availability with up to 20 Million emulated subscribers. The solution performed very well across all test scenarios."

See report at  
[www.eantc.de/fileadmin/eantc/downloads/News/2018/EANTC-vBRAS-phase2.pdf](http://www.eantc.de/fileadmin/eantc/downloads/News/2018/EANTC-vBRAS-phase2.pdf)

## 9. IANA Considerations

IANA is requested to create an "S-CUSP Parameters" web page and include thereon the registries set up in the Sub-Sections below.

### 9.1 Message Types

IANA is requested to create an S-CUSP Message Types registry on the S-CUSP Parameters Web Page as follows:

Registry Name: S-CUSP Message Types  
 Registration Procedure: Expert Review  
 Reference: [this document]

Type	Name	Section of [this document]
0	- Reserved	
1	Hello	6.2.1.
2	Keepalive	6.2.2.
3	Sync_Request	6.2.3.
4	Sync_Begin	6.2.4.
5	Sync_Data	6.2.5.
6	Sync_End	6.2.6.
7	Update_Request	6.2.7.
8	Update_Response	6.2.8.
9	Report	6.4.
10	Event	6.3.
11	Vendor	6.6.
12	Error	6.7.
13-199	- Unassigned	
200	Addr_Allocation_Req	6.5.1.
201	Addr_Allocation_Ack	6.5.2.
202	Addr_Renew_Req	6.5.3.
203	Addr_Renew_Ack	6.5.4.
204	Addr_Release_Req	6.5.5.
205	Addr_Release_Ack	6.5.6.
206-254	- Unassigned	
255	- Reserved	

### 9.2 TLV Types

IANA is requested to create an S-CUSP TLV Types registry on the S-CUSP Parameters Web Page as follows:

Registry Name: S-CUSP TLV Types  
 Registration Procedure: Expert Review  
 Reference: [this document]

Type	Name	Usage Description
0	reserved	-
1	BAS Function	Carries the BNG access functions to be enabled or disabled on specified interfaces.
2	Basic Subscriber	Carries the basic information about a BNG subscriber.
3	PPP Subscriber	Carries the PPP information about a BNG subscriber.
4	IPv4 Subscriber	Carries the IPv4 address of a BNG subscriber.
5	IPv6 Subscriber	Carries the IPv6 address of a BNG subscriber.
6	Subscriber Policy	Carries the policy information applied to a BNG subscriber.
7	IPv4 Routing	Carries the IPv4 network routing information.
8	IPv6 Routing	Carries the IPv6 network routing information.
9	IPv4 Static Subscriber Detect	Carries the IPv4 static subscriber detect information.
10	IPv6 Static Subscriber Detect	Carries the IPv6 static subscriber detect information.
11	L2TP-LAC Subscriber	Carries the L2TP LAC subscriber information.
12	L2TP-LNS Subscriber	Carries the L2TP LNS subscriber information.
13	L2TP-LAC-Tunnel	Carries the L2TP LAC tunnel subscriber information.
14	L2TP-LNS-Tunnel	Carries the L2TP LNS tunnel subscriber information.
15	Subscriber CGN Port Range	Carries the public IPv4 address and related port range of a BNG subscriber.
16-99	unassigned	-
100	Hello	Used for version and Keep Alive timers negotiation.
101	Error Information	Carried in the Ack of the control message. Carries the processing result, success, or error.
102	Keep Alive	Carried in the Hello message for Keep Alive timers negotiation.

103-199	unassigned	-
200	Interface Status	Interfaces status reported by the UP including physical interfaces, sub-interfaces, trunk interfaces, and tunnel interfaces.
201	Board Status	Board information reported by the UP including the board type and in-position status.
202-299	unassigned	-
300	Subscriber Traffic Statistics	User traffic statistics.
301	Subscriber Detection Results	User detection information.
302	Update Response	The processing result of a subscriber session update.
303-299	unassigned	-
400	Address Allocation Request	Request address allocation.
401	Address Allocation Response	Address allocation response.
402	Address Renewal Request	Request for address lease renewal.
403	Address Renewal Response	Response to a request for extending an IP address lease.
404	Address Release Request	Request to release the address.
405	Address Release Response	Ack of a message releasing an IP address.
406-1023	unassigned	-
1024	Vendor	As implemented by vendor.
1039-4095	unassigned	-

### 9.3 TLV Operation Codes

IANA is requested to create an S-CUSP TLV Operation Codes registry on the S-CUSP Parameters Web Page as follows:

Registry Name: S-CUSP TLV Operation Codes  
 Registration Procedure: Expert Review  
 Reference: [this document]

Code	Operation
----	-----
0	- reserved
1	Update
2	Delete
3-15	- unassigned

#### 9.4 Sub-TLV Types

IANA is requested to create an S-CUSP Sub-TLV Types registry on the S-CUSP Parameters Web Page as follows:

Registry Name: S-CUSP Sub-TLV Types  
 Registration Procedure: Expert Review  
 Reference: [this document]

Type	Name	Section of [this document]
0	- reserved	
1	VRF Name	7.3.1.
2	Ingress-QoS-Profile	7.3.1.
3	Egress-QoS-Profile	7.3.1.
4	User-ACL-Policy	7.3.1.
5	Multicast-ProfileV4	7.3.1.
6	Multicast-ProfileV6	7.3.1.
7	Ingress-CAR	7.3.2.
8	Egress-CAR	7.3.3.
9	NAT-Instance	7.3.1.
10	Pool-Name	7.3.1.
11	If-Desc	7.3.4.
12	IPv6-Address List	7.3.5.
13	Vendor	7.3.6.
12-64534	- unassigned	
65535	- reserved	

#### 9.5 Error Codes

IANA is requested to create an S-CUSP ERRID Codes registry on the S-CUSP Parameters Web Page as follows:

Registry Name: S-CUSP ERRID Codes  
 Registration Procedure: Expert Review  
 Reference: [this document]

Value	Name	Remarks
0	Success	Success
1	Fail	Malformed message received.
2	TLV-Unknown	One or more of the TLVs was not understood.
3	TLV-Length	The TLV length is abnormal.
4-999	- unassigned	Unassigned basic error codes.
1000	- reserved	
1001	Version-Mismatch	The version negotiation fails. Terminate.

		The subsequent service processes corresponding to the UP are suspended.
1002	Keepalive Error	The keepalive negotiation fails.
1003	Timer Expires	The establishment timer expired.
1004-1999	- unassigned	Unassigned error codes for version negotiation.
2000	- reserved	
2001	Synch-NoReady	The data to be smoothed is not ready.
2002	Synch-Unsupport	The request for smooth data is not supported.
2003-2999	- unassigned	Unassigned data synchronization error codes.
3000	- reserved	
3001	Pool-Mismatch	The corresponding address pool cannot be found.
3002	Pool-Full	The address pool is fully allocated and no address segment is available.
3003	Subnet-Mismatch	The address pool subnet cannot be found.
3004	Subnet-Conflict	Subnets in the address pool have been classified into other clients.
3005-3999	- unassigned	Unassigned error codes for address allocation.
4000	- reserved	
4001	Update-Fail-No-Res	The forwarding table fails to be delivered because the forwarding resources are insufficient.
4002	QoS-Update-Success	The QoS policy takes effect.
4003	QoS-Update-Sq-Fail	Failed to process the queue in the QoS policy.
4004	QoS-Update-CAR-Fail	Processing of the CAR in the QoS policy fails.
4005	Statistic-Fail-No-Res	Statistics processing failed due to insufficient statistics resources.
4006-4999	- unassigned	forwarding table delivery error codes.
5000-4294967295	- reserved	



## 10. Security Considerations

The Service, Control, and Management Interfaces between the CP and UP might be across the general Internet or other hostile environment. The ability of an adversary to block or corrupt messages or introduce spurious messages on any one or more of these interfaces would give the adversary the ability to stop subscribers from accessing network services, disrupt existing subscriber sessions, divert traffic, mess up accounting statistics, and generally cause havoc. Damage would not necessarily be limited to one or a few subscribers but could disrupt routing or deny service to one or more instances of the CP or otherwise cause extensive interference. If the adversary knows the details of the UP equipment and its forwarding rule capabilities, the adversary may be able to cause a copy of most or all user data to be sent to an address of the adversary's choosing, thus enabling eavesdropping.

Thus, appropriate protections **MUST** be implemented to provide integrity, authenticity, and secrecy of traffic over those interfaces. Whether such protection is used is a network operator decision. See [RFC6241] for Management Interface / NETCONF security. Security on the Service Interface is dependent on the tunneling protocol used which is out of scope for this document. Security for the Control Interface, over which the S-CUSP protocol flows, is further discussed below.

S-CUSP messages do not provide security. Thus, if these messages are exchanged in an environment where security is a concern, that security **MUST** be provided by another protocol such as TLS 1.3 [RFC8446] or IPSEC. TLS 1.3 is the mandatory to implement protocol for interoperability. The use of a particular security protocol on the Control Interface is determined by configuration. Such security protocols need not always be used and lesser security precautions might be appropriate because, in some cases, the communication between the CP and UP is in a benign environment.

Contributors

Zhouyi Yu  
Huawei Technologies

Email: yuzhouyi@huawei.com

Chengguang Niu  
Huawei Technologies

Email: chengguang.niu@huawei.com

Zitao Wang  
Huawei Technologies

Email: wangzitao@huawei.com

Jun Song  
Huawei Technologies

Email: song.jun@huawei.com

Dan Meng  
H3C Technologies  
No.1 Lixing Center  
No.8 guangxun south street, Chaoyang District,  
Beijing, 100102  
China

Email: mengdan@h3c.com

Hanlei Liu  
H3C Technologies  
No.1 Lixing Center  
No.8 guangxun south street, Chaoyang District,  
Beijing, 100102  
China

Email: hanlei\_liu@h3c.com

## Normative References

- [RFC20] Cerf, V., "ASCII format for network interchange", STD 80, RFC 20, DOI 10.17487/RFC0020, October 1969, <<https://www.rfc-editor.org/info/rfc20>>.
- [RFC793] Postel, J., "Transmission Control Protocol", STD 7, RFC 793, DOI 10.17487/RFC0793, September 1981, <<https://www.rfc-editor.org/info/rfc793>>.
- [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>>.
- [RFC2661] Townsley, W., Valencia, A., Rubens, A., Pall, G., Zorn, G., and B. Palter, "Layer Two Tunneling Protocol "L2TP"", RFC 2661, DOI 10.17487/RFC2661, August 1999, <<https://www.rfc-editor.org/info/rfc2661>>.
- [RFC2865] Rigney, C., Willens, S., Rubens, A., and W. Simpson, "Remote Authentication Dial In User Service (RADIUS)", RFC 2865, DOI 10.17487/RFC2865, June 2000, <<https://www.rfc-editor.org/info/rfc2865>>.
- [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>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.

## Informative References

- [802.1Q] IEEE, "IEEE Standard for Local and metropolitan area networks / Bridges and Bridged Networks", IEEE Std 802.1Q-2014, 3 November 2013.
- [RFC1661] Simpson, W., Ed., "The Point-to-Point Protocol (PPP)", STD 51, RFC 1661, DOI 10.17487/RFC1661, July 1994, <<https://www.rfc-editor.org/info/rfc1661>>.
- [RFC2131] Droms, R., "Dynamic Host Configuration Protocol", RFC 2131, DOI 10.17487/RFC2131, March 1997, <<https://www.rfc-editor.org/info/rfc2131>>.
- [RFC2516] Mamakos, L., Lidl, K., Evarts, J., Carrel, D., Simone, D., and R. Wheeler, "A Method for Transmitting PPP Over Ethernet (PPPoE)", RFC 2516, DOI 10.17487/RFC2516, February 1999, <<https://www.rfc-editor.org/info/rfc2516>>.
- [RFC2698] Heinanen, J. and R. Guerin, "A TwoRate Three Color Marker", RFC 2698, DOI 10.17487/RFC2698, September 1999, <<https://www.rfc-editor.org/info/rfc2698>>.
- [RFC3022] Srisuresh, P. and K. Egevang, "Traditional IP Network Address Translator (Traditional NAT)", RFC 3022, DOI 10.17487/RFC3022, January 2001, <<https://www.rfc-editor.org/info/rfc3022>>.
- [RFC3336] Thompson, B., Koren, T., and B. Buffam, "PPP Over Asynchronous Transfer Mode Adaptation Layer 2 (AAL2)", RFC 3336, DOI 10.17487/RFC3336, December 2002, <<https://www.rfc-editor.org/info/rfc3336>>.
- [RFC5511] Farrel, A., "Routing Backus-Naur Form (RBNF): A Syntax Used to Form Encoding Rules in Various Routing Protocol Specifications", RFC 5511, DOI 10.17487/RFC5511, April 2009, <<https://www.rfc-editor.org/info/rfc5511>>.
- [RFC7042] Eastlake 3rd, D. and J. Abley, "IANA Considerations and IETF Protocol and Documentation Usage for IEEE 802 Parameters", BCP 141, RFC 7042, DOI 10.17487/RFC7042, October 2013, <<https://www.rfc-editor.org/info/rfc7042>>.
- [RFC7348] Mahalingam, M., Dutt, D., Duda, K., Agarwal, P., Kreeger, L., Sridhar, T., Bursell, M., and C. Wright, "Virtual eXtensible Local Area Network (VXLAN): A Framework for Overlaying Virtualized Layer 2 Networks over Layer 3 Networks", RFC 7348, DOI 10.17487/RFC7348, August 2014, <<https://www.rfc-editor.org/info/rfc7348>>.

- [RFC7942] Sheffer, Y. and A. Farrel, "Improving Awareness of Running Code: The Implementation Status Section", BCP 205, RFC 7942, DOI 10.17487/RFC7942, July 2016, <<https://www.rfc-editor.org/info/rfc7942>>.
- [RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <<https://www.rfc-editor.org/info/rfc8446>>.
- [TR-384] Broadband Forum, "Cloud Central Office Reference Architectural Framework", BBF TR-384, 2018.

Authors' Addresses

Shujun Hu  
China Mobile  
32 Xuanwumen West Ave, Xicheng District  
Beijing, Beijing 100053  
China

Email: hushujun@chinamobile.com

Donald Eastlake, 3rd  
Futurewei Technologies  
1424 Pro Shop Court  
Davenport, FL 33896  
USA

Phone: +1-508-333-2270  
Email: d3e3e3@gmail.com

Mach (Guoyi) Chen  
Huawei Technologies  
Huawei Bldg., No. 156 Beiqing Road  
Beijing 100095 China

Email: mach.chen@huawei.com

Fengwei Qin  
China Mobile  
32 Xuanwumen West Ave, Xicheng District  
Beijing, Beijing 100053  
China

Email: qinfengwei@chinamobile.com

Zhenqiang Li  
China Mobile  
32 Xuanwumen West Ave, Xicheng District  
Beijing, Beijing 100053  
China

Email: lizhenqiang@chinamobile.com

Tee Mong Chua  
Singapore Telecommunications Limited  
31 Exeter Road, #05-04 Comcentre Podium Block  
Singapore City 239732  
Singapore

Email: teemong@singtel.com

Daniel Huang  
ZTE

Email: huang.guangping@zte.com.cn

Copyright, Disclaimer, and Additional IPR Provisions

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 (<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.





Network Working Group  
Internet Draft  
Intended status: Informational  
Expires: January 26, 2021

L. Dunbar  
Futurewei  
A. Malis  
Malis Consulting  
C. Jacquenet  
Orange  
July 26, 2020

Networks Connecting to Hybrid Cloud DCs: Gap Analysis  
draft-ietf-rtgwg-net2cloud-gap-analysis-07

Abstract

This document analyzes the IETF routing area technical gaps that may affect the dynamic connection to workloads and applications hosted in hybrid Cloud Data Centers from enterprise premises.

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), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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."

The list of current Internet-Drafts can be accessed at  
<http://www.ietf.org/ietf/lid-abstracts.txt>

The list of Internet-Draft Shadow Directories can be accessed at  
<http://www.ietf.org/shadow.html>

This Internet-Draft will expire on January 26, 2009.

## Copyright Notice

Copyright (c) 2020 IETF Trust and the persons identified as the document authors. All rights reserved.

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

## Table of Contents

1. Introduction.....	3
2. Conventions used in this document.....	3
3. Gap Analysis for Accessing Cloud Resources.....	4
3.1. Multiple PEs connecting to virtual CPEs in Cloud DCs.....	6
3.2. Access Control for workloads in the Cloud DCs.....	6
3.3. NAT Traversal.....	7
3.4. BGP between PEs and remote CPEs via Internet.....	7
3.5. Multicast traffic from/to the remote edges.....	8
4. Gap Analysis of Traffic over Multiple Underlay Networks.....	9
5. Aggregating VPN paths and Internet paths.....	10
5.1. Control Plane for Cloud Access via Heterogeneous Networks.....	11
5.2. Using BGP UPDATE Messages.....	12
5.2.1. Lack ways to differentiate traffic in Cloud DCs.....	12
5.2.2. Miss attributes in Tunnel-Encap.....	12
5.3. SECURE-EVPN/BGP-EDGE-DISCOVERY.....	12
5.4. SECURE-L3VPN.....	13
5.5. Preventing attacks from Internet-facing ports.....	14
6. Gap Summary.....	14
7. Manageability Considerations.....	15
8. Security Considerations.....	16
9. IANA Considerations.....	16
10. References.....	16
10.1. Normative References.....	16
10.2. Informative References.....	16
11. Acknowledgments.....	17

## 1. Introduction

[Net2Cloud-Problem] describes the problems enterprises face today when interconnecting their branch offices with dynamic workloads hosted in third party data centers (a.k.a. Cloud DCs). In particular, this document analyzes the available routing protocols to identify whether there are any gaps that may impede such interconnection which may for example justify additional specification effort to define proper protocol extensions.

For the sake of readability, an edge, C-PE, or CPE are used interchangeably throughout this document. More precisely:

- . Edge: may include multiple devices (virtual or physical);
- . C-PE: provider-owned edge, e.g. for SECURE-EVPN's PE-based BGP/MPLS VPN, where PE is the edge node;
- . CPE: device located in enterprise premises.

## 2. Conventions used in this document

Cloud DC: Third party Data Centers that usually host applications and workload owned by different organizations or tenants.

Controller: Used interchangeably with Overlay controller to manage overlay path creation/deletion and monitor the path conditions between sites.

CPE-Based VPN: Virtual Private Network designed and deployed from CPEs. This is to differentiate from most commonly used PE-based VPNs a la RFC 4364.

OnPrem: On Premises data centers and branch offices

### 3. Gap Analysis for Accessing Cloud Resources

Because of the ephemeral property of the selected Cloud DCs for specific workloads/Apps, an enterprise or its network service provider may not have direct physical connections to the Cloud DCs that are optimal for hosting the enterprise's specific workloads/Apps. Under those circumstances, an overlay network design can be an option to interconnect the enterprise's on-premises data centers & branch offices to its desired Cloud DCs.

However, overlay paths established over the public Internet can have unpredictable performance, especially over long distances. Therefore, it is highly desirable to minimize the distance or the number of segments that traffic had to be forwarded over the public Internet.

The Metro Ethernet Forum's Cloud Service Architecture [MEF-Cloud] also describes a use case of network operators using Overlay paths over an LTE network or the public Internet for the last mile access where the VPN service providers cannot always provide the required physical infrastructure.

In some scenarios, some overlay edge nodes may not be directly attached to the PEs that participate to the delivery and the operation of the enterprise's VPN.

When using an overlay network to connect the enterprise's sites to the workloads hosted in Cloud DCs, the existing C-PEs at enterprise's sites have to be upgraded to connect to the said overlay network. If the workloads hosted in Cloud DCs need to be connected to many sites, the upgrade process can be very expensive.

[Net2Cloud-Problem] describes a hybrid network approach that extends the existing MPLS-based VPNs to the Cloud DC Workloads over the access paths that are not under the VPN provider's control. To make it work properly, a small number of the PEs of the BGP/MPLS VPN can be designated to connect to the remote workloads via secure IPsec tunnels. Those designated PEs are shown as fPE (floating PE or smart PE) in Figure 3. Once the secure IPsec tunnels are established, the workloads hosted in Cloud DCs can be reached by the enterprise's VPN without upgrading all of the enterprise's CPEs. The

only CPE that needs to connect to the overlay network would be a virtualized CPE instantiated within the cloud DC.

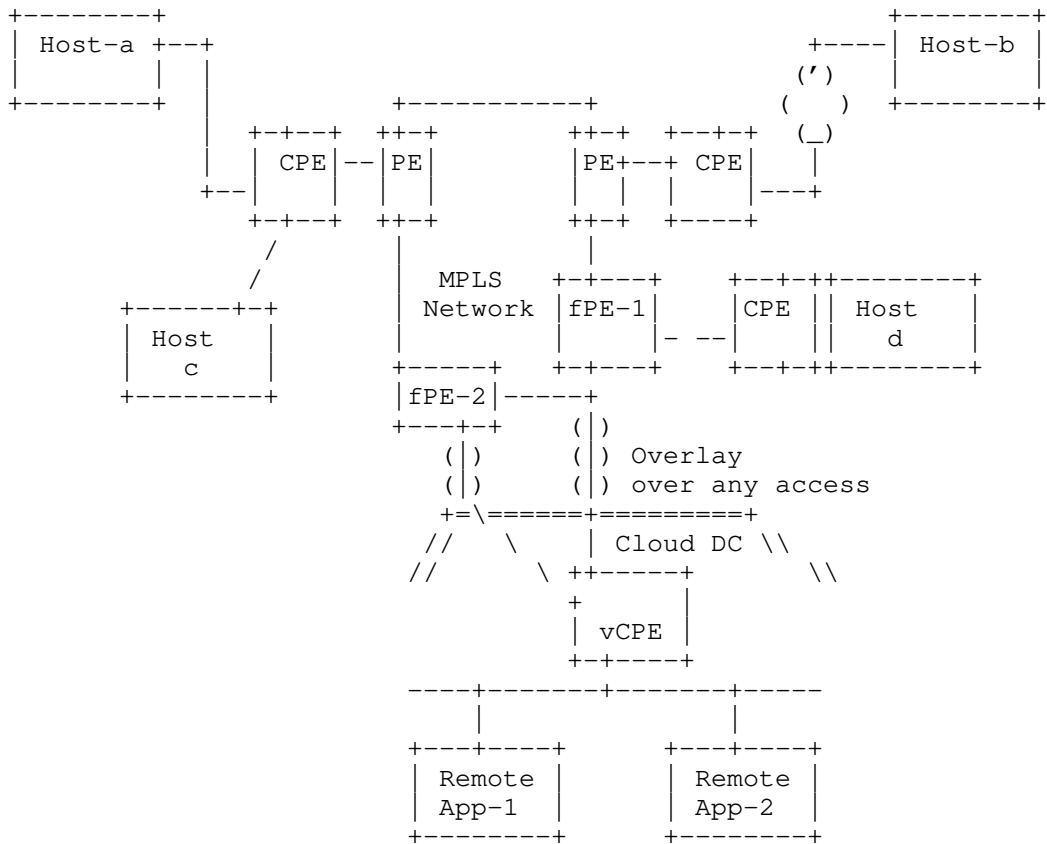


Figure 1: VPN Extension to Cloud DC

In Figure 1, the optimal Cloud DC to host the workloads (as a function of the proximity, capacity, pricing, or any other criteria chosen by the enterprises) does not have a direct connection to the PEs of the NGP/MPLS VPN that interconnects the enterprise's sites.

### 3.1. Multiple PEs connecting to virtual CPEs in Cloud DCs

To extend BGP/MPLS VPNs to virtual CPEs in Cloud DCs, it is necessary to establish secure tunnels (such as IPsec tunnels) between the PEs and the vCPEs.

Even though a set of PEs can be manually selected for a specific cloud data center, there are no standard protocols for those PEs to interact with the vCPEs instantiated in the third party cloud data centers over unsecure networks. The interaction includes exchanging performance, route information, etc..

When there is more than one PE available for use (as there should be for resiliency purposes or because of the need to support multiple cloud DCs geographically scattered), it is not straightforward to designate an egress PE to remote vCPEs based on applications. It might not be possible for PEs to recognize all applications because too much traffic traversing the PEs.

When there are multiple floating PEs that have established IPsec tunnels with a remote CPE, the remote CPE can forward outbound traffic to the optimal PE, which in turn forwards traffic to egress PEs to reach the final destinations. However, it is not straightforward for the ingress PE to select which egress PEs to send traffic. For example, in Figure 1:

- fPE-1 is the optimal PE for communication between App-1 <-> Host-a due to latency, pricing or other criteria.
- fPE-2 is the optimal PE for communication between App-1 <-> Host-b.

### 3.2. Access Control for workloads in the Cloud DCs

There is widespread diffusion of access policy for Cloud Resource, some of which is not easy for verification and validation. Because there are multiple parties involved in accessing Cloud Resources, policy enforcement points are not easily visible for policy refinement, monitoring, and testing.

The current state of the art for specifying access policies for Cloud Resources could be improved by having automated and reliable tools to map the user-friendly (natural language) rules into machine readable policies and to provide interfaces for enterprises to self-manage policy enforcement points for their own workloads.

### 3.3. NAT Traversal

Cloud DCs that only assign private IPv4 addresses to the instantiated workloads assume that traffic to/from the workload usually needs to traverse NATs.

There is no automatic way for an enterprise's network controller to be informed of the NAT properties for its workloads in Cloud DCs

One potential solution could be utilizing the messages sent during initialization of an IKE VPN when NAT Traversal option is enabled. There are some inherent problems while sending IPSec packets through NAT devices. One way to overcome these problems is to encapsulate IPSec packets in UDP. To do this effectively, there is a discovery phase in IKE (Phase1) that tries to determine if either of the IPSec gateways is behind a NAT device. If a NAT device is found, IPSec-over-UDP is proposed during IPSec (Phase 2) negotiation. If there is no NAT device detected, IPSec is used

Another potential solution could be allowing the virtual CPE in Cloud DCs to solicit a STUN (Session Traversal of UDP Through Network Address Translation, [RFC3489]) Server to get the information about the NAT property, the public IP addresses and port numbers so that such information can be communicated to the relevant peers.

### 3.4. BGP between PEs and remote CPEs via Internet

Even though an EBGp (external BGP) Multi-Hop design can be used to connect peers that are not directly connected to each other, there are still some issues about extending BGP from MPLS VPN PEs to remote CPEs in cloud DCs via any access path (e.g., Internet).

The path between the remote CPEs and VPN PEs that maintain VPN routes can traverse untrusted segments.



EBGP Multi-hop design requires configuration on both peers, either manually or via NETCONF from a controller. To use EBGP between a PE and remote CPEs, the PE has to be manually configured with the "next-hop" set to the IP address of the CPEs. When remote CPEs, especially remote virtualized CPEs are dynamically instantiated or removed, the configuration of Multi-Hop EBGP on the PE has to be changed accordingly.

Egress peering engineering (EPE) is not sufficient. Running BGP on virtualized CPEs in Cloud DCs requires GRE tunnels to be established first, which requires the remote CPEs to support address and key management capabilities. RFC 7024 (Virtual Hub & Spoke) and Hierarchical VPN do not support the required properties.

Also, there is a need for a mechanism to automatically trigger configuration changes on PEs when remote CPEs' are instantiated or moved (leading to an IP address change) or deleted.

EBGP Multi-hop design does not include a security mechanism by default. The PE and remote CPEs need secure communication channels when connecting via the public Internet.

Remote CPEs, if instantiated in Cloud DCs might have to traverse NATs to reach PEs. It is not clear how BGP can be used between devices located beyond the NAT and the devices located behind the NAT. It is not clear how to configure the Next Hop on the PEs to reach private IPv4 addresses.

### 3.5. Multicast traffic from/to the remote edges

Among the multiple floating PEs that are reachable from a remote CPE in a Cloud DC, multicast traffic sent by the remote CPE towards the MPLS VPN can be forwarded back to the remote CPE due to the PE receiving the multicast packets forwarding the multicast/broadcast frame to other PEs that in turn send to all attached CPEs. This process may cause traffic loops.

This problem can be solved by selecting one floating PE as the CPE's Designated Forwarder, similar to TRILL's Appointed Forwarders [RFC6325].

BGP/MPLS VPNs do not have features like TRILL's Appointed Forwarders.

#### 4. Gap Analysis of Traffic over Multiple Underlay Networks

Very often the Hybrid Cloud DCs are interconnected by multiple types of underlay networks, such as VPN, public Internet, wireless and wired infrastructures, etc. Sometimes the enterprises' VPN providers do not have direct access to the Cloud DCs that host some specific applications or workloads operated by the enterprise.

When reached by an untrusted network, all sensitive data to/from this virtual CPE have to be encrypted, usually by means of IPsec tunnels. When reached by a trusted direct connect paths, sensitive data can be forwarded without encryption for better performance.

If a virtual CPE in Cloud DC can be reached by both trusted and untrusted paths, better performance can be achieved to have a mixed encrypted and unencrypted traffic depending which paths the traffic is forwarded. However, there is no appropriate control plane protocol to achieve this automatically.

Some networks achieve the IPsec tunnel automation by using the modified NHRP protocol [RFC2332] to register network facing ports of the edge nodes with their Controller (or NHRP server), which then maps a private VPN address to a public IP address of the destination node/port. DSVPN [DSVPN] or DMVPN [DMVPN] are used to establish tunnels between WAN ports of SDWAN edge nodes.

NHRP was originally intended for ATM address resolution, and as a result, it misses many attributes that are necessary for dynamic virtual C-PE registration to the controller, such as:

- Interworking with the MPLS VPN control plane. An overlay edge can have some ports facing the MPLS VPN network over which packets can be forwarded without any encryption and some ports facing the

public Internet over which sensitive traffic needs to be encrypted.

- Scalability: NHRP/DSVPN/DMVPN work fine with small numbers of edge nodes. When a network has more than 100 nodes, these protocols do not scale well.
- NHRP does not have the IPsec attributes, which are needed for peers to build Security Associations over the public Internet.
- NHRP messages do not have any field to encode the C-PE supported encapsulation types, such as IPsec-GRE or IPsec-VxLAN.
- NHRP messages do not have any field to encode C-PE Location identifiers, such as Site Identifier, System ID, and/or Port ID.
- NHRP messages do not have any field to describe the gateway(s) to which the C-PE is attached. When a C-PE is instantiated in a Cloud DC, it is desirable for the C-PE's owner to be informed about how and where the C-PE is attached.
- NHRP messages do not have any field to describe C-PE's NAT properties if the C-PE is using private IPv4 addresses, such as the NAT type, Private address, Public address, Private port, Public port, etc.

## 5. Aggregating VPN paths and Internet paths

Most likely, enterprises (especially the largest ones) already have their C-PEs interconnected by VPNs, based upon VPN techniques like EVPN, L2VPN, or L3VPN. Their VPN providers might have direct paths/links to the Cloud DCs that host their workloads and applications.

When there is short term high traffic volume that can't justify increasing the VPNs capacity, enterprises can utilize public internet to reach their Cloud vCPEs. Then it is necessary for the vCPEs to communicate with the controller on how traffic is distributed among multiple heterogeneous underlay networks and to manage secure tunnels over untrusted networks.

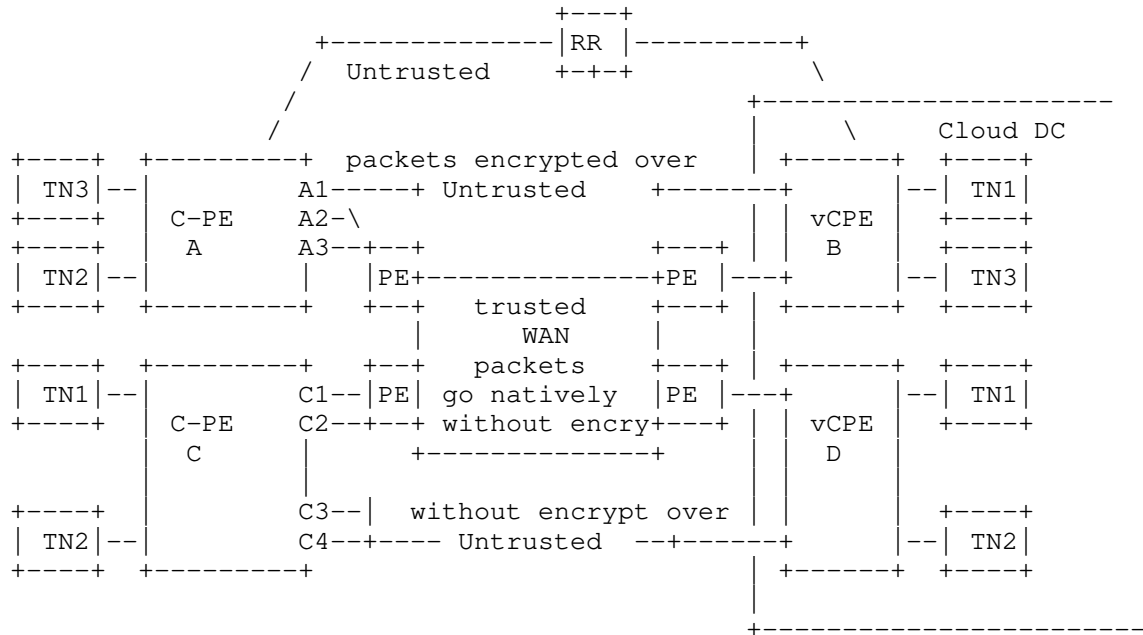


Figure 2: vCPEs reached by Hybrid Paths

### 5.1. Control Plane for Cloud Access via Heterogeneous Networks

The Control Plane for managing applications and workloads in cloud DCs reachable by heterogeneous networks need to include the following properties:

- vCPE in a cloud DCs needs to communicate with its controller of the properties of the directly connected underlay networks.
- Need Controller-facilitated IPsec SA attributes and NAT information distribution
  - o The controller facilitates and manages the peer authentication for all IPsec tunnels terminated at the vCPEs.
- Establishing and Managing the topology and reachability for services attached to the vCPEs in Cloud DCs.
  - o This is for the overlay layer's route distribution, so that a vCPE can populate its overlay routing table with

entries that identify the next hop for reaching a specific route/service attached to the vCPEs.

## 5.2. Using BGP UPDATE Messages

### 5.2.1. Lack ways to differentiate traffic in Cloud DCs

One enterprise can have different types of applications in one Cloud DC. Some can be production applications, some can be testing applications, and some can belong to one specific departments. The traffic to/from different applications might need to traverse different network paths or need to be differentiated by Control plane and data plane.

BGP already has built-in mechanisms, like Route Target, to differentiate different VPNs. But Route Target (RT) is for MPLS based VPNs, therefore RT is not appropriate to directly apply to virtual paths laid over mixed VPNs, IPsec or public underlay networks.

### 5.2.2. Miss attributes in Tunnel-Encap

[Tunnel-Encap] describes the BGP UPDATE Tunnel Path Attribute that advertises endpoints' tunnel encapsulation capabilities for the respective attached client routes encoded in the MP-NLRI Path Attribute. The receivers of the BGP UPDATE can use any of the supported encapsulations encoded in the Tunnel Path Attribute for the routes encoded in the MP-NLRI Path Attribute.

Here are some of the issues raised by using [Tunnel-Encap] to distribute the property of client routes be carried by mixed of hybrid networks:

- [Tunnel-Encap] doesn't have encoding methods to advertise that a route can be carried by mixed of IPsec tunnels and other already supported tunnels.
- The mechanism defined in [Tunnel-Encap] does not facilitate the exchange of IPsec SA-specific attributes.

## 5.3. SECURE-EVPN/BGP-EDGE-DISCOVERY

[SECURE-EVPN] describes a solution that utilize BGP as control plane for the Scenario #1 described in [BGP-SDWAN-Usage]. It relies upon a

BGP cluster design to facilitate the key and policy exchange among PE devices to create private pair-wise IPsec Security Associations. [Secure-EVPN] attaches all the IPsec SA information to the actual client routes.

[BGP-Edge-DISCOVERY] proposes BGP UPDATES from client routers only include the IPsec SA identifiers (ID) to reference the IPsec SA attributes being advertised by separate Underlay Property BGP UPDATE messages. If a client route can be encrypted by multiple IPsec SAs, then multiple IPsec SA IDs are included in the Tunnel-Encap Path attribute for the client route.

[BGP-Edge-DISCOVERY] proposes detailed IPsec SA attributes are advertised in a separate BGP UPDATE for the underlay networks.

[Secure-EVPN] and [BGP-Edge-Discovery] differs in the information included in the client routes. [Secure-EVPN] attaches all the IPsec SA information to the actual client routes, whereas the [BGP-Edge-Discovery] only includes the IPsec SA IDs for the client routes. The IPsec SA IDs used by [BGP-Edge-Discovery] is pointing to the SA-Information which are advertised separately, with all the SA-Information attached to routes which describe the SDWAN underlay, such as WAN Ports or Node address.

#### 5.4. SECURE-L3VPN

[SECURE-L3VPN] describes a method to enrich BGP/MPLS VPN [RFC4364] capabilities to allow some PEs to connect to other PEs via public networks. [SECURE-L3VPN] introduces the concept of Red Interface & Black Interface used by PEs, where the RED interfaces are used to forward traffic into the VPN, and the Black Interfaces are used between WAN ports through which only IPsec-formatted packets are forwarded to the Internet or to any other backbone network, thereby eliminating the need for MPLS transport in the backbone.

[SECURE-L3VPN] assumes PEs use MPLS over IPsec when sending traffic through the Black Interfaces.

[SECURE-L3VPN] is useful, but it misses the aspects of aggregating VPN and Internet underlays. In addition:

- The [SECURE-L3VPN] assumes that a CPE "registers" with the RR. However, it does not say how. It assumes that the remote CPEs are pre-configured with the IPsec SA manually. For overlay networks to connect Hybrid Cloud DCs, Zero Touch Provisioning is expected. Manual configuration is not an option.

- The [SECURE-L3VPN] assumes that C-PEs and RRs are connected via an IPsec tunnel. For management channel, TLS/DTLS is more economical than IPsec. The following assumption made by [SECURE-L3VPN] can be difficult to meet in the environment where zero touch provisioning is expected:

- A CPE must also be provisioned with whatever additional information is needed in order to set up an IPsec SA with each of the red RRs

- IPsec requires periodic refreshment of the keys. The [SECURE-L3VPN] does not provide any information about how to synchronize the refreshment among multiple nodes.
- IPsec usually sends configuration parameters to two endpoints only and lets these endpoints negotiate the key. The [SECURE-L3VPN] assumes that the RR is responsible for creating/managing the key for all endpoints. When one endpoint is compromised, all other connections may be impacted.

#### 5.5. Preventing attacks from Internet-facing ports

When C-PEs have Internet-facing ports, additional security risks are raised.

To mitigate security risks, in addition to requiring Anti-DDoS features on C-PEs, it is necessary for C-PEs to support means to determine whether traffic sent by remote peers is legitimate to prevent spoofing attacks, in particular.

#### 6. Gap Summary

Here is the summary of the technical gaps discussed in this document:

- For Accessing Cloud Resources
  - a) Traffic Path Management: when a remote vCPE can be reached by multiple PEs of one provider VPN network, it is not

straightforward to designate which egress PE to the remote vCPE based on applications or performance.

- b) NAT Traversal: There is no automatic way for an enterprise's network controller to be informed of the NAT properties for its workloads in Cloud DCs.
- c) There is no loop prevention for the multicast traffic to/from remote vCPE in Cloud DCs.

Needs a feature like Appointed Forwarder specified by TRILL to prevent multicast data frames from looping around.

- d) BGP between PEs and remote CPEs via untrusted networks.

- Missing control plane to manage the propagation of the property of networks connected to the virtual nodes in Cloud DCs.

BGP UPDATE propagate client's routes information, but don't distinguish underlay networks.

- Issues of aggregating traffic over private paths and Internet paths

- a) Control plane messages for different overlay segmentations needs to be differentiated. User traffic belonging to different segmentations need to be differentiated.
- b) BGP Tunnel Encap doesn't have ways to indicate a route or prefix that can be carried by both IPsec tunnels and VPN tunnels
- c) Missing clear methods in preventing attacks from Internet-facing ports

## 7. Manageability Considerations

Zero touch provisioning of overlay networks to interconnect Hybrid Clouds is highly desired. It is necessary for a newly powered up edge node to establish a secure connection (by means of TLS, DTLS, etc.) with its controller.



## 8. Security Considerations

Cloud Services are built upon shared infrastructures, therefore not secure by nature.

Secure user identity management, authentication, and access control mechanisms are important. Developing appropriate security measurements can enhance the confidence needed by enterprises to fully take advantage of Cloud Services.

## 9. IANA Considerations

This document requires no IANA actions. RFC Editor: Please remove this section before publication.

## 10. References

### 10.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

### 10.2. Informative References

[RFC8192] S. Hares, et al, "Interface to Network Security Functions (I2NSF) Problem Statement and Use Cases", July 2017

[RFC5521] P. Mohapatra, E. Rosen, "The BGP Encapsulation Subsequent Address Family Identifier (SAFI) and the BGP Tunnel Encapsulation Attribute", April 2009.

[BGP-EDGE-DISCOVERY] L. Dunbar, et al, "BGP UPDATE for SDWAN Edge Discovery ", draft-dunbar-idr-sdwan-edge-discovery-00, Work-in-progress, July 2020.

[BGP-SDWAN-Usage] L. Dunbar, et al, "BGP Usage for SDWAN Overlay Networks ", draft-dunbar-bess-bgp-sdwan-usage-08, work-in-progress, July 2020.

- [Tunnel-Encap] K. Patel, et al, "The BGP Tunnel Encapsulation Attribute", draft-ietf-idr-tunnel-encaps-17, July 2020.
- [SECURE-EVPN] A. Sajassi, et al, draft-sajassi-bess-secure-evpn-01, work in progress, March 2019.
- [SECURE-L3VPN] E. Rosen, "Provide Secure Layer L3VPNs over Public Infrastructure", draft-rosen-bess-secure-l3vpn-00, work-in-progress, July 2018
- [DMVPN] Dynamic Multi-point VPN:  
<https://www.cisco.com/c/en/us/products/security/dynamic-multipoint-vpn-dmvpn/index.html>
- [DSVPN] Dynamic Smart VPN:  
<http://forum.huawei.com/enterprise/en/thread-390771-1-1.html>
- [ITU-T-X1036] ITU-T Recommendation X.1036, "Framework for creation, storage, distribution and enforcement of policies for network security", Nov 2007.
- [Net2Cloud-Problem] L. Dunbar and A. Malis, "Seamless Interconnect Underlay to Cloud Overlay Problem Statement", draft-dm-net2cloud-problem-statement-02, June 2018

## 11. Acknowledgments

Acknowledgements to John Drake for his review and contributions. Many thanks to John Scudder for stimulating the clarification discussion on the Tunnel-Encap draft so that our gap analysis can be more accurate.

This document was prepared using 2-Word-v2.0.template.dot.

Authors' Addresses

Linda Dunbar  
Futurewei  
Email: ldunbar@futurewei.com

Andrew G. Malis  
Malis Consulting  
Email: agmalis@gmail.com

Christian Jacquenet  
Orange  
Rennes, 35000  
France  
Email: Christian.jacquenet@orange.com



Network Working Group  
Internet Draft  
Intended status: Informational  
Expires: September 3, 2022

L. Dunbar  
Futurewei  
Andy Malis  
Malis Consulting  
C. Jacquenet  
Orange  
M. Toy  
Verizon  
March 7, 2022

Dynamic Networks to Hybrid Cloud DCs Problem Statement  
draft-ietf-rtgwg-net2cloud-problem-statement-12

Abstract

This document describes the problems that enterprises face today when interconnecting their branch offices with dynamic workloads in third party data centers (a.k.a. Cloud DCs). There can be many problems associated with network connecting to or among Clouds, many of which probably are out of the IETF scope. The objective of this document is to identify some of the problems that need additional work in IETF Routing area. Other problems are out of the scope of this document.

This document focuses on the network problems that many enterprises face when they have workloads & applications & data split among different data centers, especially for those enterprises with multiple sites that are already interconnected by VPNs (e.g., MPLS L2VPN/L3VPN).

Current operational problems are examined to determine whether there is a need to improve existing protocols or whether a new protocol is necessary to solve them.

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), its areas, and its working groups. Note that

other groups may also distribute working documents as Internet-Drafts.

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."

The list of current Internet-Drafts can be accessed at <http://www.ietf.org/ietf/lid-abstracts.txt>

The list of Internet-Draft Shadow Directories can be accessed at <http://www.ietf.org/shadow.html>

This Internet-Draft will expire on September 7, 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 (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

#### Table of Contents

1. Introduction.....	3
1.1. Key Characteristics of Cloud Services:.....	3
1.2. Connecting to Cloud Services.....	3
1.3. Reaching App instances in the optimal Cloud DC locations..	4
2. Definition of terms.....	5
3. High Level Issues of Connecting to Multi-Cloud.....	6
3.1. 5G Edge Clouds.....	6
3.2. Security Issues.....	6
3.3. Authorization and Identity Management.....	7

3.4. API abstraction.....	7
3.5. DNS for Cloud Resources.....	8
3.6. NAT for Cloud Services.....	9
3.7. Cloud Discovery.....	10
4. Interconnecting Enterprise Sites with Cloud DCs.....	10
4.1. Sites to Cloud DC.....	10
4.2. Inter-Cloud Interconnection.....	12
5. Edge Clouds.....	14
6. Problems with MPLS-based VPNs extending to Hybrid Cloud DCs....	14
7. Problem with using IPsec tunnels to Cloud DCs.....	15
7.1. Scaling Issues with IPsec Tunnels.....	16
7.2. Poor performance over long distance.....	16
8. End-to-End Security Concerns for Data Flows.....	16
9. Requirements for Dynamic Cloud Data Center VPNs.....	17
10. Security Considerations.....	17
11. IANA Considerations.....	18
12. References.....	18
12.1. Normative References.....	18
12.2. Informative References.....	18
13. Acknowledgments.....	18

## 1. Introduction

### 1.1. Key Characteristics of Cloud Services:

Key characteristics of Cloud Services are on-demand, scalable, highly available, and usage-based billing. Cloud Services, such as, compute, storage, network functions (most likely virtual), third party managed applications, etc. are usually hosted and managed by third parties Cloud Operators. Here are some examples of Cloud network functions: Virtual Firewall services, Virtual private network services, Virtual PBX services including voice and video conferencing systems, etc. Cloud Data Center (DC) is shared infrastructure that hosts the Cloud Services to many customers.

### 1.2. Connecting to Cloud Services

With the advent of widely available third-party cloud DCs and services in diverse geographic locations and the advancement of tools for monitoring and predicting application behaviors, it is very attractive for enterprises to instantiate applications and workloads in locations that are geographically closest to their end-users. Such proximity can improve end-to-end latency and overall user experience. Conversely, an enterprise can easily shutdown

applications and workloads whenever end-users are in motion (thereby modifying the networking connection of subsequently relocated applications and workloads). In addition, enterprises may wish to take advantage of more and more business applications offered by cloud operators.

The networks that interconnect hybrid cloud DCs must address the following requirements:

- to access all workloads in the desired cloud DCs:  
Many enterprises include cloud in their disaster recovery strategy, such as enforcing periodic backup policies within the cloud, or running backup applications in the Cloud.
- Global reachability from different geographical zones, thereby facilitating the proximity of applications as a function of the end users' location, to improve latency.
- Elasticity: prompt connection to newly instantiated applications at Cloud DCs when usages increase and prompt release of connection after applications at locations being removed when demands change.
- Scalable policy management: apply the appropriate policies to the newly instantiated application instances at any Cloud DC location.

### 1.3. Reaching App instances in the optimal Cloud DC locations

Many applications have multiple instances instantiated in different Cloud DCs. The current state of the art solutions is typically based on DNS assisted with load balancer by responding a FQDN (Fully Qualified Domain Name) inquiry with an IP address of the closest or lowest cost DC that can reach the instance. Here are some problems associated with DNS based solutions:

- Dependent on client behavior
  - Client can cache results indefinitely
  - Client may not receive service even though there are servers available (before cache timeout) in other Cloud DCs.



- No inherent leverage of proximity information present in the network (routing) layer, resulting in loss of performance
  - Client on the west coast can be mapped to a DC on the east coast
- Inflexible traffic control:
  - Local DNS resolver become the unit of traffic management. This requires DNS to receive periodical update of the network condition, which is difficult.

## 2. Definition of terms

**Cloud DC:** Third party Data Centers that usually host applications and workload owned by different organizations or tenants.

**Controller:** Used interchangeably with SD-WAN controller to manage SD-WAN overlay path creation/deletion and monitoring the path conditions between two or more sites.

**DSVPN:** Dynamic Smart Virtual Private Network. DSVPN is a secure network that exchanges data between sites without needing to pass traffic through an organization's headquarter virtual private network (VPN) server or router.

**Heterogeneous Cloud:** applications and workloads split among Cloud DCs owned or managed by different operators.

**Hybrid Clouds:** Hybrid Clouds refers to an enterprise using its own on-premises DCs in addition to Cloud services provided by one or more cloud operators. (e.g. AWS, Azure, Google, Salesforces, SAP, etc).

**VPC:** Virtual Private Cloud is a virtual network dedicated to one client account. It is logically isolated from other virtual networks in a Cloud DC. Each client can launch his/her desired resources, such as compute, storage, or network functions into his/her VPC. Most Cloud

operators' VPCs only support private addresses, some support IPv4 only, others support IPv4/IPv6 dual stack.

### 3. High Level Issues of Connecting to Multi-Cloud

There are many problems associated with connecting to hybrid Cloud Services, many of which are out of the IETF scope. This section is to identify some of the high-level problems that can be addressed by IETF, especially by Routing area. Other problems are out of the scope of this document. By no means has this section covered all problems for connecting to Hybrid Cloud Services, e.g. difficulty in managing cloud spending is not discussed here.

#### 3.1. 5G Edge Clouds

5G edge cloud data centers have routers connecting to the 5G Core functions, such as Radio Control Functions, Session Management Function (SMF), Access Mobility Functions (AMF), User Plane Functions (UPF), etc. Those functions need to be connected to the Radio Data Unit (R-DU) on the Cell Tower. The UPFs need to be connected to the 5G Local Data Networks' ingress routers which might co-located the cloud edge data centers.

In addition, the 5G edge cloud data centers may host edge computing servers for Ultra-low latency services that need to be near the UEs (User equipment). Those edge computing applications need to have very low latency to the UEs, and also connect to backend servers or databases in another location.

#### 3.2. Security Issues

Cloud Services is built upon shared infrastructure, therefore not secure by nature. Security has been a primary, and valid, concern from the start of cloud computing, e.g. not being able to see the exact location where the data are stored or trace of access. Headlines highlighting data breaches, compromised credentials, and broken authentication, hacked interfaces and APIs, account hijacking haven't helped alleviate concerns.

Many Cloud operators offer monitoring services for data stored in Clouds, such as AWS CloudTrail, Azure Monitor, and many third-party monitoring tools to improve visibility to data stored in Clouds. But

there is still underline security concerns on illegitimate data and workloads access.

Secure user identity management, authentication, and access control mechanisms are important. Developing appropriate security measurements can enhance the confidence needed by enterprises to fully take advantage of Cloud Services.

### 3.3. Authorization and Identity Management

One of the more prominent challenges for Cloud Services is Identity Management and Authorization. The Authorization not only includes user authorization, but also the authorization of API calls by applications from different Cloud DCs managed by different Cloud Operators. In addition, there are authorization for Workload Migration, Data Migration, and Workload Management.

There are many types of users in cloud environments, e.g. end users for accessing applications hosted in Cloud DCs, Cloud-resource users who are responsible for setting permissions for the resources based on roles, access lists, IP addresses, domains, etc.

There are many types of Cloud authorizations: including MAC (Mandatory Access Control) - where each app owns individual access permissions, DAC (Discretionary Access Control) - where each app requests permissions from an external permissions app, RBAC (Role-based Access Control) - where the authorization service owns roles with different privileges on the cloud service, and ABAC (Attribute-based Access Control) - where access is based on request attributes and policies.

IETF hasn't yet developed comprehensive specification for Identity management and data models for Cloud Authorizations.

### 3.4. API abstraction

Different Cloud Operators have different APIs to access their Cloud resources, security functions, the NAT, etc.

It is difficult to move applications built by one Cloud operator's APIs to another. However, it is highly desirable to have a single and consistent way to manage the networks and respective security policies for interconnecting applications hosted in different Cloud DCs.

The desired property would be having a single network fabric to which different Cloud DCs and enterprise's multiple sites can be attached or detached, with a common interface for setting desired policies.

The difficulty of connecting applications in different Clouds might be stemmed from the fact that they are direct competitors. Usually traffic flow out of Cloud DCs incur charges. Therefore, direct communications between applications in different Cloud DCs can be more expensive than intra Cloud communications.

It is desirable to have a common API shim layer or abstraction for different Cloud providers to make it easier to move applications from one Cloud DC to another.

### 3.5. DNS for Cloud Resources

DNS name resolution is essential for on-premises and cloud-based resources. For customers with hybrid workloads, which include on-premises and cloud-based resources, extra steps are necessary to configure DNS to work seamlessly across both environments.

Cloud operators have their own DNS to resolve resources within their Cloud DCs and to well-known public domains. Cloud's DNS can be configured to forward queries to customer managed authoritative DNS servers hosted on-premises, and to respond to DNS queries forwarded by on-premises DNS servers.

For enterprises utilizing Cloud services by different cloud operators, it is necessary to establish policies and rules on how/where to forward DNS queries to. When applications in one Cloud need to communication with applications hosted in another Cloud, there could be DNS queries from one Cloud DC being forwarded to the enterprise's on-premise DNS, which in turn be forwarded to the DNS service in another Cloud. Needless to say, configuration can be complex depending on the application communication patterns.

However, even with carefully managed policies and configurations, collisions can still occur. If you use an internal name like `.cloud` and then want your services to be available via or within some other cloud provider which also uses `.cloud`, then it can't work. Therefore, it is better to use the global domain name even when an organization does not make all its namespace globally resolvable. An organization's globally unique DNS can include subdomains that cannot be resolved at all outside certain restricted paths, zones that resolve differently based on the origin of the query, and zones that resolve the same globally for all queries from any source.

Globally unique names do not equate to globally resolvable names or even global names that resolve the same way from every perspective. Globally unique names do prevent any possibility of collision at the present or in the future and they make DNSSEC trust manageable. Consider using a registered and fully qualified domain name (FQDN) from global DNS as the root for enterprise and other internal namespaces.

### 3.6. NAT for Cloud Services

Cloud resources, such as VM instances, are usually assigned with private IP addresses. By configuration, some private subnets can have the NAT function to reach out to external network and some private subnets are internal to Cloud only.

Different Cloud operators support different levels of NAT functions. For example, AWS NAT Gateway does not currently support connections towards, or from VPC Endpoints, VPN, AWS Direct Connect, or VPC Peering. <https://docs.aws.amazon.com/AmazonVPC/latest/UserGuide/vpc-nat-gateway.html#nat-gateway-other-services>. AWS Direct Connect/VPN/VPC Peering does not currently support any NAT functionality.

Google's Cloud NAT allows Google Cloud virtual machine (VM) instances without external IP addresses and private Google Kubernetes Engine (GKE) clusters to connect to the Internet. Cloud NAT implements outbound NAT in conjunction with a default route to allow instances to reach the Internet. It does not implement inbound NAT. Hosts outside of VPC network can only respond to established connections initiated by instances inside the Google Cloud; they cannot initiate their own, new connections to Cloud instances via NAT.

For enterprises with applications running in different Cloud DCs, proper configuration of NAT has to be performed in Cloud DC and in their on-premises DC.

### 3.7. Cloud Discovery

One of the concerns of using Cloud services is not aware where the resource is located, especially Cloud operators can move application instances from one place to another. When applications in Cloud communicate with on-premise applications, it may not be clear where the Cloud applications are located or to which VPCs they belong.

It is highly desirable to have tools to discover cloud services in much the same way as you would discover your on-premises infrastructure. A significant difference is that cloud discovery uses the cloud vendor's API to extract data on your cloud services, rather than the direct access used in scanning your on-premises infrastructure.

Standard data models, APIs or tools can alleviate concerns of enterprise utilizing Cloud Resources, e.g. having a Cloud service scan that connects to the API of the cloud provider and collects information directly.

## 4. Interconnecting Enterprise Sites with Cloud DCs

Considering that many enterprises already have existing VPNs (e.g. MPLS based L2VPN or L3VPN) interconnecting branch offices & on-premises data centers, connecting to Cloud services will be mixed of different types of networks. When an enterprise's existing VPN service providers do not have direct connections to the corresponding cloud DCs that the enterprise prefers to use, the enterprise has to face additional infrastructure and operational costs to utilize the Cloud services.

### 4.1. Sites to Cloud DC

Most Cloud operators offer some type of network gateway through which an enterprise can reach their workloads hosted in the Cloud DCs. AWS (Amazon Web Services) offers the following options to reach workloads in AWS Cloud DCs:

- AWS Internet gateway allows communication between instances in AWS VPC and the internet.
- AWS Virtual gateway (vGW) where IPsec tunnels [RFC6071] are established between an enterprise's own gateway and AWS vGW, so that the communications between those gateways can be secured from the underlay (which might be the public Internet).
- AWS Direct Connect, which allows enterprises to purchase direct connect from network service providers to get a private leased line interconnecting the enterprises gateway(s) and the AWS Direct Connect routers. In addition, an AWS Transit Gateway can be used to interconnect multiple VPCs in different Availability Zones. AWS Transit Gateway acts as a hub that controls how traffic is forwarded among all the connected networks which act like spokes.

Microsoft's ExpressRoute allows extension of a private network to any of the Microsoft cloud services, including Azure and Office365. ExpressRoute is configured using Layer 3 routing. Customers can opt for redundancy by provisioning dual links from their location to two Microsoft Enterprise edge routers (MSEEs) located within a third-party ExpressRoute peering location. The BGP routing protocol is then setup over WAN links to provide redundancy to the cloud. This redundancy is maintained from the peering data center into Microsoft's cloud network.

Google's Cloud Dedicated Interconnect offers similar network connectivity options as AWS and Microsoft. One distinct difference, however, is that Google's service allows customers access to the entire global cloud network by default. It does this by connecting your on-premises network with the Google Cloud using BGP and Google Cloud Routers to provide optimal paths to the different regions of the global cloud infrastructure.

Figure below shows an example of some of a tenant's workloads are accessible via a virtual router connected by AWS Internet Gateway; some are accessible via AWS vGW, and others are accessible via AWS Direct Connect.

Different types of access require different level of security functions. Sometimes it is not visible to end customers which type of network access is used for a specific application instance. To get better visibility, separate virtual routers (e.g. vR1 & vR2) can be deployed to differentiate traffic to/from different cloud GWs. It

is important for some enterprises to be able to observe the specific behaviors when connected by different connections.

Customer Gateway can be customer owned router or ports physically connected to AWS Direct Connect GW.

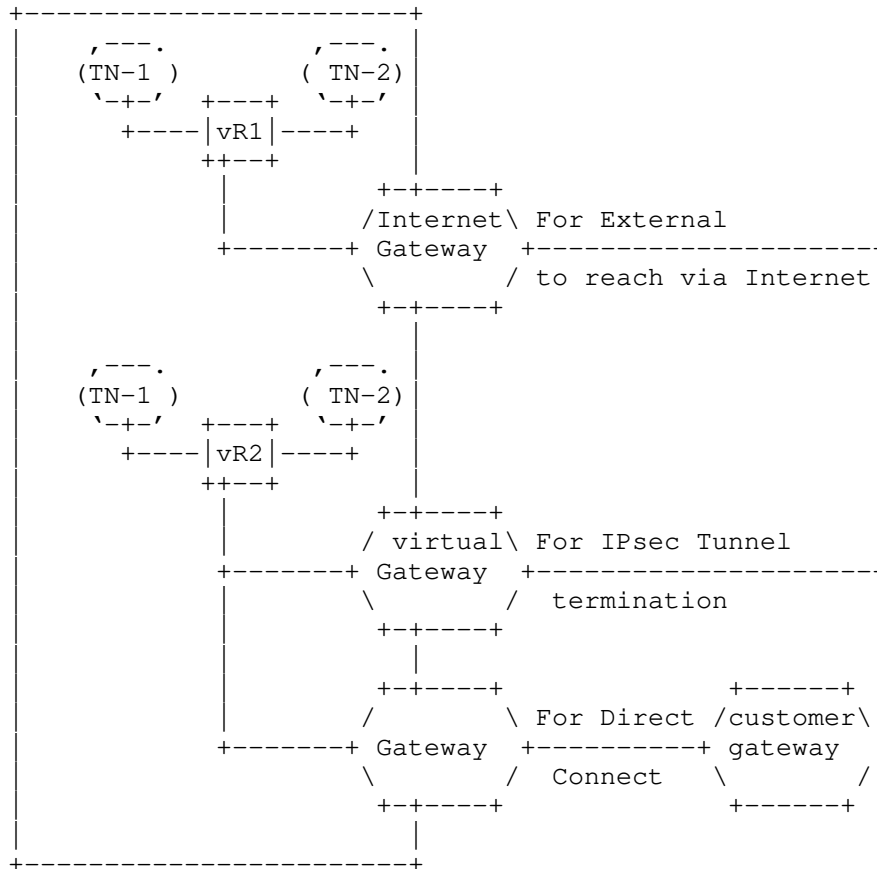


Figure 1: Examples of Multiple Cloud DC connections.

#### 4.2. Inter-Cloud Interconnection

The connectivity options to Cloud DCs described in the previous section are for reaching Cloud providers' DCs, but not between cloud DCs. When applications in AWS Cloud need to communicate with applications in Azure, today's practice requires a third-party gateway (physical or virtual) to interconnect the AWS's Layer 2 DirectConnect path with Azure's Layer 3 ExpressRoute.



Enterprises can also instantiate their own virtual routers in different Cloud DCs and administer IPsec tunnels among them, which by itself is not a trivial task. Or by leveraging open source VPN software such as strongSwan, you create an IPsec connection to the Azure gateway using a shared key. The StrongSwan instance within AWS not only can connect to Azure but can also be used to facilitate traffic to other nodes within the AWS VPC by configuring forwarding and using appropriate routing rules for the VPC.

Most Cloud operators, such as AWS VPC or Azure VNET, use non-globally routable CIDR from private IPv4 address ranges as specified by RFC1918. To establish IPsec tunnel between two Cloud DCs, it is necessary to exchange Public routable addresses for applications in different Cloud DCs.

In summary, here are some approaches, available now (which might change in the future), to interconnect workloads among different Cloud DCs:

- a) Utilize Cloud DC provided inter/intra-cloud connectivity services (e.g., AWS Transit Gateway) to connect workloads instantiated in multiple VPCs. Such services are provided with the cloud gateway to connect to external networks (e.g., AWS DirectConnect Gateway).
- b) Hairpin all traffic through the customer gateway, meaning all workloads are directly connected to the customer gateway, so that communications among workloads within one Cloud DC must traverse through the customer gateway.
- c) Establish direct tunnels among different VPCs (AWS' Virtual Private Clouds) and VNET (Azure's Virtual Networks) via client's own virtual routers instantiated within Cloud DCs. DMVPN (Dynamic Multipoint Virtual Private Network) or DSVPN (Dynamic Smart VPN) techniques can be used to establish direct Multi-point-to-Point or multi-point-to multi-point tunnels among those client's own virtual routers.

Approach a) usually does not work if Cloud DCs are owned and managed by different Cloud providers.

Approach b) creates additional transmission delay plus incurring cost when exiting Cloud DCs.

For the Approach c), DMVPN or DSVPN use NHRP (Next Hop Resolution Protocol) [RFC2735] so that spoke nodes can register their IP

addresses & WAN ports with the hub node. The IETF ION (Internetworking over NBMA (non-broadcast multiple access) WG standardized NHRP for connection-oriented NBMA network (such as ATM) network address resolution more than two decades ago.

There are many differences between virtual routers in Public Cloud DCs and the nodes in an NBMA network. NHRP cannot be used for registering virtual routers in Cloud DCs unless an extension of such protocols is developed for that purpose, e.g. taking NAT or dynamic addresses into consideration. Therefore, DMVPN and/or DSVPN cannot be used directly for connecting workloads in hybrid Cloud DCs.

## 5. Edge Clouds

## 6. Problems with MPLS-based VPNs extending to Hybrid Cloud DCs

Traditional MPLS-based VPNs have been widely deployed as an effective way to support businesses and organizations that require network performance and reliability. MPLS shifted the burden of managing a VPN service from enterprises to service providers. The CPEs attached to MPLS VPNs are also simpler and less expensive, because they do not need to manage routes to remote sites; they simply pass all outbound traffic to the MPLS VPN PEs to which the CPEs are attached (albeit multi-homing scenarios require more processing logic on CPEs). MPLS has addressed the problems of scale, availability, and fast recovery from network faults, and incorporated traffic-engineering capabilities.

However, traditional MPLS-based VPN solutions are sub-optimized for connecting end-users to dynamic workloads/applications in cloud DCs because:

- The Provider Edge (PE) nodes of the enterprise's VPNs might not have direct connections to third party cloud DCs that are used for hosting workloads with the goal of providing an easy access to enterprises' end-users.
- It takes some time to deploy provider edge (PE) routers at new locations. When enterprise's workloads are changed from one cloud DC to another (i.e., removed from one DC and re-instantiated to another location when demand changes), the

enterprise branch offices need to be connected to the new cloud DC, but the network service provider might not have PEs located at the new location.

One of the main drivers for moving workloads into the cloud is the widely available cloud DCs at geographically diverse locations, where apps can be instantiated so that they can be as close to their end-users as possible. When the user base changes, the applications may be migrated to a new cloud DC location closest to the new user base.

- Most of the cloud DCs do not expose their internal networks. An enterprise with a hybrid cloud deployment can use an MPLS-VPN to connect to a Cloud provider at multiple locations. The connection locations often correspond to gateways of different Cloud DC locations from the Cloud provider. The different Cloud DCs are interconnected by the Cloud provider's own internal network. At each connection location (gateway), the Cloud provider uses BGP to advertise all of the prefixes in the enterprise's VPC, regardless of which Cloud DC a given prefix is actually in. This can result in inefficient routing for the end-to-end data path.

Another roadblock is the lack of a standard way to express and enforce consistent security policies for workloads that not only use virtual addresses, but in which are also very likely hosted in different locations within the Cloud DC [RFC8192]. The current VPN path computation and bandwidth allocation schemes may not be flexible enough to address the need for enterprises to rapidly connect to dynamically instantiated (or removed) workloads and applications regardless of their location/nature (i.e., third party cloud DCs).

#### 7. Problem with using IPsec tunnels to Cloud DCs

As described in the previous section, many Cloud operators expose their gateways for external entities (which can be enterprises themselves) to directly establish IPsec tunnels. Enterprises can also instantiate virtual routers within Cloud DCs to connect to their on-premises devices via IPsec tunnels.

### 7.1. Scaling Issues with IPsec Tunnels

If there is only one enterprise location that needs to reach the Cloud DC, an IPsec tunnel is a very convenient solution.

However, many medium-to-large enterprises have multiple sites and multiple data centers. For multiple sites to communicate with workloads and apps hosted in cloud DCs, Cloud DC gateways have to maintain many IPsec tunnels to all those locations. In addition, each of those IPsec Tunnels requires pair-wise periodic key refreshment. For a company with hundreds or thousands of locations, there could be hundreds (or even thousands) of IPsec tunnels terminating at the cloud DC gateway, which is very processing intensive. That is why many cloud operators only allow a limited number of (IPsec) tunnels & bandwidth to each customer.

Alternatively, you could use a solution like group encryption where a single IPsec SA is necessary at the GW but the drawback is key distribution and maintenance of a key server, etc.

### 7.2. Poor performance over long distance

When enterprise CPEs or gateways are far away from cloud DC gateways or across country/continent boundaries, performance of IPsec tunnels over the public Internet can be problematic and unpredictable. Even though there are many monitoring tools available to measure delay and various performance characteristics of the network, the measurement for paths over the Internet is passive and past measurements may not represent future performance.

Many cloud providers can replicate workloads in different available zones. An App instantiated in a cloud DC closest to clients may have to cooperate with another App (or its mirror image) in another region or database server(s) in the on-premises DC. This kind of coordination requires predictable networking behavior/performance among those locations.

## 8. End-to-End Security Concerns for Data Flows

When IPsec tunnels established from enterprise on-premises CPEs are terminated at the Cloud DC gateway where the workloads or applications are hosted, some enterprises have concerns regarding traffic to/from their workload being exposed to others behind the data center gateway (e.g., exposed to other organizations that have workloads in the same data center).

To ensure that traffic to/from workloads is not exposed to unwanted entities, IPsec tunnels may go all the way to the workload (servers, or VMs) within the DC.

#### 9. Requirements for Dynamic Cloud Data Center VPNs

To address the aforementioned issues, any solution for enterprise VPNs that includes connectivity to dynamic workloads or applications in cloud data centers should satisfy a set of requirements:

- The solution should allow enterprises to take advantage of the current state-of-the-art in VPN technology, in both traditional MPLS-based VPNs and IPsec-based VPNs (or any combination thereof) that run over the public Internet.
- The solution should not require an enterprise to upgrade all their existing CPEs.
- The solution should support scalable IPsec key management among all nodes involved in DC interconnect schemes.
- The solution needs to support easy and fast, on-the-fly, VPN connections to dynamic workloads and applications in third party data centers, and easily allow these workloads to migrate both within a data center and between data centers.
- Allow VPNs to provide bandwidth and other performance guarantees.
- Be a cost-effective solution for enterprises to incorporate dynamic cloud-based applications and workloads into their existing VPN environment.

#### 10. Security Considerations

The draft discusses security requirements as a part of the problem space, particularly in sections 4, 5, and 8.

Solution drafts resulting from this work will address security concerns inherent to the solution(s), including both protocol aspects and the importance (for example) of securing workloads in cloud DCs and the use of secure interconnection mechanisms.

## 11. IANA Considerations

This document requires no IANA actions. RFC Editor: Please remove this section before publication.

## 12. References

### 12.1. Normative References

### 12.2. Informative References

[RFC2735] B. Fox, et al "NHRP Support for Virtual Private networks". Dec. 1999.

[RFC8192] S. Hares, et al "Interface to Network Security Functions (I2NSF) Problem Statement and Use Cases", July 2017

[ITU-T-X1036] ITU-T Recommendation X.1036, "Framework for creation, storage, distribution and enforcement of policies for network security", Nov 2007.

[RFC6071] S. Frankel and S. Krishnan, "IP Security (IPsec) and Internet Key Exchange (IKE) Document Roadmap", Feb 2011.

[RFC4364] E. Rosen and Y. Rekhter, "BGP/MPLS IP Virtual Private Networks (VPNs)", Feb 2006

[RFC4664] L. Andersson and E. Rosen, "Framework for Layer 2 Virtual Private Networks (L2VPNs)", Sept 2006.

## 13. Acknowledgments

Many thanks to Alia Atlas, Chris Bowers, Paul Vixie, Paul Ebersman, Timothy Morizot, Ignas Bagdonas, Michael Huang, Liu Yuan Jiao, Katherine Zhao, and Jim Guichard for the discussion and contributions.

Authors' Addresses

Linda Dunbar  
Futurewei  
Email: Linda.Dunbar@futurewei.com

Andrew G. Malis  
Malis Consulting  
Email: agmalis@gmail.com

Christian Jacquenet  
Orange  
Rennes, 35000  
France  
Email: Christian.jacquenet@orange.com

Mehmet Toy  
Verizon  
One Verizon Way  
Basking Ridge, NJ 07920  
Email: mehmet.toy@verizon.com





Network Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: August 23, 2021

Z. Li  
S. Chen  
Huawei  
Y. Qu  
Futurewei  
Y. Gu  
Huawei  
February 19, 2021

Protocol Assisted Protocol (PASP)  
draft-li-rtgwg-protocol-assisted-protocol-04

Abstract

For routing protocol troubleshooting, different approaches exhibit merits w.r.t. different situations. They can be generally divided into two categories, the distributive way and the centralized way. A very commonly used distributive approach is to log in possibly all related devices one by one to check massive data via CLI. Such approach provides very detailed device information, however it requires operators with high NOC (Network Operation Center) experience and suffers from low troubleshooting efficiency and high cost. The centralized approach is realized by collecting data from devices via approaches, like the streaming Telemetry or BMP (BGP Monitoring Protocol) RFC7854 [RFC7854], for the centralized server to analyze all gathered data. Such approach allows a comprehensive view fo the whole network and facilitates automated troubleshooting, but is limited by the data collection boundary set by different management domains, as well as high network bandwidth and CPU computation costs.

This document proposes a semi-distributive and semi-centralized approach for fast routing protocol troubleshooting, localizing the target device and possibly the root cause, more precisely. It defines a new protocol, called the PASP (Protocol assisted Protocol), for devices to exchange protocol related information between each other in both active and on-demand manners. It allow devices to request specific information from other devices and receive replies to the requested data. It also allows actively transmission of information without request to inform other devices to better react w.r.t. network issues.

Requirements Language

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

## Status of This Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <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 August 23, 2021.

## Copyright Notice

Copyright (c) 2021 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. Motivation . . . . .	3
1.2. PASP Usage Use cases . . . . .	5
1.2.1. Use Case 1: BGP Route Oscillation . . . . .	5
1.2.2. Use Case 2: RSVP-TE Set Up Failure . . . . .	6
2. Terminology . . . . .	6
3. PASP Overview . . . . .	7
3.1. PASP Encapsulation . . . . .	7
3.2. PASP Speaker and PASP Agent . . . . .	7
3.3. PASP Event . . . . .	7
3.4. Summary of Operation . . . . .	8
3.4.1. PASP Capability Negotiation Process . . . . .	8
3.4.2. PASP Request and Reply Process . . . . .	8
3.4.3. PASP Notification Process . . . . .	9

4.	PASP Message Format . . . . .	9
4.1.	Common Header . . . . .	9
4.1.1.	Capability Negotiation Message . . . . .	10
4.2.	Request Message . . . . .	11
4.3.	Reply Message . . . . .	12
4.4.	Notification Message . . . . .	13
4.5.	ACK Message . . . . .	14
5.	PASP Operations . . . . .	14
5.1.	Capability Negotiation Process . . . . .	14
5.1.1.	PASP Peering Relation Establish Process . . . . .	14
5.1.2.	PASP Capability Enabling Notification Process . . . . .	15
5.1.3.	PASP Capability Disabling Notification Process . . . . .	16
5.2.	PASP Request and Reply Process . . . . .	16
5.3.	PASP Notification Process . . . . .	18
6.	PASP Error Handling . . . . .	19
7.	Discussion . . . . .	19
8.	Security Considerations . . . . .	20
9.	IANA . . . . .	20
10.	Contributors . . . . .	20
11.	Acknowledgments . . . . .	20
12.	References . . . . .	20
	Authors' Addresses . . . . .	22

## 1. Introduction

A healthy control plane, providing network connectivity, is the foundation of a well-functioning network. There have been rich routing and signaling protocols designed and used for IP networks, such as IGP (ISIS, OSPF), BGP, LDP, RSVP-TE and so on. The health issues of these protocols, such as neighbor/peer disconnect/set up failure, LSP set up failure, route flapping and so on, have been devoted with ongoing efforts for diagnosing and remediation.

### 1.1. Motivation

The distributive protocol troubleshooting approach is typically realized through manual per-device check. It's both time- and labor-consuming, and requires NOC experience of the operators. Amongst all, localizing the target device is usually the most difficult and time-consuming part. For example, in the case of route loop, operators first log in a random device that reports TTL alarms, and then check the looped route in the Forwarding Information Base (FIB) and/or the Routing Information Base (RIB). It requires device by device check, as well as manual data correlation, to pin point to the exact responsible device, since the information retrieval and analysis of such distributive way is fragmented. In addition, the low efficiency and manual troubleshooting activities may further impact new network services and/or enlarge affected areas.

The centralized network OAM, by collecting network-wide data from devices, enables automatic routing protocol troubleshooting. Data collection protocols, such as SNMP (Simple Network Management Protocol) [RFC1157], NETCONF (Network Configuration Protocol) [RFC6241], and (BMP) [RFC7854], can provide various information retrieval, such as network states, routing data, configurations and so on. Such centralized way relies on the existence of a centralized server/controller, which is not supported by some legacy networks. What's more, even with the existence of a centralized server/controller, it can only collect the data within its own management domain, while the cross-domain data are not available due to independent management of different ISPs. Thus, the lack of such information may lead to troubleshooting failure. In addition, centralized approaches may suffer from high network bandwidth and CPU computation consumptions.

Another way of protocol troubleshooting is utilizing the protocol itself to convey diagnosing information. For example, some reason codes are carried in the Path-Err/ResvErr messages of RSVP-TE, so that other nodes may know the why the tunnel fails to be set up. Such approaches is semi-distributive and semi-centralized. It does not rely on the deployment of a centralized server, but still gets partial global view of the network. However, there still requires non-trivial augmentation works to existing routing protocols in order to support troubleshooting. This then raises the question that whether such non-routing data is suitable to be carried in these routing protocols. The extra encapsulation, parsing and analyzing work for the non-routing data would further slow down the network convergence. Thus, it's better to separate the routing and non-routing data transmission as well as data parsing. In addition, coexisting with legacy devices may cause interop issues. Thus, relying on augmenting existing routing protocols without network-wide upgrading may not only fail to provide the troubleshooting benefit, but further affect the operation of the existing routing system. What's more, the failure of routing protocol instance would lead to the failure of diagnosing itself. All in all, it's reasonable to separate the protocol diagnosing data generation/encapsulation/transmission/parsing from the protocol itself.

This document proposes a new protocol, called the PASP (Protocol assisted Protocol), for devices to exchange protocol related information between each other. It allows both active and on-demand data exchange. Considering that massiveness of protocol/routing related data, the intuitive of designing PASP is not to exchange the comprehensive protocol/routing status between devices, but to provide very specific information required for fast troubleshooting. The

benefits of such a semi-distributive and semi-centralized approach are summarized as follows:

1. It facilitates automatic troubleshooting without requiring manual device by device check.
2. It allows individual device to have a more global view by requesting data from other devices.
3. It does not rely on the deployment of a centralized server/controller.
4. It passes the data collection boundary set by different management domains by cross-domain data exchange between devices.
5. It relieves the bandwidth pressure of network-wide data collection, and the processing pressure of the centralized server.
6. It does not affect the running of existing routing protocols.

#### 1.2. PASP Usage Use cases

PASP allows both data request/reply and data notification between devices. PASP speakers use the exchanged PASP data to help fast localize the network issues.

##### 1.2.1. Use Case 1: BGP Route Oscillation

A BGP route oscillation can be caused by various reasons, and usually leaves network-wide impact. In order to find the root cause and take remediation actions, the first step is to localize the oscillation source. In this case, a BGP speaker can send a PASP Request Message to the next hop device of the oscillating route asking "Are you the oscillation source?". If the BGP speaker is the oscillation source, possibly knows by running a device diagnosing system, replies with a PASP Reply Message saying that "I'm the oscillation source!" to the device who sends the PASP Request Message. If the BGP speaker is not the oscillation source, it further asks the same question with a PASP Request Message to its next hop device of the oscillating route. This request and reply process continues until the request has reached the oscillation source. The source device then sends a PASP Reply Message to tell its upstream device along the PASP request path that "I am the oscillation source!", and then "xx is the oscillation source!" information is further sent back hop by hop to the device who originates the request.

### 1.2.2. Use Case 2: RSVP-TE Set Up Failure

The MPLS label switch path set up, either using RSVP-TE or LDP, may fail due to various reasons. Typical troubleshooting procedures are to log in the device, and then check if the failure lies on the configuration, or path computation error, or link failure. Sometimes, it requires the check of multiple devices along the tunnel. Certain reason codes can be carried in the Path-Err/ResvErr messages of RSVP-TE, while other data are currently not supported to be transmitted to the path ingress/egress node, such as the authentication failure. Using PASP, the device, which is responsible for the tunnel set up failure, can send the PASP Notification Message to the Ingress device, and possibly with some reason codes so that the ingress device can not only localize the target device but also the root cause.

## 2. Terminology

IGP: Interior Gateway Protocol

IS-IS: Intermediate System to Intermediate System

OSPF: Open Shortest Path First

BGP: Boarder Gateway Protocol

BGP-LS: Boarder Gateway Protocol-Link State

MPLS: Multi-Protocol Label Switching

RSVP-TE: Resource Reservation Protocol-Traffic Engineering

LDP: Label Distribution Protocol

BMP: BGP Monitoring Protocol

LSP: Link State Packet

IPFIX: Internet Protocol Flow Information Export

PASP: Protocol assisted Protocol

UDP: User Datagram Protocol

### 3. PASP Overview

#### 3.1. PASP Encapsulation

PASP uses UDP as its transport protocol, which is connectionless. The reason that UDP is selected over TCP is because PASP is intended for on-demand communications. The PASP packet is defined as follows. This document requires the assignment of a User Port registry for the UDP Destination Port.

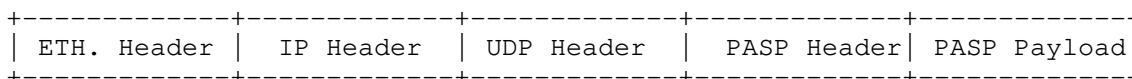


Figure 1. Encapsulation in UDP

#### 3.2. PASP Speaker and PASP Agent

This document uses PASP speakers to refer to routing devices that communicate with each other using PASP. PASP speakers SHOULD be implemented with a supporting module (or multiple modules) to receive, parse, analyze, generate, and send PASP messages. For example, a BGP diagnosing module used for BGP related PASP message handling functions as a PASP agent. A PASP Agent is the union of multiple such modules regarding different protocols, or one module for all protocols. Such supporting module is called PASP Agent in this document. PASP Agent, standalone, SHOULD be able to provide protocol troubleshooting capability with local information. Enabling PASP exchange capability, PASP agent gains information from remote PASP speakers to improve diagnosing accuracy. The primary function of PASP is to provide a unified tunnel for protocol diagnosing information exchange without augmenting each specific protocol.

#### 3.3. PASP Event

A PASP Event is referred to as the a troubleshooting instance running within a PASP Agent. A PASP Agent may instantiate one or multiple PASP Events for each protocol at the same time depending on the configured troubleshooting triggering condition. For example, an PASP Event is initiated automatically when device CPU is over high, or manually with related command line input from a device operator. Once a PASP Event is generated, corresponding PASP processes are to be called on demand. Notice, the initiation of PASP Capability Negotiation does not require the existance of a PASP Event.

### 3.4. Summary of Operation

The communications between two PASP speakers should follow three major processes, i.e., the Capability Negotiation Process, the Request and Reply Process, and the Notification Process. This document defines 5 PASP Message types, i.e., Negotiation Message, Request Message, Reply Message, Notification Message, and ACK Message, which are used in the above PASP processes.

#### 3.4.1. PASP Capability Negotiation Process

The purpose of the Capability Negotiation process is to inform two PASP speakers of each other's PASP capabilities. The PASP capability indicates, for which specific protocol(s), that PASP supports its/their diagnosing information exchange. The process can be further divided into three procedures: 1) PASP Peering Relations Establish process, 2) PASP Capability Enabling Notification Process, 3) PASP Capability Disabling Notification Process. The Capability Negotiation Process is realized by the exchange of PASP Capability Negotiation Message, which is defined in Section 4.

Although PASP is connectionless, a successful PASP Peering Relations Establish Process is required to be successfully performed before any other PASP process. This process can be initiated by either the local or remote PASP speaker through sending out a PASP Capability Negotiation Message. The Negotiation Message may or may not require an ACK Message, as indicated in the Negotiation Message. A successful Peering is established if both PASP speakers have correctly received the other speaker's Capability Negotiation Message. After a successful negotiation, two PASP speakers can exchange any PASP Message on-demand. The PASP Capability Enabling Notification Process is used to inform the PASP peer its newly supported capability, which can be initiated by the PASP speaker at any moment after a PASP Peering is established with the respective PASP Peer. The PASP Capability Disabling Notification Process is used to inform the PASP peer its newly unsupported capability, which can be initiated by the PASP speaker at any moment after a PASP Peering is established with the respective PASP Peer.

#### 3.4.2. PASP Request and Reply Process

The purpose of the PASP Request and Reply Process is to acquire information needed by a PASP speaker from other PASP speakers for a specific PASP Event. The Request and Reply Messages can be customized for different events. The process is triggered by the instantiation of a PASP Event, and starts with sending a Request Message to a target PASP peer. The target PASP peer is selected by the PASP agent regarding the current PASP Event, which is out of the



scope of this document. The remote PASP speaker, after receiving the Request Message, sends out a Reply Message to the request sender. ACK is required or not as indicated in the Message Flag.

One Request Message received at the local PASP speaker from a PASP peer may further results in a new Request Message generation regarding a third PASP speaker, if the local PASP speaker does not have the right Reply to this PASP peer. This local PASP speaker does not send Reply Message to the requesting PASP peer until it receives a new Reply Message from this third PASP speaker. So the whole process in order to avoid Request/Reply loops, a Residua Hop value is used to limit the Request/Reply rounds.

### 3.4.3. PASP Notification Process

The Notification Process is used by a PASP speaker voluntarily to notify other PASP speakers of certain information regarding a PASP Event. The process is triggered by the instantiation of a PASP Event, and starts with sending a Notification Message to one or multiple target PASP peer(s). The target PASP peer(s) is/are selected by the PASP agent regarding the current PASP Event, which is out of the scope of this document. The Notification Message may or may not require an ACK Message, as indicated in the Notification Message.

#### 4. PASP Message Format

#### 4.1. Common Header

The common header is encapsulated in all PASP messages. It is defined as follows.

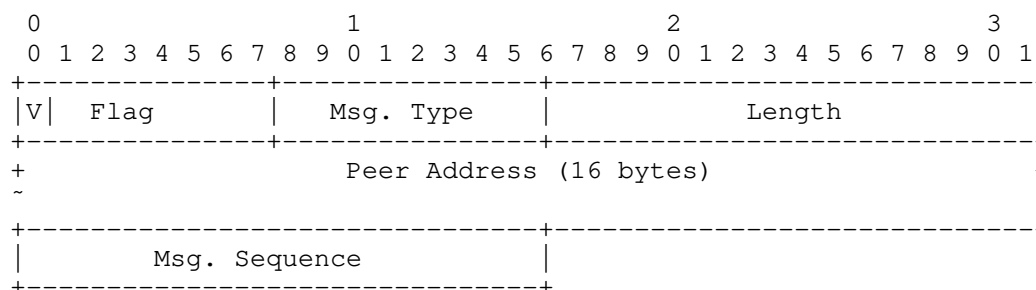


Figure 2. PASP Common Header

- o Flag (1 byte): The V flag indicates that the source IP address is an IPv6 address. For IPv4 address, this is set to 0.

- o Message Type (1 byte): This indicates the PASP message type. The following types are defined, and listed as follows.
  - \* Type = TBD1: Capability Negotiation Message. It is used for two devices to inform each other of the capabilities they support and no longer support.
  - \* Type = TBD2: Request Message.
  - \* Type = TBD3: Reply Message.
  - \* Type = TBD4: Notification Message.
  - \* Type = TBD5: ACK Message. It is used to confirm to the local device that the remote device has received a previous sent PASP message, which can be either a Negotiation Message, a Request Message, a Reply Message or an Notification Message.
- o Length (2 bytes): Length of the message in bytes, including the Common Header and the following Message.
- o Source IP Address (16 bytes): It indicates the IP address who initiates the PASP message. It is 4 bytes long if an IPv4 address is carried in this field (with the 12 most significant bytes zero-filled) and 16 bytes long if an IPv6 address is carried in this field.
- o Message Sequence (2 bytes): It indicates the sequence number of each PASP message.

#### 4.1.1. Capability Negotiation Message

The Negotiation Message is used in the PASP Capability Negotiation Process. It is defined as follows.

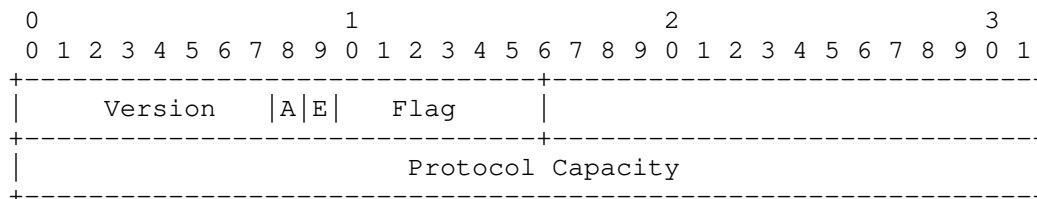


Figure 3. PASP Negotiation Message

- o Version (1 byte): It indicates the PASP version. The current version is 0.

- o Flags (1 bytes): Two flag bits are currently defined.
  - \* The A bit is used to indicate if an ACK Message from the remote PASP speaker is required for each Negotiation Message sent. If an ACK is required, then the A bit SHOULD be set to "1", and "0" otherwise.
  - \* The E bit is used to indicate the enabling/disabling of the capabilities that carried in the Protocol Capability field. If the local device wants to inform the remote device of enabling one or more capabilities, the E bit SHOULD be set to "1". If the local device wants to inform the remote device of disabling one or more capabilities, the E bit SHOULD be set to "0".
- o Protocol Capability (4 bytes): It is 4-byte bitmap that indicates the capability of information exchange regarding various protocols. Each bit represents one protocol. The following protocol capability is defined (from the rightmost bit).
  - \* Bit 0: ISIS
  - \* Bit 1: OSPF
  - \* Bit 2: BGP
  - \* Bit 3: LDP

#### 4.2. Request Message

The Request Message is used for the local device to request specific data regarding one specific protocol or application from the remote device. It MUST be sent after a successful Capability Negotiation Process (described in Section 5.1), and the requested protocol/application MUST be supported by both the local and remote devices, as indicated in the Negotiation Messages exchanged between the local and remote devices. It is defined as follows.

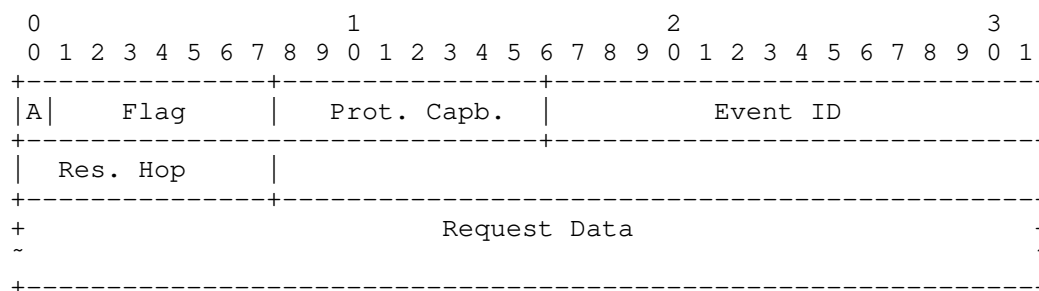


Figure 4. PASP Request Message

- o Flags (1 byte): It is currently reserved. The A bit is used to indicate if an ACK Message from the remote PASP speaker is required for each Request Message sent. If an ACK is required, then the A bit SHOULD be set to "1", and "0" otherwise.
- o Capability (1 byte): It represents the bit index of the protocol, which the Request Message is requesting data for.
- o Event ID (2 bytes): It indicates the event number that this Request message is regarding.
- o Residua hop (1 byte): it indicates the residua Request hops of the current PASP Event. It is reduced by 1 at each PASP speaker when generating a further PASP Request to a third PASP speaker.
- o Request Data (Variable): Specifies information of the data that the local device is requesting. The specific format remains to be determined per each protocol, as well as each use case.

#### 4.3. Reply Message

The Reply Message is used to carry the information that the local device requests from the remote device through the Request Message. It is defined as follows.

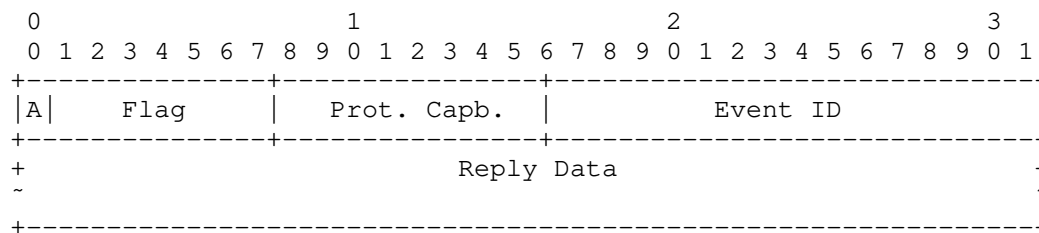


Figure 5. PASP Reply Message

- o Flags (1 byte): It is currently reserved. The A bit is used to indicate if an ACK Message from the remote PASP speaker is required for each Reply Message sent. If an ACK is required, then the A bit SHOULD be set to "1", and "0" otherwise.
- o Capability (1 byte): It represents the bit index of the protocol, which the Reply Message is replying data for.
- o Event ID (2 bytes): It indicates the event number that this Reply message is regarding.
- o Reply Data (Variable): Specifies information of the data that the local device is replying. The specific format remains to be determined per each protocol, as well as each use case.

#### 4.4. Notification Message

The Notification Message is used to carry the information that the local device sends to the remote device.

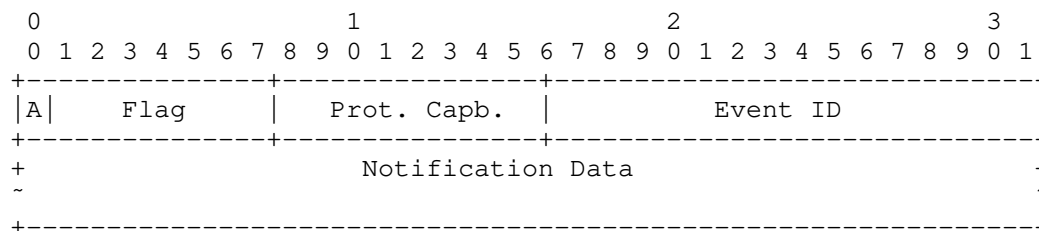


Figure 6. PASP Notification Message

- o Flags (1 byte): It is currently reserved. The A bit is used to indicate if an ACK Message from the remote PASP speaker is required for each Notification Message sent. If an ACK is required, then the A bit SHOULD be set to "1", and "0" otherwise.
- o Capability (1 byte): It represents the bit index of the protocol, which the Notification Message is notifying for.
- o Event ID (2 bytes): It indicates the event number that this Notification Message is regarding.
- o Notification Data (Variable): Specifies information of the data that the local device is notifying. The specific format remains to be determined per each protocol, as well as each use case.
- o

#### 4.5. ACK Message

The ACK Message is used to confirm that the remote device has received a PASP Message with the A bit set to "1". The ACK Message includes only the PASP Common Header. The Msg. Sequence MUST be set to the sequence number carried in the received PASP message, which requires this ACK.

#### 5. PASP Operations

The PASP operations include the following 3 major processes, the Capability Negotiation Process, the Data Request and Reply Process, and the Data Notification Process.

##### 5.1. Capability Negotiation Process

###### 5.1.1. PASP Peering Relation Establish Process

A successful PASP Peering relation MUST be Established between two PASP speakers before any other PASP process.

As the first step, a Capability Negotiation Message can be initiated at any time by a PASP speaker, as long as the target PASP peer is IP reachable. It usually companies the establishment of neighboring/peering relation between two routing devices. The "A" bit in the Negotiation Message MUST be set as 1 during the PASP Peering Establish Process, meaning ACK required. The "E" in the Negotiation Message MUST be set to 1 during this process, meaning the capabilities indicated in the Protocol Capability field are enabled by default. The Protocol Capability field SHOULD indicate all the protocol capabilities that are supported by the local PASP Agent and currently enabled. After the first Negotiation Message is sent, the local device SHOULD wait for the ACK Message from the remote device for a certain time period before taking further actions, and if no ACK Message is received within this time frame, the local device SHOULD resend the Negotiation Message to the remote device. The waiting period can be configured locally. This send and wait process CAN be repeated for at most 3 times before receiving a ACK Message from the remote device. If after 3 times of resending the Negotiation Message, still no ACK received, then this peering establishment is treated as unsuccessful.

The next step for the local PASP speaker is to wait for the Negotiation Message from the remote PASP speaker. If no Negotiation Message is received from the remote PASP speaker within a time frame after its own Negotiation Message is sent, the local PASP speaker CAN resend the Negotiation Message. This time frame is also configured locally. This send and wait process CAN be repeated for

at most 3 times before receiving a Negotiation Message from the remote PASP speaker. If after 3 times of resending the Negotiation Message, still no Negotiation Message received, then this negotiation is treated as unsuccessful. If a Negotiation Message is received and parsed correctly, an ACK MUST be sent to the remote PASP speaker.

Once an ACK Message and a Negotiation Message are received from the remote PASP speaker and correctly parsed, a PASP Peering relation is considered as successfully established. The local PASP speaker maintains locally the protocol capabilities of the remote PASP speaker, and uses them during other PASP processes.

#### 5.1.2. PASP Capability Enabling Notification Process

Once the PASP Peering relation is set up between two PASP speakers, they become PASP peers. Thereafter, any PASP speaker supports a new protocol capability, it SHOULD call the Capability Enabling Notification Process to inform all its PASP peers.

When the local PASP speaker initiates a PASP Capability Enabling Notification Process: The "A" bit in the Negotiation Message MUST be set as 1 during the PASP Capability Enabling Notification Process, meaning ACK required. The "E" in the Negotiation Message MUST be set to 1 during this process, meaning the capabilities indicated in the Protocol Capability field are enabled. The Protocol Capability field SHOULD indicate all the protocol capabilities that are supported by the local PASP Agent and currently enabled. After the Negotiation Message is sent, the local PASP speaker SHOULD wait for the ACK Message from the PASP peer for a certain time period before taking further actions, and if no ACK Message is received within this time frame, the local device SHOULD resend the Negotiation Message to the remote device. The waiting period can be configured locally. This send and wait process CAN be repeated for at most 3 times before receiving a ACK Message from the remote device. If after 3 times of resending the Negotiation Message, still no ACK received, then this Capability Enabling Notification Process is treated as unsuccessful. This process MAY be initiated at another time thereafter. If a ACK is received, the Capability Enabling Notification Process is considered successful.

When a PASP peer initiates a PASP Capability Enabling Notification Process: The local PASP speaker, after receiving the PASP Negotiation Message and correctly parsing it, sends out an ACK. This Capability Enabling Notification Process is considered successful. The local PASP speaker updates the capability status maintained accordingly.

### 5.1.3. PASP Capability Disabling Notification Process

Whenever a PASP speaker disables a PASP capability, it SHOULD initiate a PASP Capability Disabling Notification Process to inform all its PASP peers.

When the local PASP speaker initiates a PASP Capability Disabling Notification Process: The "A" bit in the Negotiation Message MUST be set as 1 during the PASP Capability Disabling Notification Process, meaning ACK required. The "E" in the Negotiation Message MUST be set to 0 during this process, meaning the capabilities indicated in the Protocol Capability field are disabled. The Protocol Capability field SHOULD indicate all the protocol capability that is disabled. After the Negotiation Message is sent, the local PASP speaker SHOULD wait for the ACK Message from the PASP peer for a certain time period before taking further actions, and if no ACK Message is received within this time frame, the local device SHOULD resend the Negotiation Message to the remote device. The waiting period can be configured locally. This send and wait process CAN be repeated for at most 3 times before receiving a ACK Message from the remote device. If after 3 times of resending the Negotiation Message, still no ACK received, then this Capability Disabling Notification Process is treated as unsuccessful. This process MAY be initiated at another time thereafter.

When a PASP peer initiates a PASP Capability Disabling Notification Process: The local PASP speaker, after receiving the PASP Negotiation Message and correctly parsing it, sends out an ACK. This Capability Disabling Notification Process is considered successful. The local PASP speaker updates the capability status maintained accordingly.

### 5.2. PASP Request and Reply Process

When a local PASP Event triggers a PASP Request and Reply Process, the local PASP speaker initiates a Request Message, and send to a target PASP peer as indicated by PASP Agent per this PASP Event. This local PASP speaker is called the Request and Reply Process Starter. It sets the Residua Hop as the maximum number of Request/Reply rounds (e.g., 10) it will wait in order to receive the final Reply. The Event ID and the Request are set by the local PASP Agent. The A bit of the Request Message MUST be set to "1" (i.e., ACK is required). The local device waits for the ACK Message from the remote device for a certain time period before taking further actions, and if no ACK Message is received within this time frame, the local device SHOULD resend the Request Message to the remote device. The waiting period can be configured locally. This send and wait process CAN be repeated for at most 3 times before receiving a ACK Message from the remote device. If after 3 times of resending



the Request Message, still no ACK received, then this Request and Reply Process is treated as unsuccessful. If ACK received, the local device waits for the Reply Message. If no Reply Message is received from the remote device within a time frame, the local device can resend the Request Message. This send and wait process CAN be repeated for at most 3 times before receiving a Reply Message from the remote device. If after 3 times of resending the Request Message, still no Reply Message received, then this Request and Reply Process is treated as unsuccessful. The waiting period can be configured locally, and SHOULD take into consideration of the Residua Hop value. If the Request and Reply Process Starter receives the Reply Message within the time frame, and the Event ID is matched to the local PASP Event, the PASP Request and Reply Process is considered as successful.

When a local PASP speaker receives a Request Message from its PASP peer (i.e., it is not the Pequest and Reply Process Starter), it sends back an ACK Message. With the received Request Message, a new PASP event is instantiated at the local PASP Agent. The PASP event triggers the troubleshooting analysis of the received Request Message, and then generate the Reply Message if the Reply condition is met, or generate a new Request Message when the Reply condition is not met. The Reply condition and the troubleshooting analysis of the PASP Agent is out of the scope of this document.

If the Reply condition is met, the local PASP speaker is called the Request and Reply Process Terminator. It generates the Reply Message and send the message back to the requesting PASP peer. The Event ID is set to be the same as the Event ID of the received Request Message. The Reply Data is set by the local PASP Agent per this generated event. The A bit of the Reply Message MUST be set to "1" (i.e., ACK is required). The local device waits for the ACK Message from the remote device for a certain time period before taking further actions, and if no ACK Message is received within this time frame, the local device SHOULD resend the Reply Message to the remote device. The waiting period can be configured locally. This send and wait process CAN be repeated for at most 3 times before receiving a ACK Message from the remote device. If after 3 times of resending the Request Message, still no ACK received, then this Request and Reply Process is treated as unsuccessful.

If the Reply condition is not met, the local PASP speaker is called the Request and Reply Process mid-handler. It generates a new Request Message and send the message to a third PASP speaker per indicated by the local PASP Agent per this generated event. In the new generated Request Message, the Residua Hop value by MUST be reduced by 1. The A bit of the Request Message MUST be set to "1" (i.e., ACK is required). The local device waits for the ACK Message

from the remote device for a certain time period before taking further actions, and if no ACK Message is received within this time frame, the local device SHOULD resend the Request Message to the remote device. The waiting period can be configured locally. This send and wait process CAN be repeated for at most 3 times before receiving a ACK Message from the remote device. If after 3 times of resending the Request Message, still no ACK received, then this Request and Reply Process is treated as unsuccessful. If ACK received, the local device waits for the Reply Message. If no Reply Message is received from the remote device within a time frame, the local device can resend the Request Message. This send and wait process CAN be repeated for at most 3 times before receiving a Reply Message from the remote device. If after 3 times of resending the Request Message, still no Reply Message received, then this Request and Reply Process is treated as unsuccessful. The waiting period can be configured locally, and SHOULD take into consideration of the Residua Hop value. If the local device receives the Reply Message within the time frame, it generates a new Reply Message and sends back to it requesting PASP peer. The Event ID of the new Reply Message is set to be the same as the Event ID of the received Request Message.

### 5.3. PASP Notification Process

When a local PASP Event triggers a PASP Notification Process, the local PASP speaker initiates a Notification Message. The target PASP peer(s) is/are selected by the PASP agent regarding the current PASP Event, which is out of the scope of this document. The Notification Message may or may not require an ACK Message, as indicated in the Notification Message. If the A bit is set to 1 (meaning ACK required), the local device waits for the ACK Message from the remote device for a certain time period before taking further actions, and if no ACK Message is received within this time frame, the local device SHOULD resend the Notification Message to the remote device. The waiting period can be configured locally. This send and wait process CAN be repeated for at most 3 times before receiving a ACK Message from the remote device. If after 3 times of resending the Request Message, still no ACK received, then this Request and Reply Process is treated as unsuccessful. The waiting period can be configured locally. If ACK is received within the time frame, the Notification Process is considered to be successful. If the A bit is set to 0 (meaning no ACK required), after sending the Notification Message, the Notification Process is considered successful.

## 6. PASP Error Handling

When any PASP process is unsuccessful, information is recorded or not by local PASP Agent. No further action is taken.

## 7. Discussion

In addition to the preceding message definition and process description, the security and reliability requirements of the PASP need to be considered. There are two possible options to implement PASP.

- Option 1: PASP is developed independently as a new protocol.

- Option 2: PASP reuses the existing protocol Generic Autonomic Signaling Protocol (GRASP) [I-D.ietf-anima-grasp] .

Option1:

1. Definition of the Message Format and Interaction Process: It can be defined independently in the PASP.

2. Reliability: The transmission mode of PASP is based on UDP mainly considering that the collected information is the auxiliary information to help locate the protocol fault, and the information loss has no impact on the service. In addition, if TCP mode is adopted, the resource consumption of the device may be large, especially when there are a large number of neighbors. If it is considered that PASP must ensure reliability, it can be done in the application layer, such as adding the sequence number to the message.

3. Security: MD5 authentication can be introduced for PASP security.

Option2:

ANIMA GRASP is a signaling protocol used for dynamic peer discovery, status synchronization, and parameter negotiation between AS nodes or AS service agents. GRASP specifies that unicast packets must be transmitted based on TCP, and multicast packets (Discovery and Flood) must be transmitted based on UDP.

1. Message format and interaction process: PASP can reuse the defined messages and procedures of the GRASP. Messages defined in the PASP include Capability Negotiation Message, Request Message, Reply Message, and Negotiation Message. These message types are also defined in GRASP.

2. Reliability: TCP mode of GRASP can be used to ensure reliability for PASP. But there may be challenge for the equipment resources.

3. Security: Autonomic Control Plane (ACP)  
[I-D.ietf-anima-autonomic-control-plane] can be reused.

## 8. Security Considerations

TBD

## 9. IANA

TBD

## 10. Contributors

We thank Jiaqing Zhang (Huawei), Tao Du (Huawei) and Lei Li (Huawei) for their contributions.

## 11. Acknowledgments

## 12. References

[I-D.brockners-inband-oam-requirements]

Brockners, F., Bhandari, S., Dara, S., Pignataro, C., Gredler, H., Leddy, J., Youell, S., Mozes, D., Mizrahi, T., Lapukhov, P., and r. remy@barefootnetworks.com, "Requirements for In-situ OAM", draft-brockners-inband-oam-requirements-03 (work in progress), March 2017.

[I-D.ietf-anima-autonomic-control-plane]

Eckert, T., Behringer, M., and S. Bjarnason, "An Autonomic Control Plane (ACP)", draft-ietf-anima-autonomic-control-plane-30 (work in progress), October 2020.

[I-D.ietf-anima-grasp]

Bormann, C., Carpenter, B., Ed., and B. Liu, Ed., "A Generic Autonomic Signaling Protocol (GRASP)", draft-ietf-anima-grasp-15 (work in progress), July 2017.

[I-D.ietf-netconf-yang-push]

Clemm, A. and E. Voit, "Subscription to YANG Datastores", draft-ietf-netconf-yang-push-25 (work in progress), May 2019.

- [I-D.song-ntf]  
Song, H., Zhou, T., Li, Z., Fioccola, G., Li, Z.,  
Martinez-Julia, P., Ciavaglia, L., and A. Wang, "Toward a  
Network Telemetry Framework", draft-song-ntf-02 (work in  
progress), July 2018.
- [RFC1157] Case, J., Fedor, M., Schoffstall, M., and J. Davin,  
"Simple Network Management Protocol (SNMP)", RFC 1157,  
DOI 10.17487/RFC1157, May 1990,  
<<https://www.rfc-editor.org/info/rfc1157>>.
- [RFC1191] Mogul, J. and S. Deering, "Path MTU discovery", RFC 1191,  
DOI 10.17487/RFC1191, November 1990,  
<<https://www.rfc-editor.org/info/rfc1191>>.
- [RFC1195] Callon, R., "Use of OSI IS-IS for routing in TCP/IP and  
dual environments", RFC 1195, DOI 10.17487/RFC1195,  
December 1990, <<https://www.rfc-editor.org/info/rfc1195>>.
- [RFC1213] McCloghrie, K. and M. Rose, "Management Information Base  
for Network Management of TCP/IP-based internets: MIB-II",  
STD 17, RFC 1213, DOI 10.17487/RFC1213, March 1991,  
<<https://www.rfc-editor.org/info/rfc1213>>.
- [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>>.
- [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>>.
- [RFC3988] Black, B. and K. Kompella, "Maximum Transmission Unit  
Signalling Extensions for the Label Distribution  
Protocol", RFC 3988, DOI 10.17487/RFC3988, January 2005,  
<<https://www.rfc-editor.org/info/rfc3988>>.
- [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>>.

- [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>>.
- [RFC7854] Scudder, J., Ed., Fernando, R., and S. Stuart, "BGP Monitoring Protocol (BMP)", RFC 7854, DOI 10.17487/RFC7854, June 2016, <<https://www.rfc-editor.org/info/rfc7854>>.

## Authors' Addresses

Zhenbin Li  
Huawei  
156 Beiqing Rd  
Beijing  
China

Email: [lizhenbin@huawei.com](mailto:lizhenbin@huawei.com)

Shuanglong Chen  
Huawei  
156 Beiqing Road  
Beijing, 100095  
China

Email: [chenshuanglong@huawei.com](mailto:chenshuanglong@huawei.com)

Yingzhen Qu  
Futurewei

Email: [yingzhen.qu@futurewei.com](mailto:yingzhen.qu@futurewei.com)

Yunan Gu  
Huawei  
156 Beiqing Rd  
Beijing  
China

Email: [guyunan@huawei.com](mailto:guyunan@huawei.com)

Network Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: January 9, 2020

B. Liu  
Huawei Technologies  
July 8, 2019

Instant Congestion Assessment Network (iCAN) for Data Plane Traffic  
Engineering  
draft-liu-ican-00

Abstract

iCAN (instant Congestion Assessment Network) is a set of mechanisms running directly on network nodes:

- o To adjust the flows paths based on real-time measurement of the candidate paths.
- o The measurement is to reflect the congestion situation of each path, so that the ingress nodes could decide which flows need to be switched from a path to another.

This is something that current SDN and TE technologies can hardly achieve:

- o SDN Controller is slow and far from the data plane, it is neither able to assess the real-time congestion situation of each path, nor able to assure the data plane always go as expected (especially in SRv6 scenarios). However, iCAN can work with SDN perfectly: controller planning multi-path transmission, and iCAN does the flow optimization automatically.
- o Traditional TE is not able to adjust the flow paths in real-time.

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 January 9, 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. iCAN Architecture and Key Technical Requirements . . . . .	3
2.1. Architecture . . . . .	3
2.2. Key technical requirements . . . . .	5
2.2.1. Path quality assessment . . . . .	5
2.2.2. Recognition and statistic of flows in devices . . . . .	5
2.2.3. Flow switching between paths . . . . .	5
3. Use Cases of iCAN . . . . .	6
3.1. Network load balancing . . . . .	6
3.2. SLA assurance . . . . .	6
3.3. Fine-Granularity reliability . . . . .	6
4. Implementation Scenarios . . . . .	6
4.1. iCAN with SRv6 . . . . .	6
4.2. iCAN with VxLAN . . . . .	7
4.3. iCAN with MPLS/MPLS-TE . . . . .	7
5. Standardization Requirements . . . . .	7
6. Security Considerations . . . . .	7
7. IANA Considerations . . . . .	7
8. Acknowledgements . . . . .	7
9. References . . . . .	8
9.1. Normative References . . . . .	8
9.2. Informative References . . . . .	8
Author's Address . . . . .	8

#### 1. Introduction

Traditional IP routing is shortest path based on static metrics, which can fulfil basic requirement of connectivity. MPLS-TE brings the capability of utilizing non-shortest paths, thus traffic dispatch



is doable; however, MPLS-TE is only a complementary mechanism because of the scalability issue. Segment routing provides even more flexibility that paths could be easily programmed; and along with the controller, it could be scaled.

However, the above mentioned mechanism all run in the control plane, which implies that they are not able to sense the data plane situation in real-time, thus they are mostly for relative static planning/controlling (minutes, hours or even day-level) of network traffic and not able to adapt to the microscopic traffic change in real-time (e.g. milli-second level). So, in real bearer networks (metro, backbones etc.), it is always underload so that the redundant resources could tolerate the traffic burst, results in a significant waste of network resources.

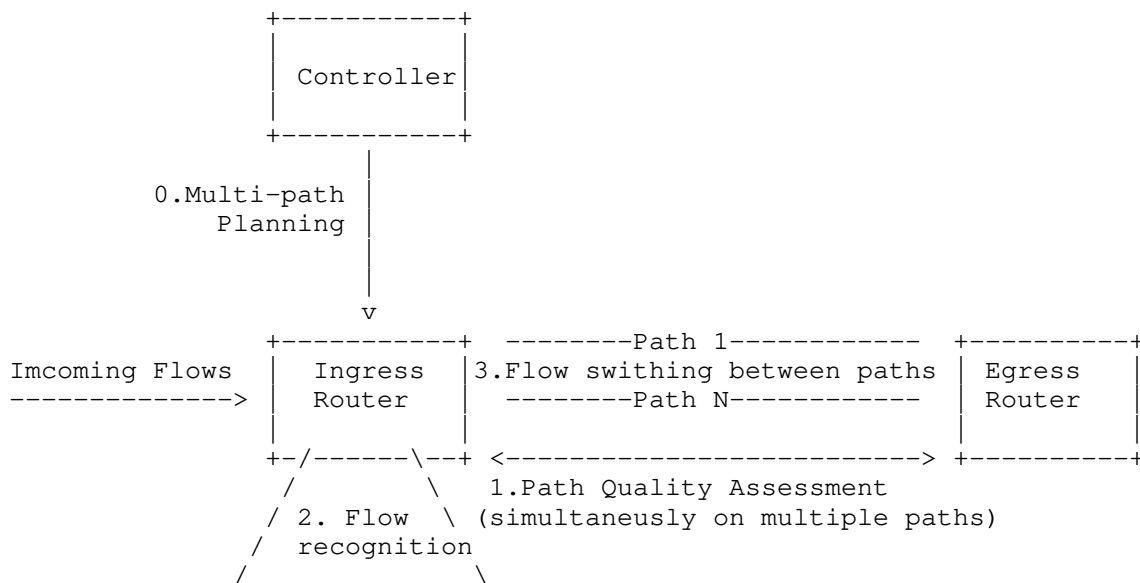
This draft proposes the iCAN (Instant Congestion Assessment Network) architecture to achieve autonomous adapt to traffic changes in real-time in terms of switching flows between multiple forwarding paths. iCAN includes following things:

- o A mechanism between ingress and egress nodes to assess the path congestion situation in RTT level speed, to recognize which paths are underload and which are heavy loaded.
- o Recognizing big flows and small flows in the device, in real time
- o Ingress node dispatches flows to multiple paths, to make load balance, or to guarantee SLA for specific flows

This draft also discusses use cases and implementation scenarios of iCAN.

## 2. iCAN Architecture and Key Technical Requirements

### 2.1. Architecture



As above figure shows, there are 3 entities:

#### 1. Controller

- Responsible for planning multiple paths for a set of flows that could be aggregated to a pair of Ingress/Egress routers.
- After delivering the planned paths to the ingress router, the controller would need nothing to do.

#### 2. Ingress router:

- Serves as a local "controller" for the iCAN system.
- Responsible for triggering the path congestion assessment, which is coordinated with the egress router through a measurement protocol.
- After getting the assessment results, the ingress router would calculate which flows need to be switched to a different path, in order to make the paths load balanced or to assure the transport quality of a certain of important flows.
- In order to do the path switching calculation, the ingress router needs to recognize the TopN flow passing by it, since switching the big flows would make the most effort.

### 3. Egress router:

- Only needs to coordinate with the ingress router to do the path assessment.

## 2.2. Key technical requirements

### 2.2.1. Path quality assessment

- o Req-1: the assessment MUST reflex the congestion status of the paths. (Note: a candidate congestion metric is proposed as: [I-D.dang-ippm-congestion].)
- o Req-2: the assessment SHOULD be done within a RTT timeslot. Since iCAN is to adapt the traffic change in real-time, the assessment needs to be done very fast.
- o Req-3: the assessment MUST be done for multiple paths between the same ingress/egress routes simultaneously. (Note: a candidate congestion metric is proposed as: [I-D.dang-ippm-multiple-path-measurement].)

### 2.2.2. Recognition and statistic of flows in devices

- o Req-1: the device SHOULD be able to recognize TopN big flows within a timeslot.
- o Req-2: the device MAY need to statistic all flows' amount within a timeslot.

### 2.2.3. Flow switching between paths

- o Req-1: the device SHOULD be able to recognize flow let. The flow switching is done from the next flow let.
- o Req-2: the device MAY need to actively generate gap to artificially create flow let. If the flow needs to be switched immediately, then the device would need to make the gap, to avoid out-of-order packets arriving to the destination through multiple paths.
- o Req-3: the device SHOULD avoid oscillation of frequently switching flows from one to another.
- o Req-4: multiple ingress devices SHOULD be able to coordinate so that they won't switch flows to the shared path at the same time, to avoid potential congestion in the shared path.

### 3. Use Cases of iCAN

#### 3.1. Network load balancing

Background problem: traffic is not balanced in current metro network.

While some links are heavily loaded, others might be still lightly loaded: unbalance could lows down the service quality (e.g. SLA could not be guaranteed in the heavily loaded links/paths); unbalance could lows down the network utilization ratio (normally with 30%, e.g. a 100G physical capacity network can only bear at most 30G traffic, a huge waste of network infrastructure).

iCAN could be used for load balance among the multiple paths between a pair of ingress/egress nodes. Once the network is balanced, the real throughput of the network could be elevated significantly.

#### 3.2. SLA assurance

Since iCAN could switch flow in real-time, it can guarantee a set of important flows. Once the path which carries the important flows is to be congested, the other flows could be switched to alternative paths, and the important flows would stablely running in the original path.

(More content TBD)

#### 3.3. Fine-Granularity reliability

Traditional reliability protocols such as BFD, can only assess the link on or off. With the path congestion assessment ability, iCAN could also asses the quality degradation.

(More content TBD)

### 4. Implementation Scenarios

#### 4.1. iCAN with SRv6

##### - SR Multiple Explicit Paths

For example, there are 3 paths between the ingress and egress nodes, and the multi-path is defined as a SR-List containing LSP1/2/3.

The probe message detects the congestion status of the three SR-list paths. The edge device adjusts the load balancing between the three paths according to the congestion status of the three

SR-lists, and switch the flows from the path with a high congestion to the path with a low congestion.

- SR Multiple Explicit+Loose Paths

In loose path scenario, there needs to be an additional approach to probe the specific paths of a SR tunnel. After that, operations on the probed paths are the same as explicit path scenario.

#### 4.2. iCAN with VxLAN

TBD.

#### 4.3. iCAN with MPLS/MPLS-TE

TBD.

### 5. Standardization Requirements

1. Multi-path Planning (North Interface between Controller and devices)
2. Path Congestion Assesment (Horizontal Interface between devices), mostly regarding to Req-1&2&3 described in Section 2.2.1 .
3. Flow Switching Negotiation (Horizontal Interface between devices), mostly regarding to Req-3&4 described in Section 2.2.3 .

(More content TBD.)

### 6. Security Considerations

TBD.

### 7. IANA Considerations

TBD.

### 8. Acknowledgements

Very valuable comments were from Shunsuke Homma, Mikael Abrahamsson and Bruno Decraene.

A commercial router hardware based prototype had been implemented to prove the machinisms discussed in the document are workable.

## 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>>.
- [RFC2629] Rose, M., "Writing I-Ds and RFCs using XML", RFC 2629, DOI 10.17487/RFC2629, June 1999, <<https://www.rfc-editor.org/info/rfc2629>>.

### 9.2. Informative References

- [I-D.dang-ippm-congestion]  
Dang, J. and J. Wang, "A One-Path Congestion Metric for IPPM", draft-dang-ippm-congestion-01 (work in progress), March 2019.
- [I-D.dang-ippm-multiple-path-measurement]  
Dang, J. and J. Wang, "A Multi-Path Concurrent Measurement Protocol for IPPM", draft-dang-ippm-multiple-path-measurement-01 (work in progress), March 2019.

### Author's Address

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

Email: [leo.liubing@huawei.com](mailto:leo.liubing@huawei.com)

Individual  
Internet-Draft  
Intended status: Informational  
Expires: January 3, 2020

R. Rokui  
Nokia  
S. Homma  
NTT  
D. Lopez  
Telefonica I+D  
X. de Foy  
InterDigital Inc.  
L. Contreras-Murillo  
J. Ordonez-Lucena  
Telefonica I+D  
P. Martinez-Julia  
NICT  
M. Boucadair  
Orange  
P. Eardley  
BT  
K. Makhijani  
Futurewei Networks  
H. Flinck  
Nokia  
July 2, 2019

5G Transport Slice Connectivity Interface  
draft-rokui-5g-transport-slice-00

Abstract

The 5G Network slicing is an approach to provide separate independent E2E logical network from user equipment (UE) to applications where each network slice has different SLA requirements. Each E2E network slice consists of multitude of RAN-slice, Core-slice and Transport-slices, each with its own controller. To provide automation, assurance and optimization of the network slices, an E2E network slice controller is needed which interacts with controller in RAN, Core and Transport slices. The interfaces between the E2E network slice controller and RAN and Core controllers are defined in various 3GPP technical specifications. However, 3GPP has not defined the same interface for transport slices.

The aim of this document is to provide the clarification of this interface and to provide the information model of this interface for automation, monitoring and optimization of the transport slices.

## 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 January 3, 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. Definition of Terms . . . . .	3
2. High level architecture of 5G end-to-end network slicing . .	5
3. Logical flow cross layers for automation of an end-to-end network slices . . . . .	8
4. Definition of Transport Slice . . . . .	11
4.1. Transport Slices in Distributed RAN . . . . .	11
4.2. Transport Slices in Centralized RAN (C-RAN) . . . . .	12
4.3. Transport Slices in cloud RAN . . . . .	13
4.4. Transport Slice as a set of networks . . . . .	14
5. Transport Slice Connectivity Interface (TSCi) . . . . .	16
5.1. Relationship between TSCi and various IETF data models .	17
5.2. Realization (aka Implementation) of transport slices . .	18
6. Transport slice connectivity interface (TSCi) information	



model . . . . .	20
6.1. transport-slice-info . . . . .	22
6.2. network-slice-info . . . . .	22
6.3. transport-slice-networks . . . . .	22
6.4. node . . . . .	22
6.5. connection-link . . . . .	23
6.6. transport-slice-policy . . . . .	23
6.6.1. transport-slice-sla-policy . . . . .	23
6.6.2. transport-slice-selection-policy . . . . .	23
6.6.3. transport-slice-assurance-policy . . . . .	23
6.7. IETF models . . . . .	24
6.7.1. ACTN model . . . . .	24
6.7.2. i2rs model . . . . .	24
7. IANA Considerations . . . . .	24
8. Security Considerations . . . . .	24
9. Informative References . . . . .	24
Authors' Addresses . . . . .	26

## 1. Introduction

Network Slicing is a mechanism which a network operator can use to allocate dedicated infrastructures and services to a customer (aka tenant) from shared operator's network. In particular a 5G network slice is inherently an E2E concept and is composed of multiple logical independent networks are created in a common operator's network from an user equipment to applications. In specific, the network slicing receives attention due to factors such as diversity of services and devices in 5G each with its own SLA requirements. Each E2E network slice consists of multitude of RAN-slice, Core-slice and Transport-slices each with its own controller.

To provide automation, assurance and optimization of the network slices, an E2E network slice orchestrator is needed which interacts with controller of RAN, Core and Transport slices. The interfaces between the E2E network slice controller and RAN and Core controllers are defined in various 3GPP technical specifications. However, 3GPP has not defined the same interface for transport slices. The aim of this document is to provide the clarification of this interface and to provide the information model of this interface for automation, monitoring and optimization of the transport slices.

### 1.1. Definition of Terms

Please refer to [I-D.homma-slice-provision-models] as well.

**Customer:** Also known as Tenant. Any application, client network, or customer of a network provider, i.e. an NS tenant is a person or group that rents and occupies NSIs from network provider.

**E2E Network Slice:** A E2E network slice is a virtual network provided by a Slice Provider that has the functionality and performance to support a specific industry sector and/or service. This capability will be the subject of a commercial agreement between the Slice Provider and Slice Buyer, and although it may well support dynamic scale-in/out, the Slice capability will normally be long-lasting i.e. only change on commercial timescales, although this may become more dynamic over time.

In specific, E2E 5G network slices are separate independent logical network E2E from user equipment (UE) to applications in a common infrastructure where each logical network has dedicated SLA. It is an E2E concept which spans multiple network domains (e.g. radio, transport and core), and in some cases more than one administrative domain.

**Network Slice Instance (NSI):** Also known as Network slice profile (NS profile), NSI is an E2E instance of a network slice blueprint which is instantiated in service provider's network for specific customer and specific service type. Note that customer and service type can be wildcard.

**Transport Slice:** It is also called Transport Sub-Slice. A set of connections between various network functions (VNF or PNF) with deterministic SLAs. They can be implemented (aka realized) with various technologies (e.g. IP, Optics, FN, Microwave) and various transport (e.g. RSVP, Segment routing, ODU, OCH etc).

**RAN Slice:** It is also called RAN Sub-Slice. The context and personality created on RAN network functions for each E2E network slice.

**Core Slice:** It is also called Core Sub-Slice. The context and personality created on Core network (CN) functions for each e2e network slice.

**S-NSSAI:** Single Network Slice Selection Assistance Information, defined by 3GPP and is the identification of a Network Slice

**Sub-Slice:** The RAN, Transport or Core Slices are also called Sub-Slice or Subnet; i.e. RAN, Transport and Core Sub-slices or Subnets

**Service Level Agreement (SLA):** An agreement between a customer (aka tenant) and network provider that describes the quality with which features and functions are to be delivered. It may include information on target KPIs (such as min guaranteed throughput, max tolerable latency, max tolerable packet loss rate); the types of

service (such as the network service functions or billing) to be executed; the location, nature, and quantities of services (such as the amount and location of compute resources and the accelerators require)

gNB: gNB incorporates two major modules; Centralized Unit (CU) and Distribute Unit (DU)

UE: UE: User Equipment such as vehicle Infotainment, Cell phone, IoT sensor etc

## 2. High level architecture of 5G end-to-end network slicing

To demonstrate the concept of 5G E2E network slicing and the role of various controllers consider a typical 5G network shown in Figure 1 where a mobile network operator Y has two customers (tenants) C1 and Public Safety. The boundaries of administrative domain of the operator includes RAN, Transport, Core and Application domains. Each customer requests to have several logical independent E2E networks from UEs (e.g. vehicle infotainment, mobile phone, IoT meters etc.) to the application, each with distinct SLAs. In 5G, each of these independent networks called an E2E network slice, which consists of RAN, Transport and Core slices (or RAN, Transport and Core Sub-slices).

In Figure 1 there are four E2E network slices, NS1 to NS4, each with its own RAN, Core and Transport slices. To create RAN slice, the RAN network functions such as eNB and gNB should be programmed to have a context for each E2E network slice. This context means that the RAN network functions should allocate certain resources for each E2E network slice such as air interface, various schedulers, policies and profiles to guarantee the SLA requirement for that specific network slice. By the same token, the Core slice means to create the context for each E2E network slice on Core network functions. This means that the RAN and Core network functions are aware of multiple E2E network slices.

When both RAN and Core slices are created, they should be connected together using a set of connections to have an E2E network slice. These set of connections are called Transport Slices, i.e. the transport slices are a group of connections with specific SLAs. The following factors dictate the number of transport slices. The details on transport slices will be discussed in see Section 4:

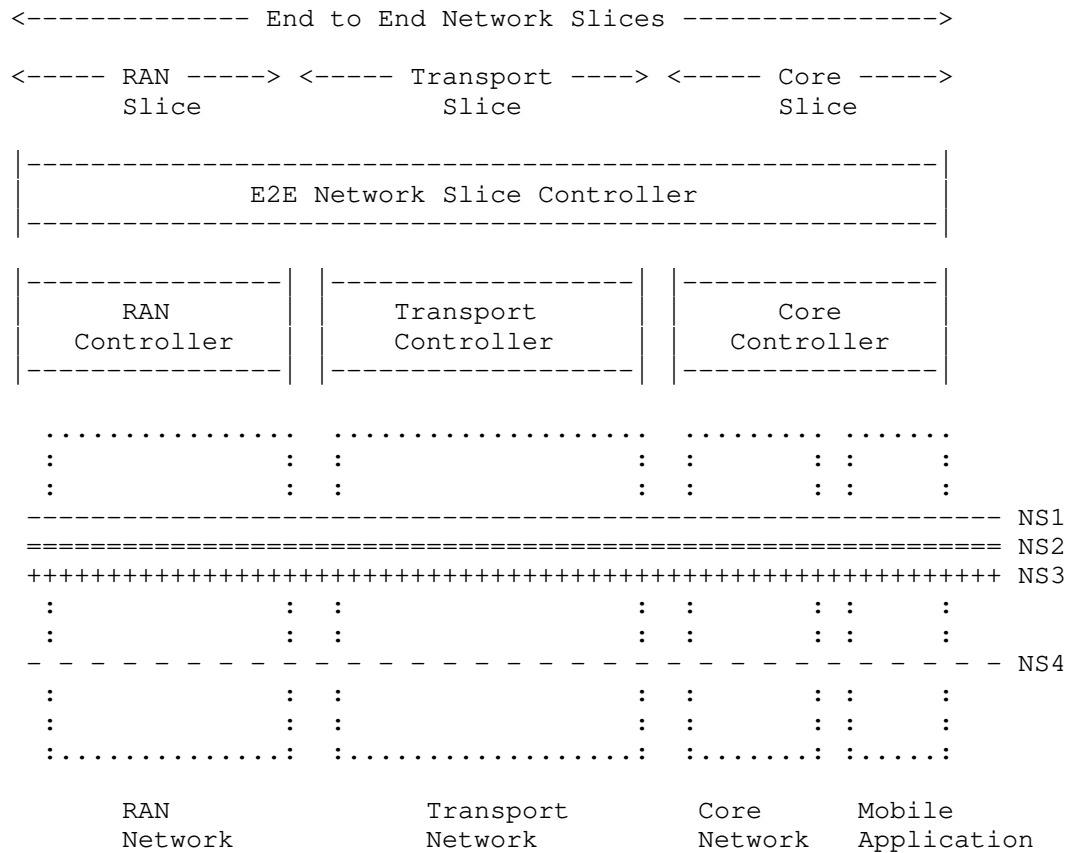
- o The RAN deployment (i.e. distributed RAN, Centralized RAN or Cloud RAN). For example, in Cloud RAN, the RAN network function (i.e. gNB) is decomposed into two network functions (called CU and DU) where one or both will be VNFs and there is a Midhaul network

between them. In this case, there will be a transport slice in RAN to connect DUs to CU

- o The location of the mobile applications. If there is a network between the mobile applications and the 5G Core, another transport slice is needed to connect the 5G Core to Applications.
- o Mobile network policy on how to allow creation of the E2E network slices and if the sharing of transport slices between multiple E2E network slices are desirable and allowed.

At the end when RAN, Core and Transport slices are created, they should be bind or associated together to form a working E2E network slice. Since the nature of an E2E network slice is dynamic and the life cycle of each network slice might be a few hours or few months, various controllers are needed to perform the life cycle of various E2E network slices in their respective domains. As shown in Figure 1, each RAN, Core and Transport slice needs a controller. Also an E2E network slice controller is needed to provide the coordination and control of network slices from an E2E perspective.

In summary, an E2E network slice will involve several domains, each with its own controller: 5G RAN domain, transport domain, and 5G core domain. These need to be coordinated in order to deliver an E2E service. Note that in this context a service is not necessarily an IP/MPLS service but rather customer (aka tenant) facing services such as CCTV service, eMBB service and Infotainment service etc.



Customers (aka Tenants): Public Safety and C1

MNO (Mobile Network Operator): Operator Y

#### Legend:

- NS1: E2E network slice 1 for customer C1,  
service type Infotainment
- ===== NS2: E2E network slice 2 for customer C1,  
service type Autonomous Driving
- +++++ NS3: E2E network slice 3 for customer C1,  
service type HD Map (NS3)
- - - NS4: E2E network slice 4 for customer Public Safety,  
service type Video Surveillance

Figure 1: High level architecture of 5G end-to-end network slicing

### 3. Logical flow cross layers for automation of an end-to-end network slices

Figure 2 provides the logical flow cross layers to achieve the automatic creation of each E2E network slices such as NS1 mentioned in Figure 1. Below are the logical flow among various controllers to achieve this. It is important to note that these steps are logical and in practice some of them can be combined. For example, step 3 can be combined with steps 4 or 5.

1. The customer C1 will request the Operator Y for creation of an E2E network slice for Infotainment service type and SLA of 10 [Mbps]
2. The E2E network slice controller receives this request and using its pre-defined network slice blueprints (aka network slice templates) creates a network slice profile (aka network slice instance) which contains all the network functions in RAN and core which should be part of this E2E network slice. It then goes through decomposition of this profile and triggers various actions. Steps 3 to 7 shows these actions.
3. A request for creation of the RAN slice will be triggered to RAN Controller.
4. If RAN network functions are virtual, the RAN Slice controller triggers the creation of VNFs in RAN (using for example ETSI interface Os-Ma-nfvo)
5. NFVO manages the life cycle of virtual RAN network functions
6. Since both physical and virtual RAN network functions which are part of the E2E network functions are known to RAN controller, it then triggers the creation of RAN slice by programming RAN network functions
7. Similar to previous steps, a request for creation of the Core slice will be triggered to Core Controller.
8. If Core network functions are virtual, the Core Slice controller triggers the creation of VNFs (using for example ETSI interface Os-Ma-nfvo) and NFVO manages the life cycle of virtual Core network functions
9. Since both physical and virtual Core network functions which are part of the E2E network functions are known to Core controller, it then triggers the creation of Core slice by programming Core network functions

10. In this step, various transport slices (i.e. various connections) need to be created between various network functions, e.g. transport slices between RAN and Core slices, transport slices between RAN network functions or transport slices between core and applications
11. The transport slices will be implemented (aka realized) in the network
12. [Optional] If the creation of transport slice involves creation of VNFs (e.g. Firewall, security gateway etc.), triggers the creation of these VNFs (using for example ETSI interface Os-Ma-nfvo)
13. The E2E network slice controller binds (or associates) all these slices together to form a single E2E network slice for specific customer and specific service type.
14. At the end, when the E2E network slice is created, the E2E network slice controller will allocate a unique network slice id (called S-NSSAI) and eventually during the UE network attach, all the UEs will be informed about the existence of this newly created E2E network slice and then they can request it using the 3GPP 5G signaling procedures.

Note that the interfaces 3 and 7 between the E2E network slice Controller and RAN and Core controllers and their information models are defined in various 3GPP technical specifications. However, 3GPP has not defined the same interface for transport slices, i.e. interface 10. The aim of this document is to define this interface. More details will be provided in Section 5.

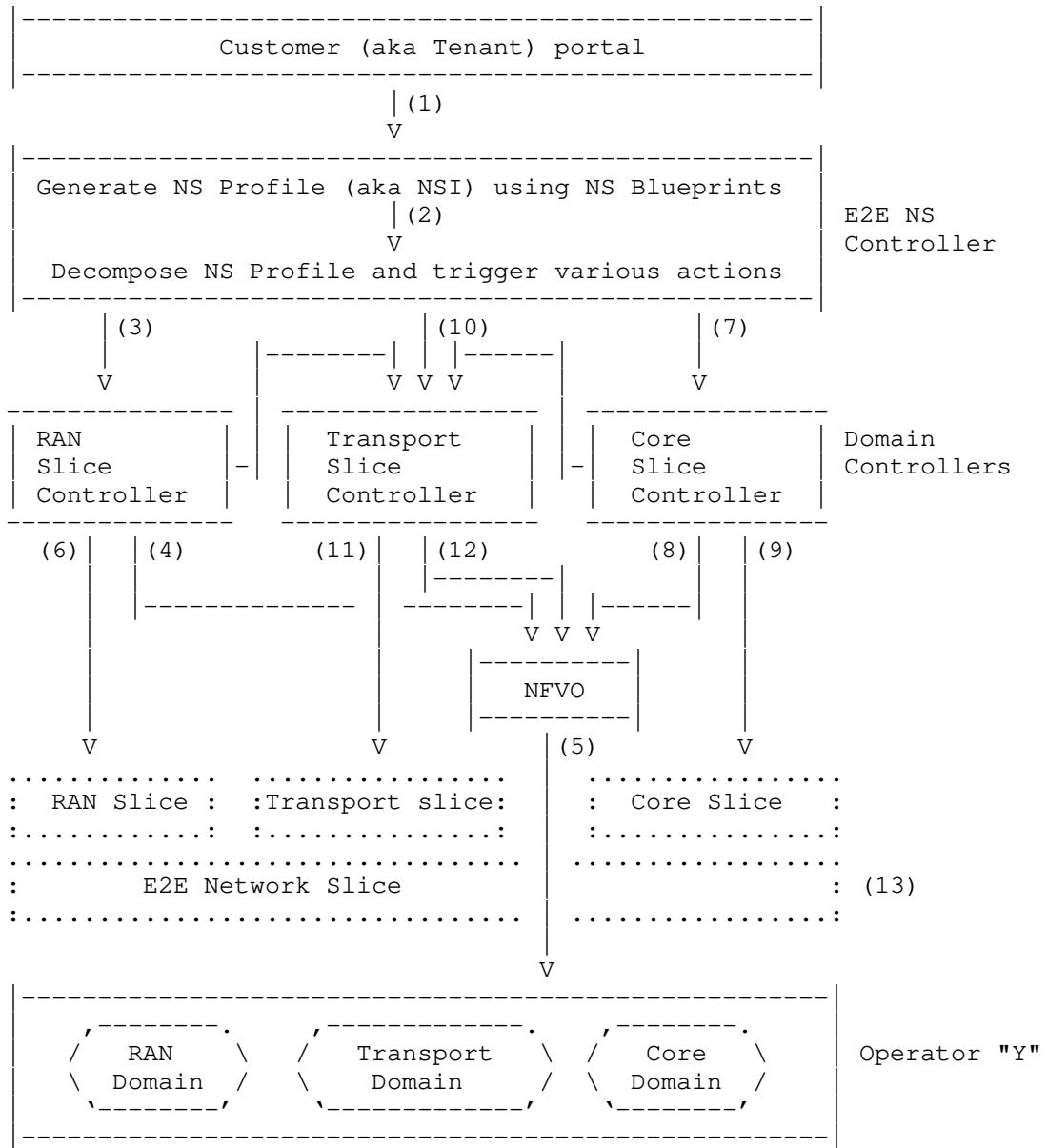


Figure 2: Logical flow cross layers for automation of an end-to-end network slices



#### 4. Definition of Transport Slice

Since the transport slice is an important concept throughout this document, this section describes this concept in more detail. To this end, consider Figure 3 where a group of physical or virtual network functions (PNF, VNF or both) are connected together. In particular, a single transport slice is defined as:

- o A distinct set of connections between multiple VNFs and PNFs each with its own deterministic SLA which is implemented (aka realized) in the network using any technology (e.g IP, Optics, Microwave and PON), any tunnel types (e.g. IP, MPLS, SR, ODU/OCH) and any L0/L1/L2/L3 service types

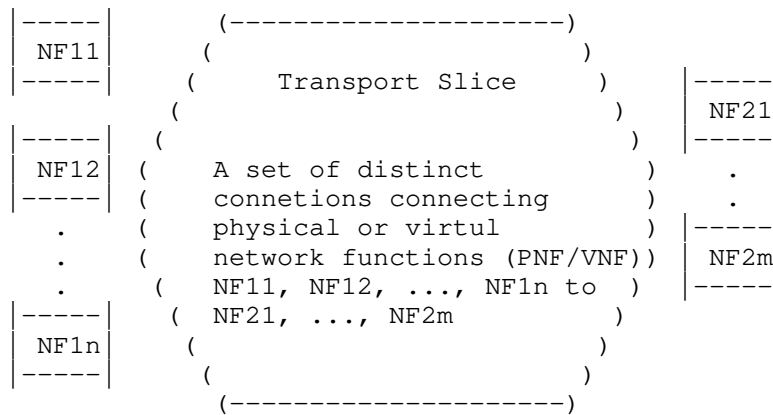


Figure 3: Definition of transport slice

For clarity, in next three sections, a few examples of the transport slices will be provided for following RAN deployments:

- o Distributed RAN
- o Centralized RAN (C-RAN)
- o Cloud RAN

##### 4.1. Transport Slices in Distributed RAN

Distributed RAN is the most common deployment of 4G and 5G RAN networks where as shown in Figure 4, the RAN network is connected to Core network using the transport network. Optionally the mobile applications can be also connected to the Core network using another

overlay network (e.g. Internet where the mobile applications are virtualized in another data center).

In this case, for a single E2E network slice, in addition to RAN and Core slices, there are two sets of transport slices; TS\_1 and TS\_2. TS\_1 is connecting the RAN to Core and TS\_2 is connecting Core to Applications.

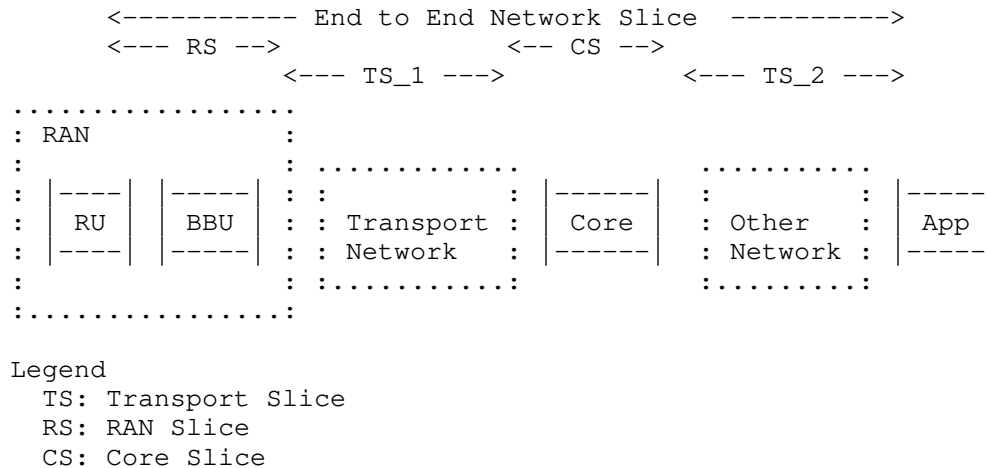
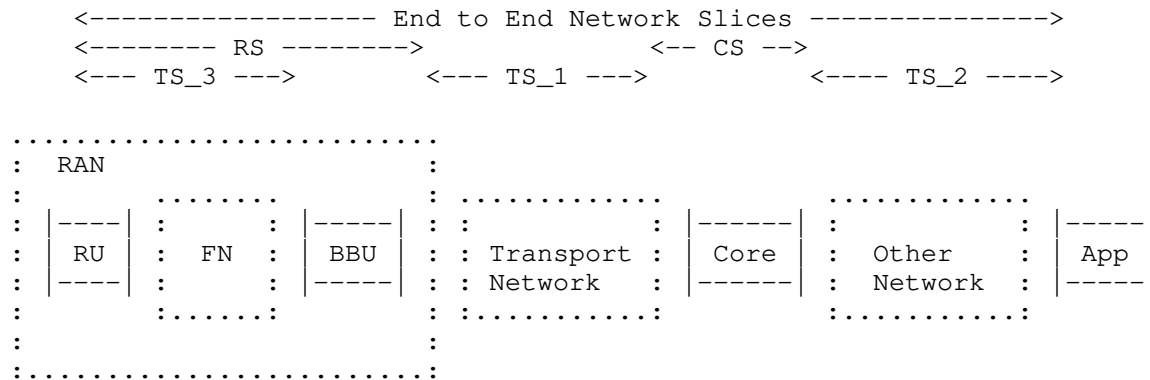


Figure 4: Transport slices in distributed RAN

#### 4.2. Transport Slices in Centralized RAN (C-RAN)

The RAN consists of two functional units, the baseband unit (BBU) and the radio unit (RU). The BBU processes the signal and is connected to the transport network. The RU transmits and receives the carrier signal that is transmitted over the air to the end user equipment (UE). In Centralized RAN (aka C-RAN) as depicted in Figure 5, the RU and BBU are separated by a network called fronthaul network. In this case a single E2E network slice contains three set of transport slices TS\_1, TS\_2 and TS\_3 where TS\_1 and TS\_2 are identical to distributed RAN case (see Figure 4) and the new TS\_3 connects the Radio Units (RU) to the BBUs.



Legend

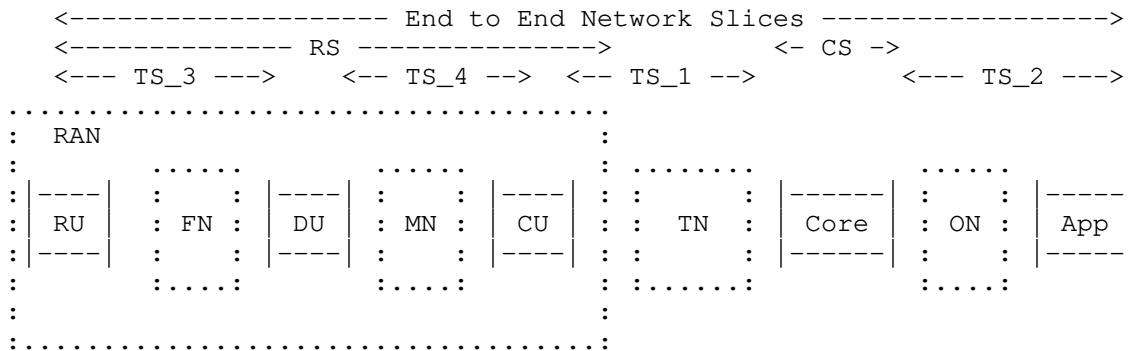
TS: Transport Slice  
RS: RAN Slice  
CS: Core Slice  
FN: Fronthaul network  
MN: Midhaul network

Figure 5: Transport slices in centralized RAN

### 4.3. Transport Slices in cloud RAN

In new cloud-RAN architecture, the baseband unit functionality is further divided into real-time and non-real-time. The former is deployed close to antenna to manages the real-time air interface resources while the non-real-time control functions are hosted centrally in the cloud. In 5G, BBU is split into two parts called CU (Central Unit) and DU (Distributed Unit) as shown in Figure 6 where these entities are connected by new network called Midhaul.

In this deployment for a single E2E network slice, there are four sets of transport slices TS\_1, TS\_2, TS\_3 and TS\_4 where TS\_1 and TS\_2 and TS\_3 are identical to distributed and centralized RAN (see Figure 4 and Figure 5). The new transport slice TS\_4 will connect DUs to CUs.



### Legend

TS: Transport Slice  
RS: RAN Slice  
CS: Core Slice  
FN: Fronthaul network  
MN: Midhaul network  
TN: Transport network  
ON: Other network

Figure 6: Transport slices in cloud RAN

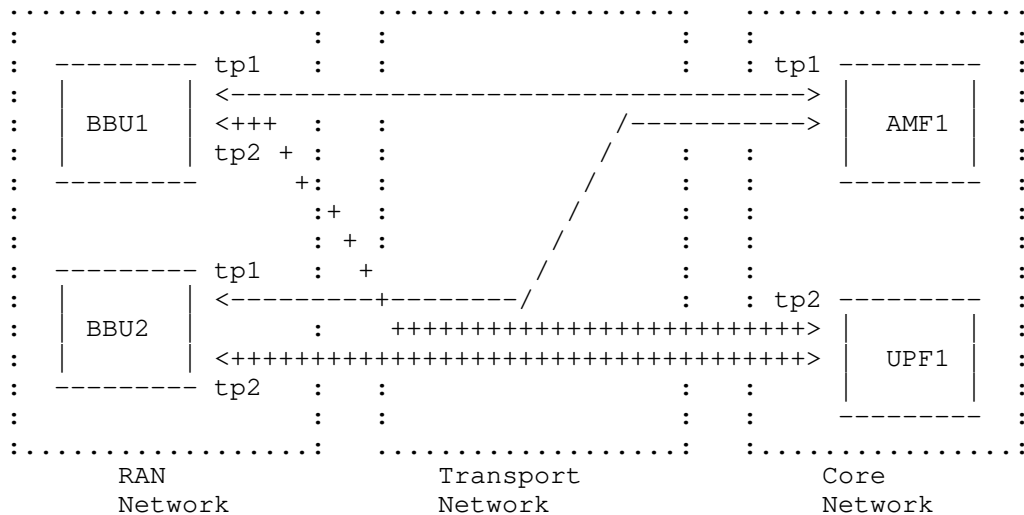
#### 4.4. Transport Slice as a set of networks

To further explore the content of a transport slice, let's focus on transport slice TS\_1 between RAN and Core. Note that the following discussion is also applicable to any other transport slices TS\_2, TS\_3 or TS\_4. As shown in Figure 7 for an individual E2E network slice belongs to a specific customer for a specific service type, the RAN domain is connected to Core domain through the transport network. In this example, the E2E network slice is identified by n-nssai=10 for customer C1 and service type Infotainment. Two RAN network elements BBU1 and BBU2 and two Core network elements AMF1 and UPF1 are all part of the E2E network slice. There are two sets of distinct connections between RAN and Core domains;

- o Network-C which connects BBU1 and BBU2 to AMF1 with SLA-C
- o Network-U which connects BBU1 and BBU2 to UPF1 with SLA-U

Customer: C1  
 Service type: Infotainment  
 S-NSSAI: 10  
 Network-C {from BBU-1.tp1, BBU-2.tp1 to AMF1.tp1 with SLA-C}  
 Network-U {from BBU-1.tp2, BBU-2.tp2 to UPF1.tp2 with SLA-U}

Transport slice TS\_1: { Network-C and Network-U }



#### Legend

tp termination point  
 ----- Network-C  
 +++++ Network-U

Figure 7: Transport Slice as a set of networks

Note that the SLA-C and SLA-U can be different. The combination of these two networks is called a single transport slice TS\_1. Note that the definition of the transport slice does not specifies how these connections should be realized (or implemented). It also does not specify which technology (e.g. IP, MPLS, Optics etc.) or tunnel type (e.g. MPLS, Segment Routing etc.) should be used during the realization. Those are part of realization of the transport slice done by transport slice controller. With this approach the definition is logically separated from implementation of transport slices. This gives a tremendous programmability and flexibility during the realization of transport slices using any type of technologies and tunnel types.

In summary, a transport slice is a distinct set of technology-agnostic connections each with its own deterministic SLA which can be implemented (aka realized) using any technology, tunnel type and service type.

## 5. Transport Slice Connectivity Interface (TSCi)

As shown in Figure 2, the interfaces 3 and 7 between the E2E network slice Controller and RAN and Core controllers, respectively and their information models are defined in various 3GPP technical specifications [TS.28.530-3GPP], [TS.28.531-3GPP], [TS.28.540-3GPP] and [TS.28.541-3GPP]. However, 3GPP has not defined the same interface for transport slices, i.e. interface 10. The aim of this document is to provide the clarification of this interface and to provide the information model of this interface. The interface is called the Transport Slice Connectivity interface (TSCi) which provides some means for automation, monitoring and optimization of the transport slices.

This new interface is needed in order to simplify the creation of the Transport slices and hides all the complexity of implementation (aka realization) from higher E2E network slice controller similar to their RAN and Core counterparts defined by 3GPP.

The aim of this document is to define a new interface and its information model between the E2E network slice controller and the transport slice controller. The characteristics on this new interface are:

- a. The interface allows a request and response about resources. It should not allow negotiation, as this would be complex and not have a clear benefit
- b. This interface is needed by the same layer that configures 3GPP RAN and Core slices to support the E2E path management in 5G networks. The provider of this interface is the higher layer controller which needs to create a connectivity between two network functions. The provider of this interface is the lower layer controller which provide the implementation (aka realization) of this connectivity (i.e. transport slice).
- c. This interface is needed in industry to achieve true standard based automation of 5G E2E network slices. It provides a technology-agnostic intent-based interface to the E2E network slice controller similar to interfaces defined by 3GPP for RAN and Core slices

- d. This interface is independent of type of network functions which needs to be connected, i.e. it is independent of any specific repository, software usage, protocol, or platform which realizes the physical or virtual network functions.
- e. The interface standardizes a way to learn about what resources are available in the network which impact the creation of the transport slices
- f. This technology independent interface simplifies the creation of the transports slices by describing it in a standard way along with all the network functions (such as eNB, gNB, CU, DU, Core, Mobile application server, etc.) that compose a transport slice, their properties, attributes and their SLA requirements, i.e. the connectivity resources requested from an E2E network slice controller to transport slice controller with their corresponding SLA requirements
- g. This interface provides capabilities for transport slice monitoring and analytics. It means that the TSCi interface allows enabling/disabling the collection of the transport slices telemetry, assurance and Threshold Crossing Alert (TCA) data and providing them to the E2E network slice controller
- h. This interface provides capabilities for optimization of the transport slices. Since the nature of an E2E network slice is dynamic, it is important to make sure the transport slice SLA are valid and in case any SLA violation happens, the transport slice controller performs the closed-loop action and inform the upper layer E2E network slice controller for the violation and closed-loop action. This interface allows this functionality.
- i. This interface allows binding and association between RAN to Transport slices and between Core to Transport slices
- j. This interface complements various IETF service, tunnel, path models by providing an abstract layer on top of these models

#### 5.1. Relationship between TSCi and various IETF data models

The transport slice connectivity interface and its relationship to various IETF data models are not addressed in any IETF WGs as it has very unique characteristics of the 5G E2E network slicing. The new interface complements various IETF service, tunnel, path data models by providing an abstract layer on top of these models.

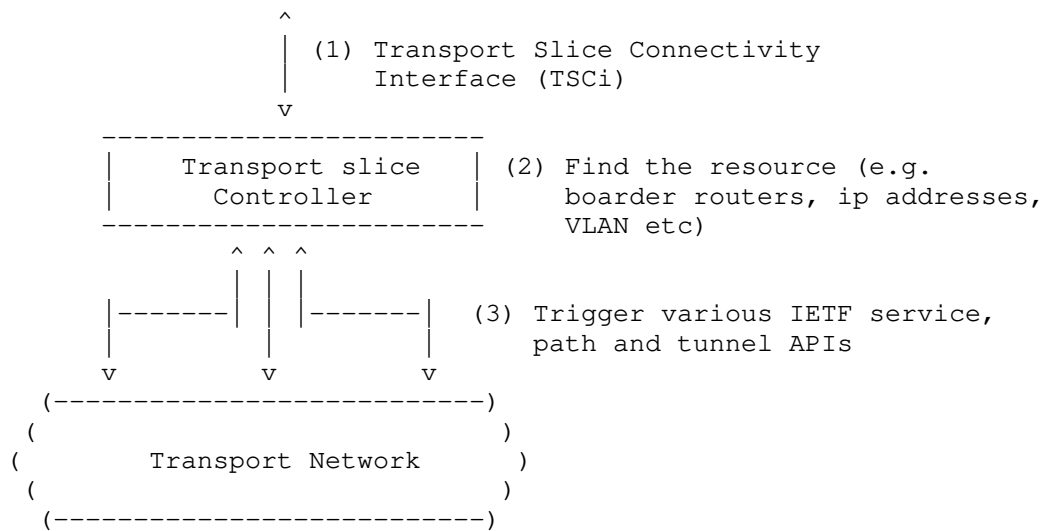


Figure 8: Relationship between transport slice interface and IETF Service/Tunnels/Path data models

Figure 8 shows more details on how the new transport slice connectivity interface (TSCi) relates to various IETF service/tunnels/path models. The transport slice controller receives a request for creation of a transport slice. This request is an abstract intent-based request which contains the required connections between various network functions in RAN and Core. The transport slice controller will then realize or implement those connections using various IETF models.

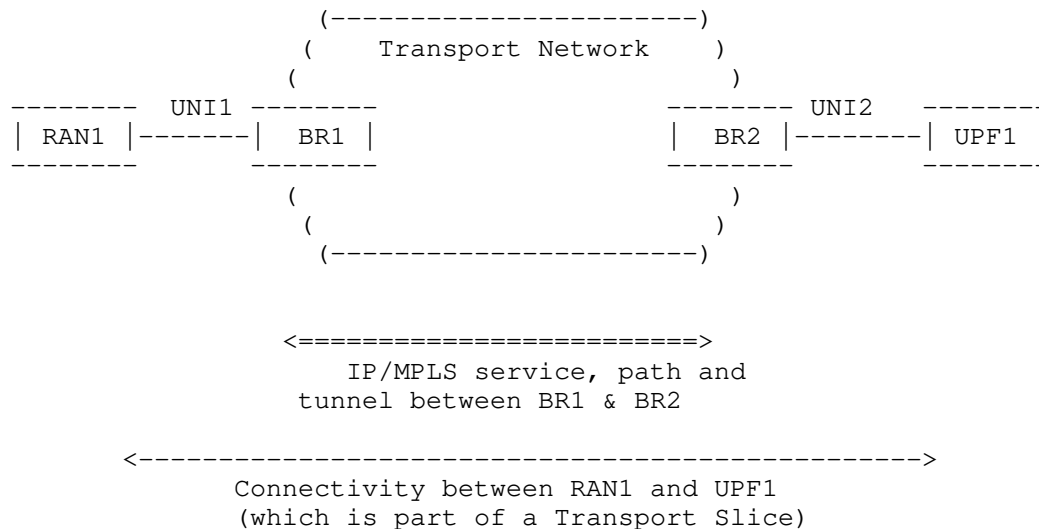
In a sense, the new transport slice connectivity interface provides an abstract layer on top of IETF models. The IETF service, path and tunnel data models can be any existing IETF service models such as L3SM or L2SM ([RFC8049] and [RFC8466]). It also can be any future data models.

## 5.2. Realization (aka Implementation) of transport slices

Since the transport slice connectivity interface describes the connections not the services, it is essential to make a distinction between connections and implemented services. Referring to Figure 9, assume using the new transport slice interface, the E2E network slice controller requests the creation of a transport slice which has multiple connections between RAN and Core. One of these connections is shown below between RAN1 and UPF1. The E2E network slice controller can request a connection between RAN1 to UPF1 for a



specific tenant, specific service type and specific SLA (e.g. customer Public Safety for service CCTV with latency of 5 [ms] or better).



**Legend:**

<----> Connection part of the transport slice

<====> Implementation (aka realization) of the transport slice

Figure 9: Distinction between Connections and Services

To realize (aka implement) this connection, the transport slice controller, will find the endpoint for the L0/L1/L2/L3 services, find the best path and create a service between these endpoints. In this example, in order to implement the connection between RAN1 and UPF1, the transport slice controller will first find the best boarder routers BR1 and BR2, find the best path between them and finally creates a Service/path from BR1 to BR2. It is important to note that:

- o The endpoints of the Connection are different from the endpoints of the Services, paths and tunnels. This is the unique characteristic of transport slices where the endpoints of the connections are different from endpoints of the Services (i.e. endpoint of the transport slices are RAN1 and UPF1 whereas the endpoint of services are BR1 and BR2)

- o The Service/path API can be any existing IETF service models such as L3SM or L2SM ([RFC8049] and [RFC8466]). It also can be any future service model
- o In order to have the connectivity between RAN1 and UPF1, the RAN and Core slices should be associated to Transport slice. This is also a by-product of the Transport slice connectivity interface when all allocated resources for access points (such as allocated VLAN IDs, IP addresses etc) are conveyed to RAN and Core Slices. This will be done by coordination between transport slice controller and E2E network slice controller.

6. Transport slice connectivity interface (TSCi) information model

Based on the definition of a transport slice (see Section 4), the high-level information model of a transport slice connectivity interface should conform with Figure 10:

```

module: transport-slice-connectivity
+--rw transport-slice
  +--rw transport-slice-info
    +--rw ts-id
    +--rw ts-name
    +--...
  +--rw network-slice-info [ns-id]
    +--rw list of s-nnsai (i.e. E2E network slice id)
    +--rw customer (aka tenant)
    +--rw service type (e.g. CCTV, infotainment etc)
    +--rw NS IDs (i.e. list of S-NSSAI)
    +--...
  +--rw transport-slice-networks* [network-id]
    +--rw network-id
    +--...
    +--rw node* [node-id]
      +--rw node-id
      +--...
    +--rw connection-link* [link-id]
      +--rw link-id
      +--rw endpoint-A
      +--rw endpoint-B
      +--...
    +--rw transport-slice-policy* [policy-id]
      +--rw policy-id
      +--rw policy-type (e.g. sla-policy, selection-policy,
                        assurance-policy)
      +--...

```

Figure 10: High-level information model of transport slice connectivity interface

The proposed transport slice information model should include the following building blocks:

- o transport-slice-info: Information related to transport slice
- o network-slice-info: A list of all E2E network slices mapped to transport slice
- o transport-slice-networks: Each transport slice is a set of networks. Each network contains:
  - \* list of nodes (i.e. list of RAN and Core network functions)
  - \* list of connection-links (i.e. list of connections between nodes)

- \* list of transport-slice-policies (i.e. various SLA, Selection and Monitoring policies)

#### 6.1. transport-slice-info

It contains information such as transport slice name, transport slice ID etc. The details will be provided in next version of this draft.

#### 6.2. network-slice-info

As discussed in Section 3, a request for creation of a transport slice starts with the fact that a customer (aka tenant) will request an E2E network slice from an operator for a certain service type (e.g. CCTV, Infotainment, URLLC, etc.). So, there is a mapping between transport slice and the E2E network slice. Depending on the deployment, it is possible to map multiple E2E network slice to a transport slice, i.e. the mapping between transport slice to E2E network slice is one to many.

The network-slice-info contains the list of E2E network slices which are mapped to the transport slice with all relevant information such as S-NSSAI, name of customer, service type etc. The details will be provided in next version of this draft.

#### 6.3. transport-slice-networks

As per Section 4, a transport slice will contain one or more transport-slice-networks.

#### 6.4. node

As discussed in Section 4, the transport slice comprises one or more connectivity networks between RAN and Core network elements. It is also possible to have the connectivity networks between RAN and RAN network elements for some RAN deployments (example for midhaul network). As discussed in section 2.2, when the transport slice is realized (implemented), the network elements which are the endpoints of realization of the transport slice might be different. As a result, the nodes in this model represent RAN or Core network elements. However, the model is flexible where nodes might represent the routers or switches which are the endpoints of the transport slice realization.

The attributes of the node are IP address, node name, domain ID and termination points. The details will be provided in next version of this draft.

#### 6.5. connection-link

Each transport-slice-networks contain one or more connections between various nodes described in Section 6.4. The connection-link is a list of links which are represented by endpoint-A, endpoint-B etc. The details will be provided in next version of this draft.

#### 6.6. transport-slice-policy

To establish a transport slice, one or more transport-slice-networks will be created each with certain SLA requirement which is represented by transport-slice-policy. This draft proposes three types of transport slice policies to be supported:

- o transport-slice-sla-policy
- o transport-slice-selection-policy
- o transport-slice-assurance-policy

The summary of these policies will be provided here. The details will be provided in next version of this draft.

##### 6.6.1. transport-slice-sla-policy

This is a mandatory policy. The transport-slice-policy represents in a generic and technology-agnostics way the SLA requirement needed to realize a group of connections which are part of a transport slice. It is defined per transport-slice-network. It contains the bounded latency, bandwidth, reliability, security etc.

##### 6.6.2. transport-slice-selection-policy

This is an optional policy. In some deployment, the E2E network slice controller might want to assist the transport slice controller on how to realize a transport slice by providing some information regarding the type of technologies and tunnels. This information will be provided in transport-slice-selection-policy.

##### 6.6.3. transport-slice-assurance-policy

This is a mandatory policy. The E2E network slice controller shall influence the transport slice controller for transport slice assurance on how to perform monitoring, analytics and optimization. To this end, the transport-slice-assurance-policy will be used. It contains, the type of assurance needed, time interval, how often inform the E2E network slice controller etc.

## 6.7. IETF models

Currently none of the IETF data models address the modeling of transport slice connectivity as shown in Figure 6. However, the various IETF data models might be augmented to address the information model of the transport slice connectivity interface. Following is the list of candidates IETF YANG models. This is not an extensive list and the next version of the draft will provide a more comprehensive list.

### 6.7.1. ACTN model

As defined in [RFC8453], [I-D.king-teas-applicability-actn-slicing] and [I-D.ietf-teas-actn-vn-yang] a VN (Virtual Network) is the abstract customer view of the TE network. The VN can be seen as a set of edge-to-edge abstract links (a Type 1 VN). Each abstract link is referred to as a VN member and is formed as an E2E tunnel across the underlying networks.

This definition is very similar to definition of the transport slice which means that the VN YANG model can be augmented to address the modeling of the transport slice shown in Figure 6. This is work in progress for next version of this document.

### 6.7.2. i2rs model

Similar to [I-D.qiang-coms-netslicing-information-model], the data model for network topologies developed in [[I-D.ietf-i2rs-yang-network-topo] can be augmented to address the modeling of the transport slice. This is work in progress for next version of this document

## 7. IANA Considerations

This memo includes no request to IANA.

## 8. Security Considerations

TBD

## 9. Informative References

[I-D.boucadair-connectivity-provisioning-protocol]  
Boucadair, M., Jacquenet, C., Zhang, D., and P. Georgatsos, "Connectivity Provisioning Negotiation Protocol (CPNP)", draft-boucadair-connectivity-provisioning-protocol-15 (work in progress), December 2017.

- [I-D.homma-slice-provision-models]  
Homma, S., Nishihara, H., Miyasaka, T., Galis, A., OV, V., Lopez, D., Contreras, L., Ordonez-Lucena, J., Martinez-Julia, P., Qiang, L., Rokui, R., Ciavaglia, L., and X. Foy, "Network Slice Provision Models", draft-homma-slice-provision-models-00 (work in progress), February 2019.
- [I-D.ietf-i2rs-yang-network-topo]  
Clemm, A., Medved, J., Varga, R., Bahadur, N., Ananthakrishnan, H., and X. Liu, "A Data Model for Network Topologies", draft-ietf-i2rs-yang-network-topo-20 (work in progress), December 2017.
- [I-D.ietf-teas-actn-vn-yang]  
Lee, Y., Dhody, D., Ceccarelli, D., Bryskin, I., Yoon, B., Wu, Q., and P. Park, "A Yang Data Model for VN Operation", draft-ietf-teas-actn-vn-yang-04 (work in progress), February 2019.
- [I-D.king-teas-applicability-actn-slicing]  
King, D. and Y. Lee, "Applicability of Abstraction and Control of Traffic Engineered Networks (ACTN) to Network Slicing", draft-king-teas-applicability-actn-slicing-04 (work in progress), October 2018.
- [I-D.qiang-coms-netslicing-information-model]  
Qiang, L., Galis, A., Geng, L., kiran.makhijani@huawei.com, k., Martinez-Julia, P., Flinck, H., and X. Foy, "Technology Independent Information Model for Network Slicing", draft-qiang-coms-netslicing-information-model-02 (work in progress), January 2018.
- [RFC7297] Boucadair, M., Jacquenet, C., and N. Wang, "IP Connectivity Provisioning Profile (CPP)", RFC 7297, DOI 10.17487/RFC7297, July 2014, <<https://www.rfc-editor.org/info/rfc7297>>.
- [RFC8049] Litkowski, S., Tomotaki, L., and K. Ogaki, "YANG Data Model for L3VPN Service Delivery", RFC 8049, DOI 10.17487/RFC8049, February 2017, <<https://www.rfc-editor.org/info/rfc8049>>.
- [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>>.

[RFC8466] Wen, B., Fioccola, G., Ed., Xie, C., and L. Jalil, "A YANG Data Model for Layer 2 Virtual Private Network (L2VPN) Service Delivery", RFC 8466, DOI 10.17487/RFC8466, October 2018, <<https://www.rfc-editor.org/info/rfc8466>>.

[TS.28.530-3GPP]

3rd Generation Partnership Project (3GPP), "3GPP TS 28.530 V15.1.0 Technical Specification Group Services and System Aspects; Management and orchestration; Concepts, use cases and requirements (Release 15)", December 2018, <[http://ftp.3gpp.org//Specs/archive/28\\_series/28.530/28530-f10.zip](http://ftp.3gpp.org//Specs/archive/28_series/28.530/28530-f10.zip)>.

[TS.28.531-3GPP]

3rd Generation Partnership Project (3GPP), "3GPP TS 28.531 V16.2.0 Technical Specification Group Services and System Aspects; Management and orchestration; Provisioning; (Release 16)", June 2019, <[http://ftp.3gpp.org//Specs/archive/28\\_series/28.531/28531-g20.zip](http://ftp.3gpp.org//Specs/archive/28_series/28.531/28531-g20.zip)>.

[TS.28.540-3GPP]

3rd Generation Partnership Project (3GPP), "3GPP TS 28.540 V16.0.0 Technical Specification Group Services and System Aspects; Management and orchestration; 5G Network Resource Model (NRM); Stage 1 (Release 16)", June 2019, <[https://www.3gpp.org/ftp/Specs/archive/28\\_series/28.540/28540-g00.zip](https://www.3gpp.org/ftp/Specs/archive/28_series/28.540/28540-g00.zip)>.

[TS.28.541-3GPP]

3rd Generation Partnership Project (3GPP), "3GPP TS 28.541 V16.1.0 Technical Specification Group Services and System Aspects; Management and orchestration; 5G Network Resource Model (NRM); Stage 2 and stage 3 (Release 16)", June 2019, <[http://www.3gpp.org/ftp//Specs/archive/28\\_series/28.541/28541-g10.zip](http://www.3gpp.org/ftp//Specs/archive/28_series/28.541/28541-g10.zip)>.

#### Authors' Addresses

Reza Rokui  
Nokia  
Canada

Email: [reza.rokui@nokia.com](mailto:reza.rokui@nokia.com)



Shunsuke Homma  
NTT  
3-9-11, Midori-cho  
Musashino-shi, Tokyo 180-8585  
Japan

Email: homma.shunsuke@lab.ntt.co.jp

Diego R. Lopez  
Telefonica I+D  
Spain

Email: diego.r.lopez@telefonica.com

Xavier de Foy  
InterDigital Inc.  
Canada

Email: Xavier.Defoy@InterDigital.com

Luis M. Contreras-Murillo  
Telefonica I+D  
Spain

Email: luismiguel.contrerasmurillo@telefonica.com

Jose J. Ordonez-Lucena  
Telefonica I+D  
Spain

Email: joseantonio.ordonezlucena@telefonica.com

Pedro Martinez-Julia  
NICT  
Japan

Email: pedro@nict.go.jp

Mohamed Boucadair  
Orange  
France

Email: mohamed.boucadair@orange.com

Philip Eardley  
BT  
UK

Email: philip.eardley@bt.com

Kiran Makhijani  
Futurewei Networks  
US

Email: kiranm@futurewei.com

Hannu Flinck  
Nokia  
Finland

Email: hannu.flinck@nokia-bell-labs.com